Politecnico di Torino

Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Aerospaziale

A.a 2022/2023

Sessione di Laurea Dicembre 2023

Data-driven Aerodynamic Shape Optimisation for Morphing Configurations

RELATORI:
Domenic D'AMBROSIO
Fernando Jose Parraucho LAU
Afzal SULEMAN

CANDIDATI:
Sara PUCCIARELLI

To myself and my dreams...

Acknowledgments

I wish to extend my heartfelt thanks to the people who have played a significant role in the coronation of my master's degree. Their support and presence throughout this journey have proven to be truly invaluable.

First and foremost, I want to thank my supervisors, Professors Fernando and Afzal and Professor Domenic. Your assistance and support have been invaluable for my personal and professional growth. Your guidance and feedback have led my research and your assistance has been of crucial importance in shaping this thesis. I am genuinely thankful for the trust you've placed in me and for the remarkable opportunities you've provided.

I would also like to express my gratitude to the entire Department of Aerospace Engineering of the Instituto Técnico Lisboa, particularly to Processor Frederico whose support has been an inspiration throughout the entire process. I'm truly grateful for your patience and for the knowledge you have shared with me. Furthermore, I wish to extend my earnest gratitude to Bernardo and all the wonderful guys in our office. Our friendship will forever hold a special place in my heart.

I want to express my deep gratitude to Mirko and Alessandra, who have been by my side during one of the most pivotal years of my life, sharing in every significant moment of this journey. I hold great affection for both of you, and I am thankful for all the cherished memories we've created together.

A special thought is to Watson and Aurora for being my shoulder, my soulmates and my home. I promise to cherish our friendship forever.

To Martino and Valeria, I'm grateful for growing up together. Our support, laughter, and deep emotions have made me the person I am today. I am truly thankful for that.

I want to send a heartfelt kiss to Manuela, Lety, and Rosy. Having you in my life makes me so happy, and I hope we can always find time to get together and share our lives over a glass of wine. Let's continue to support each other with kindness and positivity.

To Hugo, I want to convey my genuine gratefulness for your presence and support over the past few months. Your love and sense of humour have illuminated even the most challenging moments.

To my parents, I want to express my deepest gratitude for everything you've done. You've imparted priceless values and lessons, supporting my decisions and fueling my dreams. Your presence and our respect are two of my life's most treasured aspects. I can never sufficiently thank you for always being there, even at the other end of the phone, and for being my daily inspiration.

Abstract

Nowadays, morphing aerofoils present a promising strategy to achieve greening aviation objectives. However, due to the numerous evaluations needed for the design of morphing shapes, gradient and CFD-based Aerodynamic Shape Optimisation (ASO) is prohibitive in terms of computational cost. Surrogate-based optimisation frameworks have been proposed to strike a balance between accuracy and computational efficiency. Given the complexity related to morphing strategies, their application is recommended for a mission profile comprising several flight conditions. This forces the researchers to constrain both the flight envelope and the design space of interest. While this approach enhances performance, the generalisation and applicability of the model are constrained.

This thesis presents an efficient data-driven framework for the ASO of two-dimensional morphing aerofoils, comprising both subsonic and transonic regimes. With the aim to address the limits of the existing deep-learning models, an extensive database comprising more than 140,000 samples, 1,200 aerofoil geometries and 120 flight conditions, has been formulated and used to train the network employed to accomplish the aerodynamic computations. Coupling the model with free-form deformation and genetic algorithm, significant drag reduction for morphing configurations is achieved, preserving the structural integrity of a wing box. However, there are important discrepancies in the model predictions, particularly in subsonic flight conditions, which reveal challenges in learning the underlying physics of the field. Summarising, the framework developed shows promise, displaying convergence and efficiency, and establishing a solid foundation for future research.

Keywords: Aerodynamic shape optimisation, machine learning, morphing architectures, aircraft design, performance

Contents

List of Tables

List of Figures

Acronyms

Adagrad Adaptive Gradient Descent.

Adam Adaptive Moment Estimation.

AI Artificial Intelligence.

ANK Approximate Newton-Krylov algorithm.

ANN Artificial Neural Network.

AoA Angle of Attack.

ASO Aerodynamic Shape Optimisation.

CFD Computational Fluid Dynamics.

CGNS CFD Generated Notation System.

CNN Convolutional Neural Network.

CST Class Function Transformation.

DL Deep Learning.

DoE Design of Experiments.

DoF Degree of Freedom.

DRL Deep Reinforcment Learning.

DV Design Variable.

FD Finite Difference Method.

FFD Free Form Deformation.

FVM Finite Volume Method.

GA Genetic Algorithm.

GAN Generative Adversarial Network.

KNN K-nearest Neighbours.

LE Leading Edge.

LHS Latin Hypercube Sampling.

MAE Mean Absolute Error.

ME Mixture of Expertise.

MDO Multidisciplinary Design Optimisation.

ML Machine Learning.

MLP Multilayer Perceptron.

MOO Multi-Objective Optimisation.

MSE Mean Square Error.

NACA National Advisory Committee for Aeronautics.

NK Newton-Krylov.

NAG Nesterov Momentum.

NASA National Aeronautics and Space Administration.

NSGA-II Non-dominated Sorting Genetic Algorithm.

PC Polynomial Chaos.

PCA Principal Component Analysis.

PCE Polynomial Chaos Expansion.

RANS Reynolds-averaged Navier–Stokes equations.

RBF Radial Basis Functions.

RE Richardon Extrapolation.

ReLU Rectified Linear Unit.

RL Reinforcment Learning.

RMSE Root Mean Square Error.

RMSProp Root Mean Square Propagation.

RNN Recurrent Neural Network.

R2 R-squared Metrics.

SA Spallart-Allmaras.

SBO Surrogate-based Optimisation.

SBX Simulated Binary Crossover.

SGD Stochastic Gradient Descent.

SLSQP Sequential Least Squares Programming Algorithm.

SM Surrogate Model.

SOO Single-Objective Optimisation.

SVD Singular Value Decomposition.

SVM Support Vector Machine.

TE Trailing Edge.

TL Transfer Learning.

UIUC University of Illinois Urbana-Champaign.

UAV Unmanned Aerial Vehicle.

Nomenclature

$\mathbf{A}$	A matrix.
$\mathbf{a}$	A vector.
$\mathcal{A}$	A generic function.
a	A scalar.

Greek symbols

α	Angle of attack.
β	Penalty multiplicative factor.
Δ	Difference.
δ	Generic scalar.
ϵ	Relative error.
γ	Specific heat ratio.
λ	Transonic shock.
μ	Molecular dynamic viscosity.
∇	Gradient.
ν	Molecular kinematic viscosity.
Ω	Design space.
ϕ	Velocity potential.
ψ	Grid solution value.
ρ	Density.
τ	Weighting function.
Θ	Activation function.
θ	Network hyperparameter.

Roman symbols

a	Speed of sound.
b	Bias term.
$\mathcal{C}$	Generic constraint equation.
c	Aerofoil chords.
C_d	Drag coefficient.
C_l	Lift coefficient.
C_m	Moment coefficient.
C_p	Pressure coefficient.
C_{d_0}	Viscous drag coefficient.
$C_{p,0}$	Incompressible pressure coefficient.
d	Design variable.
D	Drag force.
D	Dimention flow domain.
$\boldsymbol{f}$	Objective function in multi-objective optimisation.
$\mathcal{F}$	Governing flow equation.
f	Generic function.
$\boldsymbol{g}$	Objective function in multi-point optimisation.
h	Altitude.
$\mathcal{J}$	Generic aerodynamic objective function.
J	Cost function.
K	Neurons per hidden layer.
k_{L}	Lift induced drag coefficient.
$\mathcal{L}$	Loss function per instance.
L	Lift force.
M	Pitch moment.
M	Mach number.
m, n	Generic scalars.

M_{cr} Critical Mach number.

$\mathrm{N_{offwall}}$ Number of extruded layer.

$\mathrm{N_{stream}}$ Number of points discretising an aerofoil.

N Number of hidden layers.

N Number of examples.

N_c Number of constrained equations.

N_{pop} Population size.

o Output activation function.

$\mathcal{P}$ Penalty function.

p Static pressure.

Q Conservative flow variable.

q Dynamic pressure.

$\mathbb{R}$ Sets of all real numbers.

r Grid refinement ratio.

R^2 Coefficient of determination.

Re Reynolds number.

s Convergence rate.

t Aerofoil thickness.

u Layer pre-activation.

u_τ Friction velocity.

V Velocity.

v Layer activation

w Weight term.

x Input.

x, y Cartesian coordinates.

y Output.

y^+ Dimensionless wall distance.

Subscripts

∞	Free-stream condition.
$\mathcal{L}^0$	$\mathcal{L}^0$ norm.
$\mathcal{L}^1$	$\mathcal{L}^1$ norm.
$\mathcal{L}^2$	$\mathcal{L}^2$ norm.
GA	Multi-point optimisation related.
i, j, k	Computational indexes.
L	Lower surface.
l	Lower boundary.
sub	Climb phase related.
$trans$	Cruise phase related.
U	Upper surface.
u	Upper boundary.

Superscripts

$*$	Optimal.
$\bar{}$	Average.
$\frown$	Penalty added function.
$\hat{}$	Perturbation.
$\tilde{}$	Predicted.
i, k	Computational indexes.

Chapter 1

Introduction

1.1 Motivation

Conventional approaches to product and engineering design prioritise a human-centred approach, demanding the employment of expertise that encompasses scientific, intuitive, experiential, and creative methods [1]. Artificial Intelligence (AI) has slightly changed this conventional approach by supporting the engineers in the decision-making process [2].

The AI-supported techniques can manage complex design operations such as comparison, evaluation and estimation, thus alleviating the workload of the designer who is allowed to focus on the creative and innovative part of the project. Moreover, AI has convincingly addressed some concerning issues, such as big data processing and high computational capability. Among the numerous AI techniques available, expert systems [3], fuzzy logic [4], artificial neural networks [5–7], and genetic algorithms [8] have traditionally held the status of being the most frequently employed classical methods for the evaluation and optimisation phases in design processes. However, in recent years, advances in data science and in parallel computing hardware have led to the escalation of data-driven modern approaches such as machine learning (ML) and deep learning (DL). This trend is depicted in Figure 1.1.

Especially DL models have proved to achieve remarkable performance in many engineering tasks due to their capacity to solve complex and undefined problems without any kind of human supervision [2, 5–7, 9]. Consequently, DL has become highly suitable for surrogate-based Aerodynamic Shape Optimisation (ASO). By leveraging DL models, one can assess various design alternatives with remarkable efficiency, eliminating the requirement for expensive and time-consuming simulations. This leads to the creation of a real-time, interactive tool for optimising the shapes of aircraft, wind turbines, and other aerodynamic systems. These applications play a pivotal role in substantially shortening the development cycle of aircraft while simultaneously enhancing design performance [10].

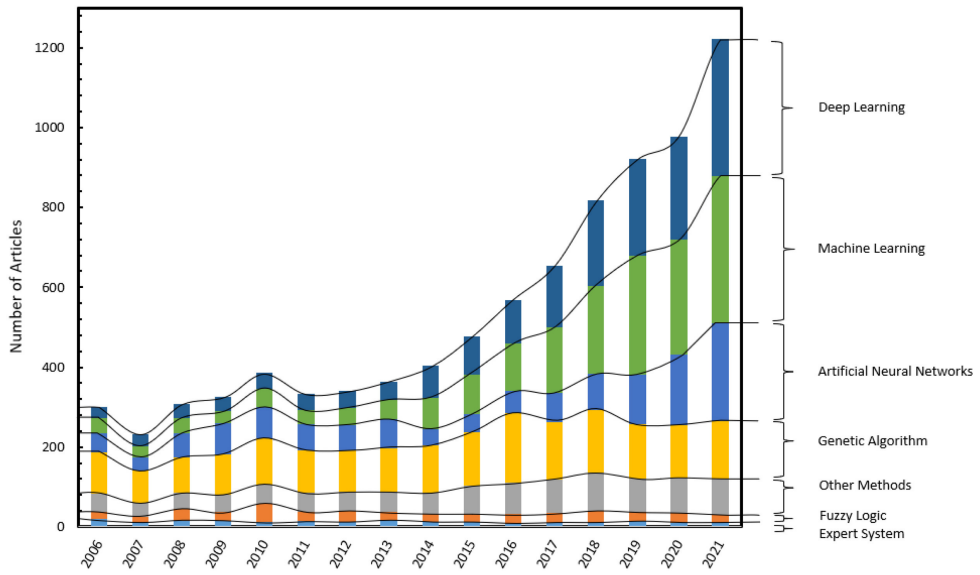


Figure 1.1: Number of articles published per year on different AI methods specifically used in design applications between the years 2006 and 2021 [2].

1.2 Problem Statement

The current trend in the aeronautical industry is to increase the efficiency of the aircraft through all phases of the mission profile to obtain better performances such as flight envelope, flight control and flight range [11]. Nowadays, aviation architectures are thought and developed on a fixed geometry equipped with control surfaces, selected from a trade-off study. Morphing wing architectures, by adapting their shape to each flight condition, present a potential solution to address the increasingly restrictive designing criteria [12–14], such as :

- Reduction of development cycle time and cost;
- Flight Safety;
- Greening objectives.

In this context, ASO is called to identify the most suitable solution, enhancing aerodynamic performance under all the prescribed constraints.

ASO has indeed emerged as a valuable tool for aerodynamic designers, enabling them to shape lifting surfaces and other components important for managing lift and drag forces. When integrated with Computational Fluid Dynamics (CFD), ASO assumes indeed a crucial role in contemporary aircraft design [10]. The gradient-based approach with the derivatives computed by the adjoint method [15, 16] is currently adopted by companies for its high reliability even though it demands high computational expenses, due to the iterative nature of the process. Nevertheless, because of the iterative and costly simulation-based evaluation within optimisation steps, ASO still cannot effectively satisfy some practical demands such as real-time and many-query applications [10].

Because of this, surrogate models (SMs) have been increasingly used in lieu of expensive full-order simulations [17]. Numerous optimisation techniques for aerodynamic design utilise surrogate models,

and these approaches heavily depend on an iterative refinement process during the surrogate definition. This iterative refinement plays a pivotal role in enhancing both the precision and efficiency of the optimisation procedure [18].

In the last years, the optimisation efficiency of these methods has been improved due to the development of DL techniques [10]. Even though many types of surrogate models exist in the literature such as Kriging [19], radial basis functions (RBF) [17] or polynomial regression [17], DL is increasingly employed for both single and multi-objective optimisation.

Due to the availability of aerodynamic data, DL has opened the door to developing data-driven surrogate models, often referred to as meta-models. These meta-models serve as efficient alternatives to the expensive high-fidelity simulations that are irreplaceable in the traditional approach [10]. The use of DL surrogates in ASO is driven by their capability to address intricate geometries. They are proficient in managing nonlinear, high-dimensional data, facilitating a substantial number of design parameters. Additionally, DL surrogates can handle constraints, accommodate multiple objectives, and offer an efficient optimisation process. This efficiency arises from the fact that DL models can be trained using high-fidelity simulation-based experiments or existing wind tunnel data, enabling the generation of sufficiently accurate solutions [20].

1.3 State-of-the-art

ASO is a closed loop composed of an optimisation model, an optimiser and an evaluation workflow aiming to find the solution which best minimises the objective function [21]. With reference to the notation used by Nemec et al. [22], the aerodynamic shape optimisation problem formally consists of determining the values of design variables, $\boldsymbol{d}$, such that the objective function, $\mathcal{J}$, is minimised:

$$\min_{\boldsymbol{d}} \mathcal{J}(\boldsymbol{d}, Q), \tag{1.1}$$

subject to constraints equations $\mathcal{C}$,

$$\mathcal{C}_j(\boldsymbol{d}, Q) \leq 0, \tag{1.2}$$

for each $j = 1, ..., N_c$.

Here Q represents the conservative flow variables and N_c denotes the number of constrained equations. In addition, the flow variables are forced to satisfy the governing flow equation, $\mathcal{F}$:

$$\mathcal{F}(\boldsymbol{d}, Q) = 0, \tag{1.3}$$

within a predetermined region of the design space Ω, such that the equations are verified for each $\boldsymbol{d} \in \Omega$.

In the ASO context, the objective function is the aerodynamic performance, namely the drag coefficient, C_d, or the lift-to-drag ratio [10]. The minimisation problem is conducted for several flight conditions, usually defined by the Mach and the Reynolds numbers, with respect to all the constraints as mentioned in Equations 1.1 - 1.3.

The initial set $\boldsymbol{d}$ of design variables (DVs) are defined by the shape parameterisation method, including commonly the angle of attack, α, to satisfy the lift constraints. The DVs are accurately defined during the geometry parameterisation stage, detailed in Section 1.3.2. Furthermore, the constraints imposed are both geometric (such as the thickness and the volume constraints) and aerodynamic (such as the target lift coefficient, C_l*) [22].

As it was mentioned, the most important breakthrough in ASO has been the gradient-based optimisation, coupled with the adjoint method for the computation of the derivatives. The adjoint method, pioneered by Antony Jameson [15, 23], made it possible to efficiently compute the aerodynamic gradients in high dimensional space. The gradient-based algorithms are the most suitable for addressing problems characterised by a high dimensionality of the design space and an expensive computation of the aerodynamic forces. Nonetheless, achieving a precise and efficient computation of derivatives is essential, since the optimisation tasks may encounter irregularities and find difficulties in catching the local minima within the design space [24].

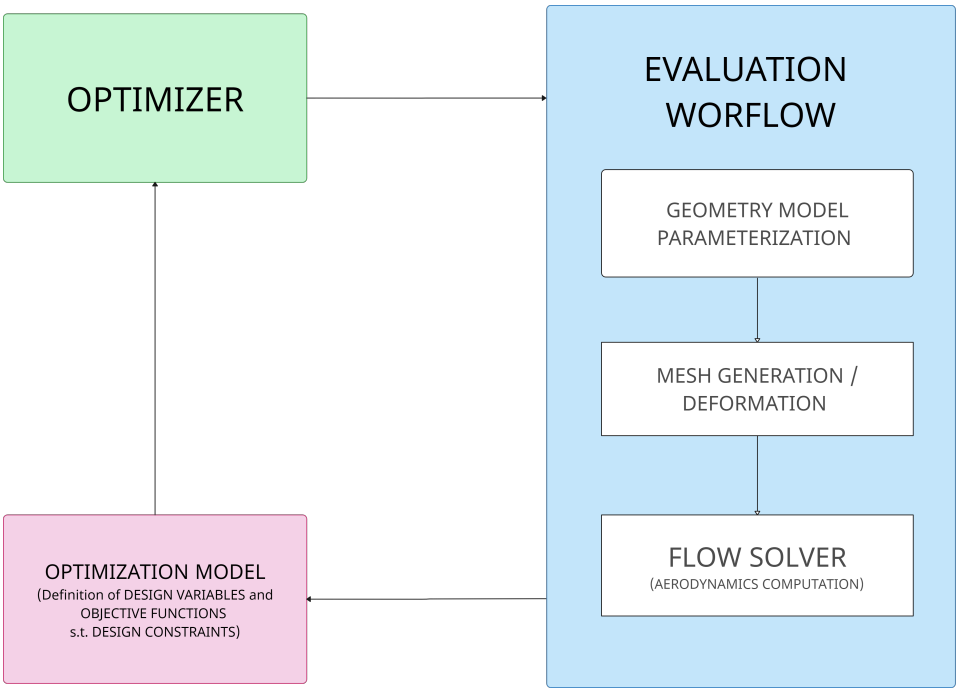


Figure 1.2: Gradient and CFD-based ASO framework.

The standard ASO is sketched in Figure 1.2. In the pre-processing phase, the CFD mesh and the geometric parametrisation are established. Therefore the objective function, the constraints and the bounded design variables are wrapped in the optimisation model. Once the baseline problem is set up, the solution is iteratively found. In each iteration, both the optimiser and the evaluation workflow are called. The gradient-based optimiser updates the design variables with respect to the constraints and the derivatives. Based on the updated design variables the shape is deformed and consequently the mesh, in order to perform the CFD analysis. The workflow output is sent back to the optimiser which uses the feedback to determine the search direction and step length of the next iteration [10, 21]. This

framework is implemented in many open-source codes such as MACH-Aero[1] and SU2[2].

Enhanced by the adjoint method, CFD-based ASO is widely used for aerofoil optimisation [22, 25, 26] and wing optimisation [27–29]. However, several issues have motivated the designers to find alternative solutions:

- The CFD solver demands additional robustness. This issue arises because the solver lacks the intuition of the designer, making it susceptible to potential failures during optimisation. Consequently, the optimisation algorithm can yield sub-optimal design configurations. As a result, the CFD solver must possess the capability to handle designs that may appear unconventional or non-intuitive [30].
- ASO requires a large number of shape design variables. Therefore the CFD-based approach is typically too expensive. Lyu et al. [28] demonstrated that hundreds of design variables are necessary for both 2-dimensional and 3-dimensional optimisation. Only the gradient-based optimisation can handle this size of variables with the support of the CFD computation [25, 31]. Hence, employing a CFD-based approach becomes impractical for applications requiring numerous iterations or repeated calls due to the challenges and limitations described.
- Furthermore, CFD-based optimisation has predominantly been deterministic in nature, neglecting the presence of aleatory uncertainties stemming from variations in operating conditions, wear and tear-induced geometry changes, and manufacturing inaccuracies [32]. Overlooking these uncertainties in practice may result in unexpected performance degradation.

In recent years, DL has promptly responded to these challenges, demonstrating the potential to parameterise geometry, predict the aerodynamic metrics, and perform optimisation in a reasonable time frame [20, 21, 33–37].

1.3.1 Aerodynamic Coefficient Surrogate Modelling

ASO based on CFD presents a challenge due to its reliance on high-performance computer clusters. This is primarily because the optimisation process involves repetitive and resource-intensive evaluations of the simulation tools. The need for these iterative and costly simulations at every optimisation stage represents a huge challenge, since the evaluation of the aerodynamic objectives and constraints functions takes most of the computational costs [10]. Despite the availability of advanced computing capabilities, the current ASO approach limits the number of simulations at a few flight conditions to respect the computational budget. Therefore, the obtained design performs optimally at cruise conditions, while the flight envelope is sub-optimally covered [38].

Aerodynamic problem-solving typically involves scenarios that require real-time responses or numerous queries [39, 40]. To alleviate these challenges and facilitate a broader exploration of the design

[1] https://github.com/mdolab/MACH-Aero
[2] https://github.com/su2code/SU2

space, researchers have introduced cost-effective surrogate models. The optimisation through the use of surrogate models is called Surrogate-Based Optimisation (SBO) [41].

SMs provide cost-effective approximations with the intent of simulating the deterministic and computationally expensive behaviours of an original system, across either the entire design space or specific regions of it. SMs guide the optimisation process by emulating the responses of the system based on its inputs while minimising prediction errors. To achieve the approximation of the hidden function within the input and the output, a dataset comprising labelled samples is collected and used for training the model. This end-to-end approach allows the SM to capture the underlying relations between the inputs (typically representing geometric parameters and flight conditions) and the output, such as aerodynamic coefficients. Hence, the trained SM can quickly make predictions of unsampled input configurations. The advantage of this approach lies in its efficiency, as the surrogate can be faster evaluated when compared to the time-consuming CFD analyses. Consequently, a larger number of potential designs is evaluated within a given timeframe, increasing the chances of discovering the optimal design. Despite the initial cost associated with training the surrogate model, this strategy can prove advantageous in the long run if the surrogate model facilitates numerous optimisations [38, 41].

SBO is constituted of two stages: the first consists of the selection and assessment of initial samples, defined by the Design of Experiments (DoE) procedure, which is used to accomplish the training task. The DoE is defined as the sampling strategy in the design variable space, in order to maximise the amount of information with a limited number of samples [17]. In the subsequent stage, the surrogate model is improved through an iterative process that involves the infilling of new training samples [10].

The choice of the surrogate is not trivial but depends on the nature of the domain and the characteristics of the problem. Within the parametric approach, Kriging (also known as Gaussian process regression) has gained popularity because of its robust fitting capability and its capacity to offer prediction confidence intervals [42]. A possible application of the algorithm in the literature is found in the work of Liu et al. [43], who have presented a Kringing surrogate model combined with parallel infill-sampling method and multi-round strategy for the optimisation of a morphing wing considering a wide Mach-number range.

To leverage the advantages of multi-fidelity simulations, several Kriging variations such as Cokriging [44, 45], Hierarchical Kriging [46], and Polynomial Chaos (PC) [47] were introduced. Researchers have been interested in these methods since the models make full use of low-fidelity information [48]. For example, in 2013 Huang et al. [49] implemented a multi-fidelity surrogate model built with the Co-Kriging method to optimise a wing by reducing the induced drag. By comparison with the simple-Kriging method, the authors found that Co-Kriging strikes a higher accuracy with the same high-fidelity samples. Moreover, it achieves convergence more rapidly than the ”conventional” methods, all while utilising a smaller portion of high-fidelity data in the training dataset. However, traditional surrogate models are ill-suited for handling extensive training data sets. While expensive and limited experiments typically provide the training data for aerodynamic design, there is an increasing demand to manage large volumes of training data when constructing an aerodynamic surrogate model for interactive design [50, 51].

The development of surrogate models for interactive design has been addressed using DL-based

shape parameterisations and modelling techniques. Surrogate models, such as a mixture of expertise (ME) and Artificial Neural Networks (ANNs), make it possible to predict aerodynamic functions in larger spaces by using large volumes of training data. It is a matter of fact that the accuracy of extrapolations increases with the dimensionality of the design space [10, 52]. By reducing the design parameters and augmenting the training dataset, the surrogate model is enhanced to obtain more exact predictions of the aerodynamic coefficient while minimising the error. In the data-based approach context, the dataset is sourced from simulations or experiments. It is important to note that the precision of the prediction mainly relies on both the quantity and quality of the training dataset. Hence, it is of utmost importance to ensure that the samples are accurately representative of the entire design space.

With the continued development of surrogate modelling techniques, state-of-the-art models have become increasingly effective and efficient [10, 53]. For example, Raul et al. [42] proposed a Kriging regression model to approximate the drag and lift coefficients and delay the aerofoil dynamic stall. Bouhlel et al. [20] has developed a data-based approach for aerofoil shape design. The proposed ML tool consists of a gradient-enhanced artificial neural network, where the gradient information is phased in gradually. The approach delivers aerofoil shape optimisation solutions that are on par with those achieved through high-fidelity CFD optimisation solutions, outperforming the existing approach models that implement a mixture of experts.

Regarding the evolution of DL in ASO, machine learning techniques play a pivotal role in understanding, predicting and optimising the aerodynamic coefficient for several applications. Researchers have explored a wide spectrum of approaches to enhance accurate performance within a constrained time frame. Due to the increasing representation capability coupled with the time-saving evaluations, Multilayer perceptron (MLP), Convolutional Neural Network (CNN) and Reinforcement Learning (RL) gained the increasing interest of researchers. MLP models are currently used in the evaluation of aerodynamic coefficients due to their adaptability. Moreover, they are easy to train and implement ensuring a high accuracy level [54]. In the literature, several examples are found, such as the work of Jahangirian et al. [33] that presented an efficient evolutionary algorithm for shape optimisation of transonic aerofoils. In this article, aerodynamic coefficients are forecasted using an MLP, leading to a subsequent optimisation technique that yields a remarkable 60% increase in computation speed. Zang et al. [34] have proposed a multi-fidelity MLP with the purpose of constructing a high-fidelity surrogate model by blending different fidelity information. The optimisation results demonstrate that the proposed tool has remarkably outperformed the single-fidelity method. Du et al. [55] used a B-spline-based generative adversarial network mode for shape parameterisation in combination with MLP, recurrent neural network (RNN) and ME to enable the prediction of scalar quantities, such as drag and lift coefficients, and vector quantities such as pressure distribution. The purpose of the study was to develop a tool able to optimise in a few seconds the aerodynamic shape of an aerofoil in a subsonic or transonic regime.

Powerful CNNs show potential in modelling aerodynamic coefficients with respect to the geometry coordinates without using any shape parameterisation [10]. Many applications are indeed found in the literature like the work of Bhatnagar et al. [56] which has implemented a CNN to predict the velocity and pressure field in unseen flow conditions and geometries given the pixelated shape of the aerofoil.

Chen et al. [35] have performed a graphical prediction method based on CNN for multiple aerodynamic coefficients of both symmetric and asymmetric aerofoils. Ultimately, Sekar et al. [36] proposed an approach to perform the inverse design of aerofoil using a CNN that has been trained with the pressure coefficient distribution to obtain a prediction model for the aerofoil shape.

RL refers to any ML-based algorithm whose aim is making the right sequence of decisions based on interactions with the environment, under uncertainties and without a prior model which guides the program [57]. Within the scope of ASO, we find a wide application of the RL coupled with the DL. The development of RL in the SBO addressed the issue of handling high-dimensionality problems, as the training is not performed via supervised learning but the training dataset is constantly modified by the environment returned rewards. For example, in 2019 Yan et al. [21] applied RL and transfer learning (TL) to optimise a missile geometry. Instead, Li et al. [37] applied deep-reinforcement learning (DRL) to optimise super-critical aerofoil in terms of aerodynamic drag. The DRL algorithms are implemented to simulate the design process by trial and error, and a pre-training phase is previously implemented to increase convergence.

It is noteworthy that besides the great savings in computation time, machine learning tools in SBO have proven to be performative and competitive while preserving the same accuracy as high-fidelity modules. To provide the reader with an overall sight, a comparison of the prediction errors obtained by the state-of-art DL models used for the prediction of aerodynamic coefficients is presented in Table 1.1.

Table 1.1: Prediction errors obtained by the state-of-art DL surrogate models used for the prediction of aerodynamic coefficients.

Study	Model	Application	Fligh Regime	Training Samples	Modelling Variables	Coefficients	$\epsilon_{\mathcal{L}^2}$ [%]	RMSE [‰]
Li et al. [51]	GE-KPLS	Aerofoil	Subsonic	81,000	16	C_d, C_l	0.26, 0.15	-
			Transonic	32,400	10		0.83, 0.40	-
Li et al. [50]	MLP	Wing	Transonic	135,108	57	C_d, C_l , C_m	0.35, 0.20, 0.36	-
Zhang et al. [58]	CNN	Aerofoil	Transonic	1,600	2403	C_l	-	70.71
Du et al. [55]	MLP	Aerofoil	Subsonic	45,696	29	C_d, C_l	2.34, 2.26	12.90, 2.77
			Transonic	39,505	29		2.87, 4.65	16.13, 8.76
Moin et al. [59]	MLP	Aerofoil	Subsonic	454,675	23	C_d, C_l, C_m	2.17, 1.74, 1.64	7.88, 14.20, 4.21

The metrics assessed are the ones provided by the authors, including:

- the Root Mean Square Error (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\widetilde{y}_i - \mathrm{y}_i)^2}, \tag{1.4}$$

where N is the number of samples used for computing the error, $\widetilde{\mathbf{y}}$ is the vector containing the predictions and $\mathbf{y}$ is the vector that stores the corresponding real values.

- the relative $\mathcal{L}^2$ error:

$$\epsilon_{\mathcal{L}^2} = \frac{\|\widetilde{\mathbf{y}} - \mathbf{y}\|_2}{\|\widetilde{\mathbf{y}}\|_2}. \tag{1.5}$$

1.3.2 Aerofoil Geometry Parameterisation

Parametric modelling of aircraft is a crucial phase in the aerodynamic design process. The parametric modelling process is expected to enhance the numerical control of the configuration. It consists of a mapping of the geometric space in a parametric space in order to represent a large geometric space by a small number of variables [60]. Although a parameterisation that provides a large degree of shape freedom might seem ideal, it can become inefficient in the design space which needs to be completely represented and constrained during the optimisation process. Conventional parametrisation methods rely on the user's sensibility for choosing the number of design variables. Shape optimisation is theoretically an infinite-dimensional problem, however the design approach always requires the discretisation of the shape with a finite set of parameters. The absence of theoretical indicators, that guide the designer in the design variable definition for each specific optimisation problem, is a considerable issue for the user who can only rely on his own experience. Increasing the number of design variables improves the geometry representation at the cost of increasing the optimisation time. Moreover, it may cause a struggling subject for the CFD-solver that needs to handle high-frequency shape variations. The increasing number of design variables influences not only the speed but also the robustness of the optimisation process. The high dimensionality of the design space guides the optimiser to explore shapes with more curvature variation, and those might cause difficulties for the flow solver [25]. A possible solution to address the CFD robustness issue may be tuning the flow solver and the optimiser parameters, but it could be time-consuming and inefficient for some applications. A better approach might be starting with a few design variables to get a rough approximation of the optimal shape and use it as a starting point for further design refinement [25].

In SBO, a similar problem can arise due to the limitation of the surrogate model used in the optimisation process. Specifically, the adopted surrogate may not differentiate the wavy aerofoil shapes as abnormal or sub-optimal, thus it can lead to convergence towards undesired solutions. Generally, the dataset used to train the surrogate model includes mainly conventional aerofoil shapes, usually taken from the University of Illinois Urbana-Champaign (UIUC) Aerofoil Coordinates Database [61]. The open-source database comprises approximately 1,600 aerofoils with diverse applications, mainly used for subsonic and transonic regimes and a wide suitable range of Reynolds numbers. The UIUC database has proven to be a valuable and indispensable resource for ASO [51, 55, 58, 62–64]. Handling the limit of the surrogate model has been an issue shared by the researchers. The models are often not capable of recognising wavy or high-dimensional geometries as low-performance aerofoils, since they may not be adequately represented in the training data. Therefore, choosing the most convenient aerofoil shape parameterisation plays a key role in obtaining the best possible performance in ASO.

Many parameterisation techniques are used to parameterise the aerodynamic shape. Among com-

mon methods, PARSEC [65], class/shape function transformation (CST) [66], free-form deformation (FFD) [67, 68], B-spline [69], Bézier curves [70], Hicks-Henne bump functions [71], Radial Basis Function (RBF) [17] and National Advisory Committee for Aeronautics (NACA) aerofoil definition [72] are included.

These aerofoil parameterisation method can be classified as [73]:

- Constructive methods, such as polynomials, splines and CST, where the aerofoil surface is purely based on the values of the parameters specified;
- Deformative methods, such as discrete, analytical and FFD methods, where the base-line aerofoil is deformed to create a new shape.

At first glance, a discrete approach where the DVs are directly defined on the surface might seem the most effective and simple. A discrete method provides fine local control and the simplicity of using the computational grid without any parameterisation. However, it is affected by high costs for a dense surface grid. Therefore, polynomials and spline approaches are implemented in order to reduce the number of DVs [74]. Despite the implementation differences, B-spline and Bézier curves are prevalent due to the efficient parameterisation of complex geometries. On one hand, B-spline curves are defined by a set of control points and basis functions, which leads to remarkable flexibility in deforming shapes while ensuring smoothness. On the other hand, Bézier-curves are defined and constrained by a fixed number of control points and thus are generally used for simple curve manipulations. The choice between those methods depends on the specific requirements of the application but, generally, B-splines are opted for their versatility while Bézier curves are chosen for their simplicity and precision [75, 76].

Another possible constructive method is the CST parameterisation. CST approach represents aerofoils and wing-shaped surfaces as analytically well-behaved and smooth shape functions. This parameterisation provides an intuitive way to control the shape and ensures local control of the curves, therefore it is suitable for fine-tuning. Nonetheless, CST parameterisation exhibits limitations in terms of flexibility and global control. Its capacity to represent complex or irregular shapes is constrained, with the inability to globally adjust a curve. [77]. PARSEC method employs the approximation of the surface by a 6^{th} order polynomial. However, the method employs only 12 design variables with a consequent reduction of the flexibility [73]. The aforementioned limitations are addressed by the coupling of the two strategies, exploiting the flexibility of the CST method with the intuitiveness of the PARSEC method. [78].

Lately, FFD is a parameterisation technique that embeds a flexible object of interest inside a flexible volume. The DVs are the nodes on the control volume and are respectively moved to deform the embedded object. The FFD approach is typically versatile for deforming and morphing shapes. It is characterised by intuitive regulation and offers both local and global control, enhancing more freedom in the design. A possible disadvantage of this technique is the complexity of the FFD that can result in computationally intensive and expensive computations [73, 79]. Even though the conventional approaches have been outperformed by the data-driven aerofoil parameterisation [10], they are still widely used in the ASO problem since they allow a controlled local deformation in the shapes. This represents a great advantage for designers who demand more control during the optimisation process [50, 80–83].

Figure 1.3 illustrates multiple unconventional aerofoil shapes gained through the manipulation of FFD control points. It is evident the precise control exerted over each vertex within the FFD framework.

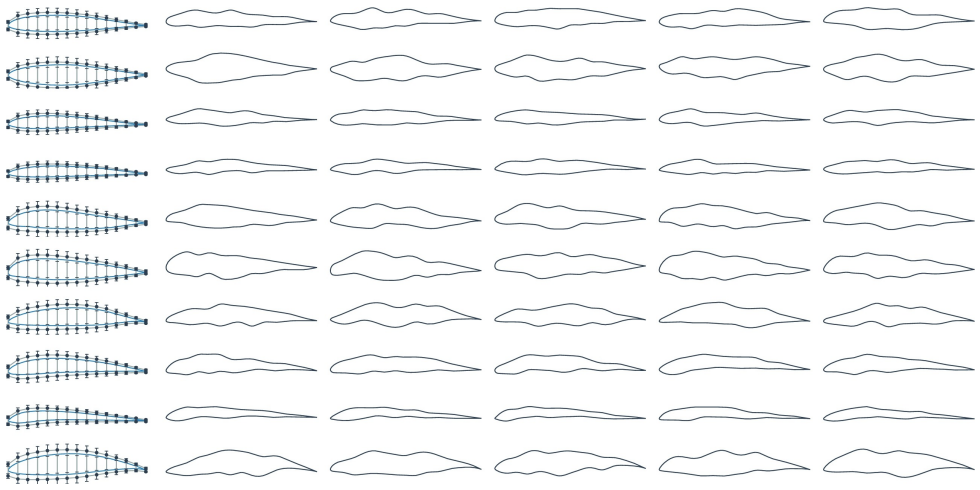


Figure 1.3: Abnormal aerofoils generated by perturbing FFD control points [84].

Advanced ML models have shown the potential to efficiently parameterise the aerodynamic shape, due to their ability to exclude abnormal shapes and reduce the dimensionality of the design space. Data-driven approaches are mainly classified into modal parameterisation methods and geometric filtering techniques. Modal parameterisation involves the consolidation of design variables while imposing geometric constraints to narrow down the design space. Principal Component Analysis (PCA) [85] has been widely used for linear modal parameterisation [73, 86]. PCA is generally applied to calculate full aerofoil modes and camber thickness modes, using Singular Value Decomposition (SVD) [50, 87]. Nowadays nonlinear modal parameterisations are substituted by advanced deep learning models such as Generative Adversarial Networks (GANs), which emphasise the generation of realistic aerodynamic shapes [55, 62, 63]. GANs excel at capturing intricate design variations, producing high-fidelity shapes that closely resemble real-world aerodynamic profiles, as highlighted in Figure 1.4. The capacity for nonlinear modelling makes GANs a superior choice when aiming to represent the diversity and complexity of aerodynamic shapes, offering a significant advantage in advancing aerodynamic research and design optimisation [88]. In conclusion, we can state that the integration of advanced DL models into the parameterisation of aerodynamic shapes represents a remarkable change in the field. These models bring efficiency by eliminating non-physical designs and reducing dimensionality. Furthermore, their capacity for nonlinear modelling, particularly with techniques like GANs, enables the creation of highly realistic and varied shapes.

1.3.3 Optimisation Schemes

In the process of formulating an optimisation problem, the selection of the optimisation algorithm is a critical decision. The final choice is the result of a tuning balance of two critical factors: the extent of optimisation achieved and the rate at which convergence is attained. Gradient-based optimisation techniques conduct a targeted exploration, but they run the risk of getting trapped in local minima.

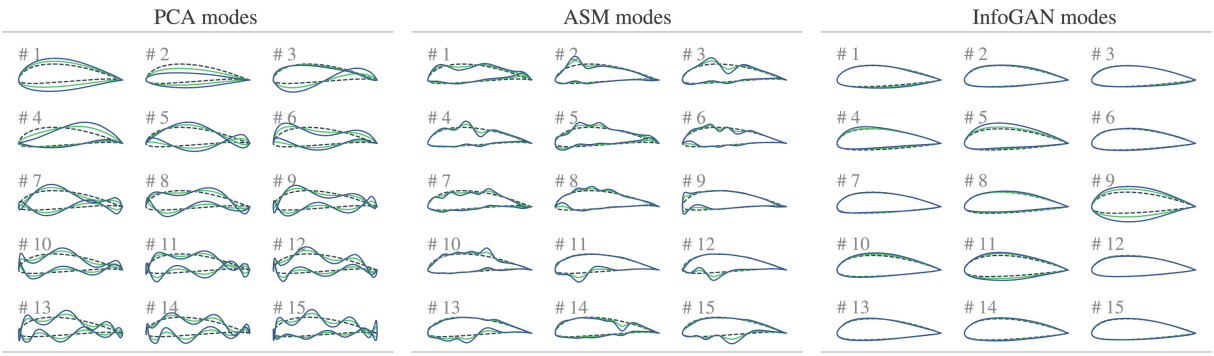


Figure 1.4: Aerofoil shape deformation by different types of modes [10].

Conversely, gradient-free methods undertake a broader search but they cannot guarantee convergence. Striking a balance between efficiency and effectiveness becomes imperative when determining the most suitable optimisation algorithm [38, 89, 90].

Gradient-based optimisation algorithms use gradients of both objective and constraint functions with respect to design variables [38]. Due to the reliable information given by the gradients, these algorithms are characterised by high efficiency when dealing with smooth and well-behaved objective functions. Algorithms such as Sequential Least Squares Programming (SLSQP) [91], Stochastic Gradient Descent (SGD) [92], Adaptive Moment Estimation (Adam) [93], use the gradients to estimate the direction of the steepest ascent or descent. As each method conducts a local search around the current solution, it becomes imperative to establish a clear research direction for the subsequent iteration, guiding the process towards the minimum. However, the differentiability of the objective functions has to be ensured to apply these reliable methods. Nevertheless, a significant drawback of the gradient-based optimisation algorithms is stucking in local minima. Although they are sensible to the choice of the initial solution, these algorithms are preferable for high-dimensional applications for their remarkable exploit capability [38, 89, 90].

Gradient-free optimisation algorithms, such as Genetic Algorithm (GA) [8], are essential tools for tackling optimisation problems when gradient information is unavailable or impractical to compute. They do not rely on gradient information, therefore a global search across the solution space is conducted. These algorithms are quite recommended for exploring complex, non-smooth, and multi-modal objective functions, making them particularly well-suited for optimising black-box systems, noisy real-world processes or simulations. Gradient-free algorithms also have the advantage of being able to escape from local optima, but this comes with increasing evaluations and reduced efficiency when compared with gradient-based algorithms. Typically the termination criteria in gradient-free optimisation are often not straightforward, therefore a computational budget is set up in advance. However, in the SBO context, the analysis of the objective function is inexpensive, therefore the rising computational costs became less significant [38, 89, 90].

1.3.4 Morphing Aerofoil Shape Optimisation

The word *"morphos"*, according to the Greek etymology, means gradually changing from one thing to another [94]. In the engineering context, morphos indicate the ability to transform shape or structure. Morphing wings are designed with the purpose of increasing aircraft performance, by modifying different design parameters in order to reach an optimal configuration [11, 13, 95].

Morphing aerofoils which involve the adaptive alteration of a wing section, have found application across various aviation domains including unmanned aerial vehicles (UAVs) [96], commercial aircraft [97] and military platforms [11]. It has been demonstrated that the morphing aerofoil can substantially enhance the aerodynamic performance of a wing, yielding an alteration of only a fraction of the lift distribution. Modifying the camber and/or adjusting the thickness represent two of the most frequently employed alterations of an aerofoil. Benefits, such as the controlled distribution of up-force during the most critical phases of the mission, increasing manoeuvrability and better stability are reached with the variation of these two parameters. Moreover, the location of the laminar-turbulent flow transition could be properly controlled with the variable thickness, and the laminar region of the boundary layer can also be extended by delaying the transition location towards the trailing edge. Indeed, the investigation and analysis of leading edge and trailing edge morphing have received considerable attention in research, since flap and slat surfaces are widely adopted for achieving high-lift configurations. It is well known that the trailing edge flaps are responsible for the airframe noise during the approach, due to the leading edge gap needed to energise the flow [98]. Hence, there is significant interest in achieving comparable performance without the need for pronounced edges, while harnessing the potential of a morphing skin and actuators [11, 13, 95].

Similarly, morphing wings are a subject of extensive research within the aerospace community, involving for example the entire wing shape by the alteration of the span or of the sweep angle. The literature highlights the coexisting challenges, including integration complexities, cost considerations, and the trade-offs between morphing capabilities and structural complexity. Nowadays, the 2-dimensional morphing approach is therefore preferred since it has been demonstrated to be the best compromise between reliability and cost of manufacturing and assembling [11, 13, 95].

In spite of the ongoing research into gradient-based optimisation of morphing profiles, there are relatively few works that develop a gradient-based framework. For example, Zhang et al. [99] present a gradient-based aerodynamic optimisation framework for a transonic aerofoil equipped with morphing leading and trailing edges with the aim of preserving the attachment of the shape at the wing-box region and providing the skin length control during the optimisation process. A second application is found in the work of Lyu et al. [100] which explores the potential of adaptive morphing trailing-edge wings in reducing fuel burn for transport aircraft. The limited number of articles on gradient-based morphing optimisation is attributed to the fact that optimising a morphing profile requires multiple evaluations of objective functions, which is why in recent years, an SBO approach has become more prevalent. Precisely, later research are focused on multi-fidelity [101, 102], multi-disciplinary [103] and multi-objective [102, 104–107] optimisation architectures.

Typically the aerodynamic optimisation of the morphing wing is performed separately from the morphing mechanism [108]. Even though data-based approaches for aerodynamic evaluation of aerofoils under a wide range of Mach number, Reynolds number and angle of attack (AoA) are commonplace at the state-of-art, the practical implementation of pure morphing aerofoils remains notably challenging. One of the related publications is the study developed by Junior et al. [83] that comprises an online data-based framework for the optimisation of aircraft aerofoil. The online solution is based on a data-driven controller combined with a surrogate model used to generate trajectories of the non-linear system representing the morphing aerofoil. As highlighted in Section 1.3.1, several DL models have been employed to achieve a data-driven computation of the aerodynamic performance of an aerofoil. Nevertheless, in order to ensure accuracy each of the proposed surrogates is usually constrained in the design space and/or the flight range of interests [40, 59, 108]. This approach is favourable to guiding the optimiser towards a solution that strikes a good compromise between accuracy and efficiency. However, it is noteworthy that DL models are affected by a consequent reduction of applicability, leading to reduced feasibility as a tool for representing a complex flight envelope. Moreover, to the best of the author's knowledge, there is no publication in the literature that constrains an embedded wing box during the surrogate-based optimisation process. The conservation of a wing box is commonly employed during the design of an aircraft, in order to ensure structural integrity, aeroelastic responsiveness and fuel storage [94, 95]. Based on these premises, this thesis work establishes its objectives.

1.4 Objectives and Contribution

The main objective of this thesis is to explore the potential of the DL models in the ASO of a morphing aerofoil. The first goal is to develop a robust and modular optimisation framework for addressing shape optimisation problems with minimal inference time. To attain this objective, a framework leveraging the state-of-the-art DL models is proposed and integrated in both multi-objective and multi-point optimisation approaches. The second goal consists of studying the applicability of existing data-driven tools for different shapes and flight conditions in order to enhance generalisation and representation in SBO. The final intent is to introduce a comprehensive data-driven architecture that addresses the lack of deep learning approaches for a morphing aerofoil, considering a wide range of flight conditions, from compressible subsonic to transonic. The framework is also designed with the purpose of enhancing flexibility and adaptability to new optimisation missions and scenarios. Taking into account the previously mentioned objectives, this thesis aims to make three principal contributions:

- The development of a data-driven framework for ASO of two-dimensional aerofoils in morphing configuration.
- The integration of the free-form deformation technique in order to reduce the dimensionality of the design space and ensure the conservation of a wing box within the profile.
- The benchmark for comparing the developed surrogate model and optimisation framework with their respective state-of-the-art counterparts.

1.5 Thesis Outline

The document is organised as follows:

- Chapter 1: encompasses an exhaustive review of the current state of ASO state-of-the-art, with specific emphasis on aerodynamic surrogate modelling and aerofoil geometry parameterisation;
- Chapter 2: An extensive theoretical overview of the core topics is provided for a better understanding of the proposed framework. The chapter details the theory behind DL-based metamodels, including an extensive discussion about regression neural networks. Moreover, the aerodynamics of subsonic and transonic regimes are described in the physics of the flow in order to better discuss the performance of the SM proposed ;
- Chapter 3: A detailed description of the methodology and its foundations are demonstrated, detailing the assumptions made to build the optimisation framework and addressing the selection of both the multi-objective and multi-point approach;
- Chapter 4: The results obtained are examined and compared with state-of-the-art techniques. The SM tailored for the prediction of the aerodynamic coefficients is examined in terms of performance and the results of morphing aerofoil shape optimisation are presented;
- Chapter 5: A final overview of the thesis achievement is presented highlighting the potentiality of the framework and addressing its limitations. Therefore, final remarks are presented and future developments are comprehensively proposed.

Chapter 2

Theoretical Background

This chapter aims to provide a comprehensive background, enabling a better comprehension of the thesis work which aims to apply a DL model for the optimisation of a two-dimensional aerofoil in morphing configuration. Among the extensive applications of DL, we find a remarkable implementation in ASO where both parametric tasks and aerodynamic computations can be performed by a data-driven model. However, considering the application of this thesis work, only the theoretical background of the regression neural network is detailed. In Section 2.1 an extensive deep insight into the MLP architecture is provided. In Section 2.2, a basic introduction to the fluidodynamic subject is presented focusing on the physics of incompressible and compressible flows.

2.1 Deep Learning

Machine learning is a branch of artificial intelligence and computer science, whose main characteristic is learning by data. T. Mitchell in his book [109] has stated: *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"*.

Every ML algorithm is designed to explore and exploit the input provided to it. Through this exploration, the algorithms establish policies which are in turn employed to train the resulting model. This iterative process of learning and adaptation is the core of machine learning, enabling models to make more accurate predictions as they gain more experience with the data. ML algorithms encompass a wide range of techniques, making it common practice in the literature to classify them into broad categories based on the type of learning experience they undergo, as depicted in Figure 2.1. This classification is instrumental in understanding the difference in the underlying logic of the learning processes employed by ML algorithms [10].

According to the literature, supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning are the most adopted approaches for the training of a ML model [10]. Among these, supervised learning is often regarded as the most straightforward approach due to its reliance on labelled data. Supervised learning applications are broadly divided into classification and regression

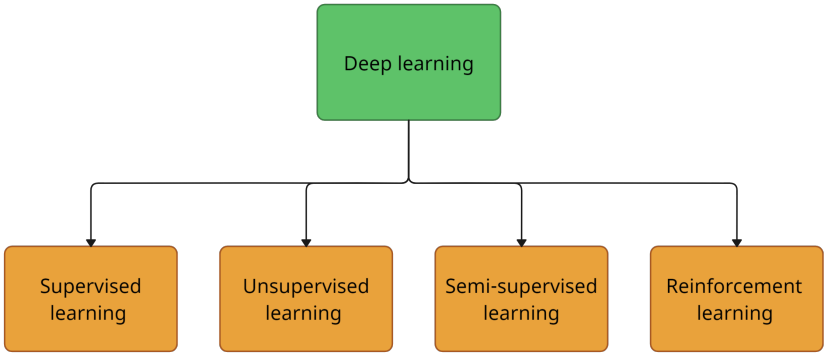


Figure 2.1: Classification of learning experiences.

[110]. In both cases, the model is supplied with inputs, x, and the corresponding output, y, to learn the connection and make accurate predictions for unsampled data. K-nearest neighbours (KNN) [111] and support vector machine (SVM) [112] are two implementations of supervised learning architectures.

Unsupervised learning uses a different approach since it does not consider labelled input. Differently, this approach extracts information from unlabelled data, thereby operating independently of any supervision signal [52]. One prevalent technique in unsupervised learning is Polynomial Chaos Expansion (PCE), which serves as an effective means to linearly parameterise the dimensions of design variables [85].

Semi-supervised learning represents a tradeoff between the two aforementioned approaches. The algorithm, in this case, includes both labelled and unlabelled data to enhance model performance. In applications where acquiring data is costly and time-consuming, semi-supervised learning represents a valid solution [10]. By leveraging a small set of labelled samples infilled with a larger cluster of unlabelled data, this iterative approach can extrapolate patterns more effectively and with a faster convergence rate [113].

In contrast to approaches that use pre-existing database to represent the subject matter, reinforcement learning centres around the concept of agents that learn the decision policies through direct interaction with the environment [57]. In this dynamic process, agents use their experiences to update their future steps. Throughout these interactions, rewards and penalties play the role of pivotal feedback that guides the learning process. RL is a valuable solution for scenarios where collecting a comprehensive and representative dataset is unfeasible or impractical. Moreover, RL excels in applications that require online training, enabling agents to make sequential decisions and be adapted to changing circumstances [37].

DL, a subfield of ML, represents a significant advancement in AI and computer science. DL is characterised by its use of neural networks with many layers to tackle complex tasks, making it especially well-suited for generation models, image recognition and object detention [114–117]. In the last decades, DL has become an essential tool in achieving a wide range of engineering objectives [2, 6, 9]. This increasing trend can be attributed to the data-centric nature of these methods, which successfully meet the several requirements associated with the design and optimisation phases of engineering projects.

2.1.1 Regression Neural Networks

Deep feed-forward network, also called Multilayer Perceptron (MLP), is the characterising model to be defined.

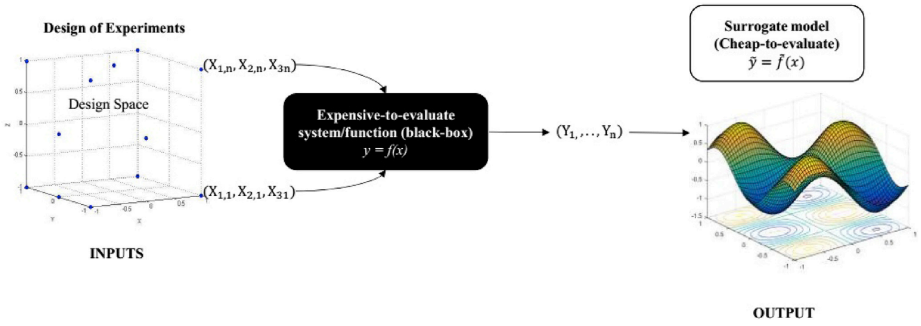


Figure 2.2: Basic surrogate model framework [41].

As depicted in Figure 2.2, a surrogate model trained through supervised learning aims to approximate an expensive-to-evaluate function, $f(\mathrm{x})$, by using a limited set of labelled samples. The result of this trained surrogate is a more cost-effective function, $\widetilde{\mathrm{y}} = \widetilde{f}(\mathrm{x})$, which maps an input x to a category $\widetilde{\mathrm{y}}$ [41].

Similarly, the primary objective of MLP is to define a mapping function that can provide predictions, $\widetilde{\mathrm{y}}$, for unsampled data. SMs are often described as *'black boxes'* because they allow users to define inputs but do not provide control over the specific methods used to perform mapping tasks. Likewise, a feed-forward network defines $\widetilde{\mathrm{y}} = \widetilde{f}(\mathrm{x}, \boldsymbol{\theta})$ as a function of hyperparameters, $\boldsymbol{\theta}$, which are tuned during the training iterative process in order to obtain the best possible approximation of the original function, $f(\mathrm{x})$ [52].

MLP is regarded as the initial architecture upon which subsequent models have been developed. The unit elements of the MLP are the perceptrons or neurons, which are clustered in multiple levels or layers. The input layer is associated with the input parameters, $\mathbf{x}$, while the output layer is associated with quantities of interest, $\widetilde{\mathbf{y}}$. Each neuron is characterised by its activation number which is properly extracted through the training process [10].

The term feed-forward describes the sequential flow of information through the neural network architecture, It involves the proper processing of input data, $\mathbf{x}$, as it passes through each intermediate layer, ultimately culminating in the computation of the best output prediction, $\widetilde{\mathbf{y}}$. Feedback connections in which outputs of the model are fed back into itself are not included [52]. MLPs are a common example of feed-forward networks, characterised by their fully connected structure, meaning that every perceptron in one layer is connected to every perceptron in the adjacent layers. This extensive connectivity enables MLPs to capture complex relationships and patterns within the data, without requiring any assumption on the probability distribution of the data. This feature makes them suitable for a wide range of tasks [118].

The MLP is associated with a direct acyclic graph describing how the functions are composed together. For example, the approximated function can be defined as $\widetilde{f}(\mathrm{x}) = \widetilde{f}^{(3)}(\widetilde{f}^{(2)}(\widetilde{f}^{(1)}(\mathrm{x}))$. These

chain structures are commonly used since each layer contributes sequentially to the approximation. Precisely, each layer takes the activation number from the previous layer as input and produces its own set of activations, which are passed to the subsequent layer. This cascade of information allows the network to progressively extract features from the input data, as it moves deeper into the underlying relations within the input samples. As the complexity network increases, its capacity to learn becomes more refined, with a corresponding rise in the training cost [52]. The iterative training process is directed by the learning algorithm, responsible for adjusting the activation values of each layer until the desired level of accuracy is achieved. When the output of each layer is not shown, hidden layers are included in the network, which is referred to as Deep Neural Network (DNN) [119]. Drawing inspiration from neuroscience, these architectures are designed such that each element operates simultaneously with the others, forming a deep and interconnected network similar to the human brain. The advantage of applying this logic to AI layers lies in the rapid computation of activations throughout the majority of the architecture. Moreover, this efficiency proves especially beneficial for forward propagation [52].

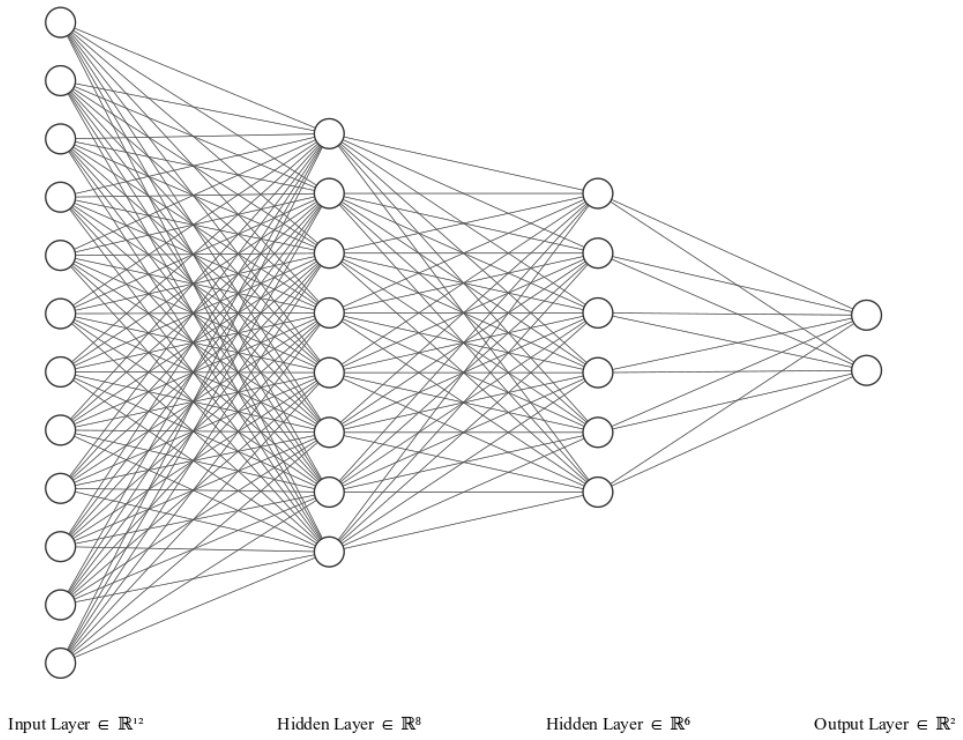


Figure 2.3: Schematics of a generic MLP.

The functioning of a MLP is focused on the computation of weighted sums of activations across its interconnected neurons, as illustrated in Figure 2.3.

The implementation of the feedforward networks begins with linear models, such as logistic and linear regression. These models hold a special place of interest because they can be efficiently fitted either in closed-form solutions or through convex optimisation techniques. The attractiveness of these linear models lies in their simplicity and transparency. However, while linear models are indeed flexible and reliable for certain tasks, they are intrinsically limited by their linearity. This limitation restricts their

suitability when faced with more complex and intricate problems. Therefore, activation functions have been introduced in the learning procedure, in order to address the limitation and enhance non-linearities [120]. Furthermore, MLPs built with hidden layers handle better the increasing complexity of the design space [88, 118, 119].
A generic regression neural network can be mathematically formulated as a series of two operations. Firstly, a linear combination of inputs is performed,

$$u(\mathbf{x}) = \boldsymbol{w} \cdot \mathbf{x} + b, \tag{2.1}$$

where $\boldsymbol{w}$ are the connection weights and b is the bias term. Secondly, the activation function is applied to represent the nonlinearities within the layers,

$$v(\mathbf{x}) = \Theta(u(\mathbf{x})) = \Theta(\boldsymbol{w} \cdot \mathbf{x} + b), \tag{2.2}$$

where $\Theta : \mathbb{R} \to \mathbb{R}$ is the activation function.

Considering a MLP composed by N hidden layers of neurons, an output $\widetilde{\mathbf{y}} \in \mathbb{R}^m$ is predicted from a set of inputs $\mathbf{x} \in \mathbb{R}^n$, where (n, m) are the dimension of both vectors. The i-th hidden layer sizes K_i neurons. The pre-activation of the i-th hidden layers can be expressed as,

$$\boldsymbol{u}^{(i)}(\mathbf{x}) = \mathbf{W}^{(i)}\boldsymbol{v}^{(i-1)} + \boldsymbol{b}^{(i)}, \tag{2.3}$$

where $\mathbf{W}^{(i)} \in \mathbb{R}^{K_i K_{i-1}}$ is a matrix of weights to be determined and $\boldsymbol{b}^{(i)} \in \mathbb{R}^{K_i}$ is an unknown bias. In this type of architecture, every neuron is associated with an unknown bias which aims to neglect meaningful perceptrons in the current iteration, increasing therefore the reliability of the network.

Afterwards, $\boldsymbol{u}^{(i)}(\mathbf{x})$ is transformed by an activation function, $\Theta : \mathbb{R}^{K_i} \to \mathbb{R}^{K_i}$, as follows:

$$\boldsymbol{v}^{(i)} = \Theta(\boldsymbol{u}^{(i)}(\mathbf{x})). \tag{2.4}$$

By doing so, the inner representation of non-linearities is propagated through the network. Lastly, the output of the model is composed by the activation, $o : \mathbb{R}^m \to \mathbb{R}^m$, of a linear combination of the last hidden layer:

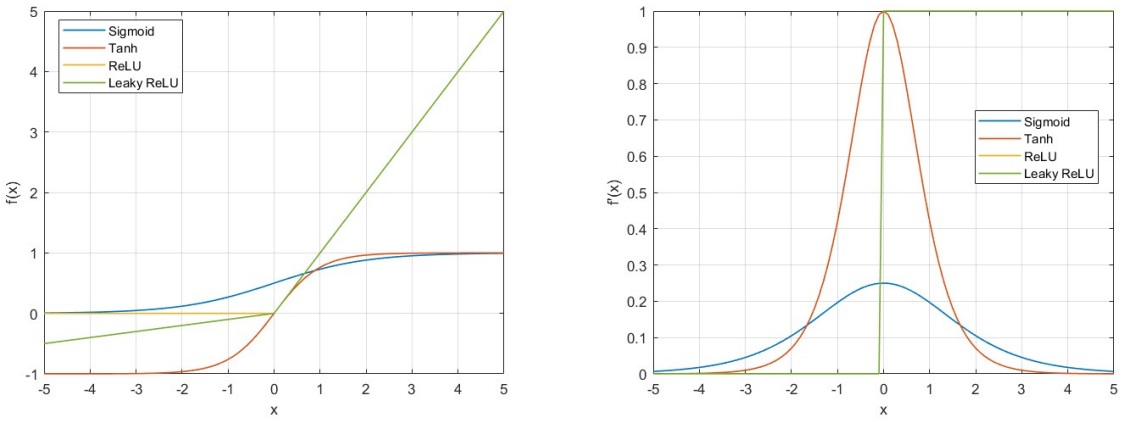
$$\widetilde{\mathbf{y}} = o(\Theta(\boldsymbol{u}^{(\mathrm{N}+1)}(\mathrm{x})) = o(\Theta(\mathbf{W}^{(\mathrm{N}+1)}\boldsymbol{v}^{(\mathrm{N})} + \boldsymbol{b}^{(\mathrm{N}+1)})) \tag{2.5}$$

Activation function

As mentioned before, $\Theta : \mathbb{R} \to \mathbb{R}$ is an activation function that injects nonlinearity into the network and allows the network to create a broad range of output values without any particular threshold or constraint. Typical activation functions include the sigmoid function, $\Theta(\mathrm{x}) = \frac{1}{1+e^{-\mathrm{x}}}$, the hyperbolic tangent function $\Theta(\mathrm{x}) = \frac{e^{\mathrm{x}}-e^{-\mathrm{x}}}{e^{\mathrm{x}}+e^{-\mathrm{x}}}$, the Rectified Linear Unit (ReLU) $\Theta(\mathrm{x}) = \max\{0, \mathrm{x}\}$ and the Leaky ReLU $\Theta(\mathrm{x}) = \max\{\delta\mathrm{x}, \mathrm{x}\}, \delta > 0$ [120].

The sigmoid function is characterised by a S-shaped curve that compresses its input into $[0, 1]$ interval. In the early days, it was widely used due to its smooth and differentiable nature [120]. Nowadays, it has been outperformed by other activation functions, such as ReLU and its variants, since the sigmoid is affected by the vanishing gradient problem [121]. This issue consists of the gradient of the activation function that approaches zero, as the input of the function increases its magnitude (either in positive or negative sign). Several functions are affected by this issue, like the hyperbolic tangent function that maps values to a range between -1 and 1. As a consequence, the vanishing gradient issue affects drastically the back-propagation learning process of DNN models, with a consequent difficulty in updating the weights in the first layers of the network. The worst scenario in these cases is the failure of the training process. Nevertheless, the sigmoid function is still used in output layers where the values strictly demand to be bound within a specific range [119–121].

The ReLU function, along with its derivatives, is the most used activation function in DNN due to its computational efficiency and the absence of the vanishing gradient problem. The linear function is 0 for negative inputs and the input value for positive inputs, whereas the gradient of the ReLU is either 0 or 1 regardless of the magnitude of the input. The learning process is aided by the bounded gradients. However, the ReLU function is affected by the dead neurons issue that arises when the output of a neuron remains negative. This phenomenon, coupled with a zero value of the gradient, prevents the neuron from learning and contributing. Nonetheless, it can be prevented by fine-tuning both the algorithm learning rate and the input parameters [122]. Several modifications of the ReLU function have been proposed, such as Leaky ReLU which adds a small positive slope to the negative input, addressing the dead neurons issue. Indeed, the modification applied improves the performance of DNNs since the positive slope avoids the inactivation of the neurons [119, 122]. The functions cited are plotted with their derivatives in Figure 2.5.



(a) Activation functions.

(b) Activation function gradients.

Figure 2.4: Typical activation functions (a) and their gradients (b): sigmoid, tanh, ReLU and leaky ReLU with $\delta = 0.1$. These plots were generated using MATLAB, drawing inspiration from the graphical representations found in [120].

2.1.2 Training routine

In Section 2.1.1 the essential theory of a regressive DL model is presented. Each neuron of the model is characterised by its expression in forms of multiple weights, $\boldsymbol{\theta}$. These parameters are not defined a priory, but rather, they are iteratively obtained during the back-propagation learning process. Among the broad range of learning algorithms, the backpropagation algorithm looks for the minimum of the error function in the weight space. This error is often referred to as the "risk" and it is computed using the method of gradient descent [123, 124]. The vector of weights, $\boldsymbol{\theta}$, is therefore computed to minimise the error function. The training process starts with providing a dataset of samples, x, each coupled with a label, y. Once the information has flown through the network, the predicted outputs, $\widetilde{\mathrm{y}}$, are compared with the labelled data in the so-called cost function, $J(\boldsymbol{\theta})$ [92].

The cost function is the objective function that the learning algorithm minimises by updating the set of weights and biases, iteration by iteration. It plays a pivotal role since it indicates the precision of the network in making predictions. Back-propagation strategy involves the computations of the gradient of the cost function, $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$, which is used in turn to set up each parameter of the network. The vector of parameters, $\boldsymbol{\theta}$, is iteratively modified in order to reach the minimum of the cost functions. It is noteworthy that the back-propagation learning process can be influenced by the initial starting point, ultimately determining whether the optimisation converges to a local or global minimum. Consequently, the quality of the starting point plays a significant role in this outcome.

The back-propagation algorithm represents an important step forward in DNN, proving efficiency. Typically, the optimisation algorithms compute a decomposition of the objective function over the training samples. By doing so, the empirical distribution of the function defined across the training set is computed iteration by iteration. Due to the hyperbolic size of the parameters to be defined for a neural network, this general approach is extremely expensive. Therefore, by updating and computing the gradient of the cost function, the back-propagation addressed this issue [52]. Nevertheless, the training process might still be slow with the increasing complexity of the architecture. Consequently, the issue might be managed by dividing the database into smaller clusters of samples and estimating the gradients for each portion. However, it should be mentioned that while fragmenting the evaluation into a small portion of the dataset is certainly less expensive than evaluating the model for the entire dataset, the minimisation of the objective function may be less accurate for the loss of a global awareness [52].

Gradient descent is one of the most popular algorithms for training DL models. These algorithms are mostly employed as *black-box* tolls, due to the lack of practical explanations regarding their advantages and limitations [124]. Three variants of gradient descent are mainly used in regression networks, classified by how the data are handled to compute the gradient of the cost function:

- Batch gradient descent computes the gradient with respect to the parameter $\boldsymbol{\theta}$ for the entire training dataset;
- Stochastic Gradient Descent (SGD) [125] performs a parameter update for each training sample $(\mathrm{x}^{(k)}, \mathrm{y}^{(k)})$,
- Mini-bath gradient descent performs an update for every mini-bath of n training samples $(\mathrm{x}^{(k)}, \mathrm{y}^{(k)})$.

The choice of one approach depends on the application and it relies on both the training set provided and the costs for the tuning of the parameters [52]. Specifically, the mini-batch approach divides all training data into mini-sets and computes the gradient for each of them. As it was mentioned above, partitioning the computation of the gradient into smaller clusters of data might be beneficial. Nevertheless, while this approach helps prevent getting stuck in local minima, the loss function may not exhibit a consistent, monotonic decrease [126]. For this reason, choosing the optimal batch size is crucial in DL algorithms to ensure accuracy and reduce generalisation errors. The batch size is the result of an accurate trade-off decision. On one hand, larger batches provide a more accurate gradient estimation with converging to a sharp minimum, at the cost of losing generalisation performance. On the other hand, smaller batches consistently converge to a flat minimum, reducing the generalisation gap but with a consequent introduction of noise into the gradient estimation. Therefore, the learning rate needs to be reduced to ensure stability. As a consequence, the steps needed to exploit the entire training set increase, leading to a higher learning time [127]. To contain this effect, learning rate schedules support the designers in adjusting the learning rate during the process by reducing it according to a predefined schedule. However, these schedules and thresholds are predetermined based only on the experience of the developer. In addition, the same learning rate is applied to all parameter updates, therefore it is not suitable for applications with a sparse training dataset. In such cases, it may be desirable to adjust individually the updated magnitudes of each neuron to better accommodate the data sparsity [52].

There are many variants of the gradient descendent algorithm, such as Adaptive Moment Estimation (Adam) [93], Adaptive Gradient Descent (Adagrag) [128], Root Mean Square Propagation (RMSProp) [52] and Momentum [129].

Momentum and Nesterov momentum (NAG) [130], are two techniques designed to speed SDG convergence in the appropriate direction by reducing oscillations and dumping the effect of the noise in the gradients. It is important to note that the SGD algorithms tend to get trapped in sub-optimal local minima during the minimisation of highly non-convex error functions, therefore it became necessary to define some solution to speed up the learning process. [124].

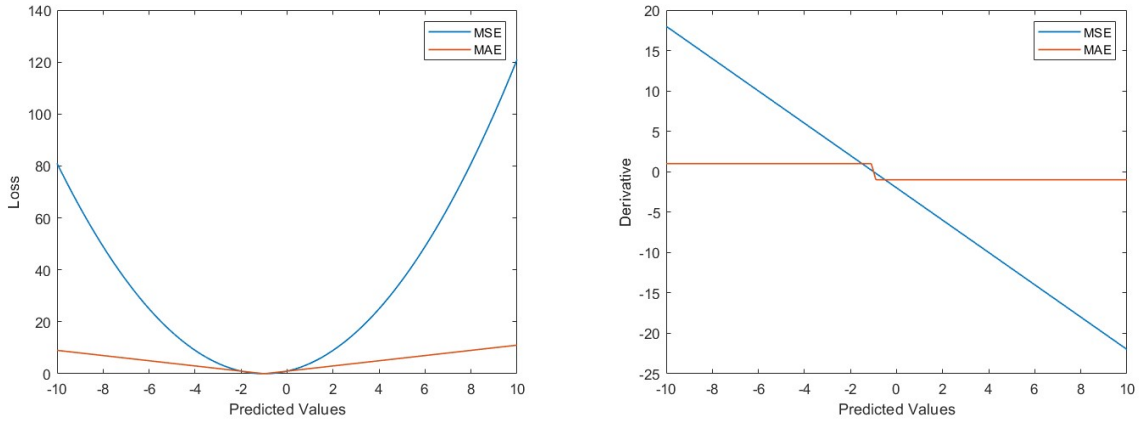
Adagrad is an algorithm well suited for managing sparse datasets, by dynamically adjusting learning rates according to individual parameters. The updating is larger in magnitude for low-frequent parameters and smaller for high-frequent parameters. Consequently, the algorithm is highly effective in convex problem settings, where it quickly converges to minimise objective functions. However, it may encounter performance issues in non-convex problems due to the monotonic learning rate decay. To address this limitation RMSProp has been introduced. The gradient accumulation mechanism of Adagrad is substituted by an exponentially weighted moving average. The latter mechanism prevents an excessive decrease in the learning rate for small gradients.

Adam computes adaptive learning rates for each parameter, combining RMSProp and moment concepts. In Adam, the algorithm integrates momentum directly into its calculations by employing an exponentially weighted moving average of the gradients, which helps estimate the first-order moment. This integration allows Adam to efficiently track the history of gradients, enabling faster convergence during the optimisation process. One of the key advantages of this optimisation algorithm is its ro-

bustness to hyperparameter choices. Moreover, due to its ability to adjust learning rates and combine momentum-based techniques, it has demonstrated superior performance compared to other adaptive learning-method algorithms. For these reasons, it is the preferred choice for many applications in the field [93].

Loss functions

The loss functions measure the difference between the predicted output and the actual output of a network, enabling the adjustment of model parameters during the training process in order to minimise the dissimilarities. Consequently, the loss function has to capture the inherent characteristics of the problem and contemporarily represent the desired design objectives [131]. Several loss functions have been developed in the context of DL training. However, in this instance, we will focus on the one specifically tailored for regression problems. It is known that the objective of a regression problem is to forecast a continuous real-valued quantity. Typically the output layer is configured with a linear activation unit, therefore the preferred functions in this type of architecture are Mean Square Error (MSE) or Mean Absolute Error (MAE) [110].



(a) Loss functions. (b) Loss function gradients.

Figure 2.5: Loss functions MAE and MSE (a) and their gradients (b), computed with $\widetilde{y} = 2x - 1$ and $y = x$. These plots were generated using MATLAB, drawing inspiration from the matters found in [110].

The MSE is calculated by taking the average of the squared differences between the real values, $\mathrm{y_i}$, and the predicted values, $\widetilde{\mathrm{y}}_\mathrm{i}$,

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\widetilde{\mathrm{y}}_i - \mathrm{y}_i)^2, \tag{2.6}$$

with N equal to the number of data points. The computation of the squared term within the MSE has the effect of making all errors positive and this amplifies the influence of data points with larger errors. Moreover, the convergence of the model during the training is accelerated, as the gradient of the loss is directly connected to the magnitude of the errors. Nevertheless, MSE is sensitive to outliers and this sensitivity can represent a challenge when the dataset is compromised. Indeed, the outliers can potentially guide the training process towards suboptimal models [131].

The Mean Absolute Error (MAE), is a loss function that computes the average absolute difference between the real and predicted values,

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\widetilde{\mathrm{y}}_i - \mathrm{y}_i|, \tag{2.7}$$

MAE is a resilient and robust loss function, recommended for training datasets that might include outliers. Nonetheless, MAE comes with its limitations. For example, it is affected by a lack of smoothness, resulting from the discontinuity in the derivative at zero. Moreover, both small and large errors are uniformly treated, owing to the constant gradient of the loss function. The inability to distinguish between errors of different magnitudes can result in potential convergence issues [131].

2.2 Aerodynamics

Aerodynamics, as emphasised by McLean [132], is the cornerstone of the global aerospace industry. It delves deep into understanding how air interacts with immersed objects, focusing on dynamic forces. To navigate this complex realm, simplified theoretical models are essential tools which aim to predict and comprehend the interactions during the flight. These models provide critical insights into lift, drag, stability, and performance, guiding from concept to reality.

2.2.1 Aerofoil Geometry

Regardless of the type of lifting surface, the aerodynamic characteristics of a wing are strongly affected by the shape of the wing section, also called aerofoil. An aerofoil is such a shape that when it is placed in an air stream, it produces an aerodynamic force, as highlighted by Patel et al. [133]. The basic geometry of an aerofoil is shown in Figure 2.6. An aerofoil performance is further influenced by these geometric features. The shape of the leading edge (LE) is crucial for the smoothness of airflow, impacting the lift and drag generation, while the trailing edge (TE) shape hardly affects the nature of airflow separation. Camber, by altering pressure distribution, plays a role in force generation. Moreover, the thickness affects not only the force distribution but also the structural strength, with thinner aerofoils characterised by lower drag but potential structural trade-offs.

2.2.2 Aerodynamic Coefficients

When a body travels through a fluid, it encounters aerodynamic forces generated by the interaction between the object and the fluid. The sources of the aerodynamic lift, drag, and moments on a body are the pressure and shear stress distributions integrated over the body [134]. The lift, L, is defined as the perpendicular component of the resultant force to the direction of the relative wind, whereas the drag, D, is the projected component along the same direction of the air velocity and opposes the motion of the body. The pitch moment, M, is instead defined as the rotational force that causes an aerofoil to rotate

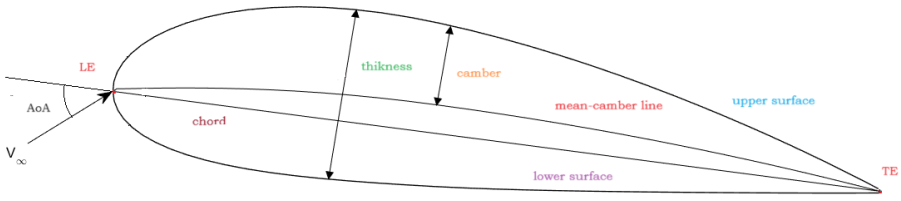


Figure 2.6: Aerofoil geoemetry.

around its lateral axis. For a given attitude of geometrically similar aerofoils, the forces tend to vary directly with the density of the air, the chord, and the square of the speed. It is accordingly convenient to express these forces in terms of non-dimensional coefficients [72]. Therefore, aerodynamic coefficients are defined as dimensionless forces and moments which are widely used to compare the performance of different bodies under different conditions. They relate the forces with geometrical parameters, such as the characteristic length of the body and the shape, and flight conditions such as the angle of attack, the velocity of the freestream and the properties of the fluid. The dynamic pressure is defined as,

$$q_\infty = \frac{1}{2}\rho_\infty V_\infty^2, \tag{2.8}$$

where ρ_∞ and V_∞ are respectively the density and the velocity in the free-stream condition. The lift, drag and moment coefficient of an aerofoil are therefore expressed as a function of the dynamic pressure as follows:

$$C_l = \frac{\mathrm{L}}{q_\infty c}, \tag{2.9}$$

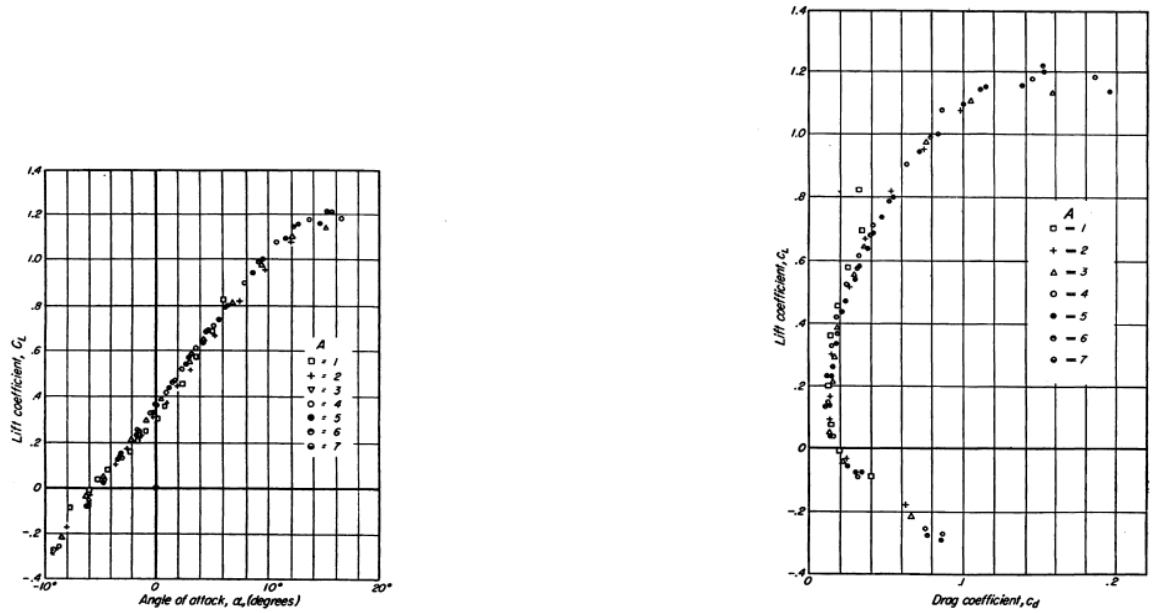
$$C_d = \frac{\mathrm{D}}{q_\infty c}, \tag{2.10}$$

$$C_m = \frac{\mathrm{M}}{q_\infty c^2}, \tag{2.11}$$

where c is the chord of the aerofoil. It is common to find the lift and drag coefficients expressed in counts, respectively 10^{-4} for one C_d count and 10^{-3} for one C_l count. A convenient way of describing the aerodynamic characteristics of an aerofoil is to plot the values of the coefficients against the angle of attack. AoA is the angle between the oncoming air and the reference chord. [133]. For a given free-stream velocity V_∞, the lift coefficient C_l varies linearly with α, as shown in Figure 2.7. As the AoA of a fixed aerofoil geometry increases, the separation of the airflow from the upper surface becomes more pronounced. This phenomenon leads to a reduction of the increasing rate of the lift coefficient [133]. This trend is preserved until the stall condition is reached, with a drastic decrease in performance and a considerable increase in the drag.

A second important graph to consider is the plot of the efficiency, given by the C_l/C_d ratio. This

graph, often referred to as the *"drag polar"*, illustrates the physical relation between the lift and drag forces. As seen in Figure 2.7 this ratio starts at zero when there is zero lift (i.e., at low angles of attack), then it steadily increases to reach its maximum value at a moderate lift coefficient. Beyond this point, as the AoA is increased, the ratio begins to decrease slowly.



(a) $C_l - \alpha$ (b) Drag polar

Figure 2.7: Exeperimental results from *Theory of wing section, Abbott and Doenhoff [72]*.

2.2.3 Flow Regimes

Assuming an inviscid and steady flow, it is possible to categorise various flow regimes based on the Mach number defined as:

$$M = \frac{V}{a}, \tag{2.12}$$

where a corresponds to sound speed and is physically defined as the speed at which pressure waves propagate through a medium [134]. At first glance, a flow is defined as incompressible if the density ρ is constant. Conventionally for $M < 0.3$, it is always assumed that $\rho = constant$. In contrast, a flow where the density is variable is called compressible.
According to J. Anderson [135], for:

- $M_\infty \leq 0.8$ the flow is generally completely **subsonic**;
- $0.8 \leq M_\infty < 1$ the flow expansion on the upper surface of the aerofoil may result in locally supersonic regions, with a resulting flow described as **transonic**;
- $M_\infty > 1$ a flowfield is defined **supersonic**.

In the context of this thesis, we will offer a brief introduction to the subsonic and transonic regimes. Furthermore, to comprehensively address this subject, we will present a theoretical formulation of the flow potential for both of these regimes.

Subsonic flow field

Subsonic flows are characterised by smooth streamlines with no discontinuity in slope [134]. Since the flow velocity is everywhere less than the speed of sound, disturbances propagate both upstream and downstream and are felt throughout the entire flow field. The subsonic flow is subdivided into two sub-fields: one pertaining to incompressible flow and the other to compressible flow. Each of these sub-fields has its own distinct aerodynamic theory [134].

The aerodynamic theory for incompressible flow over thin aerofoils at small angles of attack was presented by Prandtl in 1930 [136]. Assuming a two-dimensional, irrotational, and isentropic flow, under the assumption of small disturbances, it is possible to derive the perturbation velocity potential equation written in terms of M_∞ :

$$(1 - M_\infty^2)\frac{\partial^2 \hat{\phi}}{\partial^2 x} + \frac{\partial^2 \hat{\phi}}{\partial^2 y} = 0 \tag{2.13}$$

where $\phi = V_\infty \cdot x + \hat{\phi}$ is the velocity potential ($V_\infty \cdot x$), defined as the sum of the mean potential and a small perturbation ($\hat{\phi}$). Eq. 2.13 is a linear partial differential equation, used to easily obtain the pressure distribution along the surface of a slender body in the subsonic regime. However, with the coming of World War II and the fast growth of the aerospace field, the incompressible flow theory was no longer applicable to aircraft of the time. Because the main aerodynamic tests have been collected over the years in low-speed set-ups, the natural approach was to add some compressibility correction to the existing incompressible flow [134]. A result easily gained with this approach is the pressure coefficient distribution in Equation. 2.14:

$$C_p = \frac{C_{p,0}}{\sqrt{1 - M_\infty^2}}. \tag{2.14}$$

Equation 2.14 is called the *Prandlt-Galuert rule* and states that if we know the incompressible pressure distribution over an aerofoil ($C_{p,0}$), then the compressible pressure distribution over the same aerofoil can be obtained from the above relation [134].

Transonic flow field

The transonic flight regime is one that all high-speed vehicles experience during an acceleration or deceleration through $M = 1$. Transonic flow historically has been an exceptionally challenging problem in aerodynamics since it is characterised by mixed regions of locally subsonic and supersonic flow that occur over a body moving at Mach numbers close to the unity [135].

To understand the physics of the transonic regime, it is necessary to define the shock wave in an extremely thin adiabatic region, typically on the order of 10^{-7} m, across which the flow properties can change drastically registering strong gradients. This phenomenon is a consequence of the high speed since small pressure perturbations cannot propagate upstream in a supersonic region [134].

By definition, the critical Mach number M_{cr} is that free-stream Mach number at which the aerodynamic drag on an aerofoil begins to increase rapidly and the transonic regime forms. Starting with an aerofoil flowing at $M = M_{cr}$, it is possible to observe with experiments a pocket of supersonic flow

extending from just downstream of the leading edge to about 35% of the chord length, where it is terminated by a nearly normal shock wave. As the Mach number increases [135]:

- The shock becomes more intense causing a separation of the viscous boundary layer in the region where the shock impinges on the surface;
- The impinging point moves towards the LE until the Mach number reaches the threshold of 1 and, according to the shape of the body, the shock becomes an oblique or normal shock.

The separated flow associated with the shock/boundary layer interaction is caused by the strong gradient of the pressure through the shock. The strong discontinuity across the pressure field represents an adverse gradient, where the pressure increases in the flow direction. Therefore, when the shock wave impinges on the surface the boundary layer encounters the adverse pressure gradient and separates. Along with the total pressure losses and the increasing entropy, the drag-divergence phenomenon outcomes from the induced separation of the flow [135].

A second effect is appreciated due to the interaction shock/boundary layer. Downstream of the separation point, a bubble made of a recirculating flow starts growing with a consequent energy transfer from the outer high-speed flow toward the separated region.

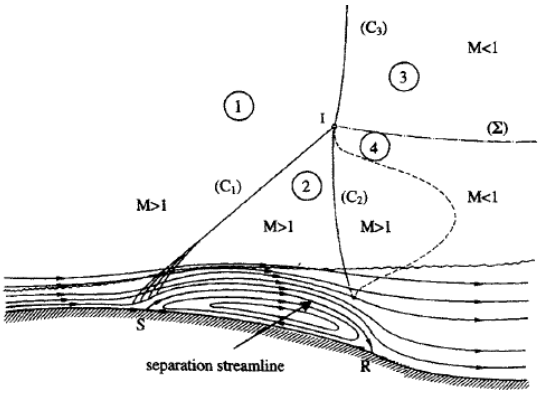


Figure 2.8: Sketch of a λ shock [137].

Additionally, for $M > M_{cr}$ the boundary layer separation induces an oblique shock after which the flow is still supersonic. The interaction between the oblique shock and the normal shock wake generated in the inviscid boundary layer results in the typical structure of a lambda shock pattern, sketched in Figure 2.8. Above the pressure line starting from the conjunction point, the flow is still supersonic. Below the separation line, the flow is subsonic with a consequent suction of the normal shock upstream, increasing the strength of the aforementioned oblique shock [137].

Theoretically, all the mentioned phenomena can not be treated directly with the linear potential equation reported in Eq. 2.13. Some corrections have been made in the transonic case as follows:

$$(1 - M_\infty^2)\frac{\partial^2 \hat{\phi}}{\partial^2 x} + \frac{\partial^2 \hat{\phi}}{\partial^2 y} = M_\infty^2 \left[(\gamma + 1)\frac{\partial \hat{\phi}}{\partial x}\frac{1}{V_\infty}\right]\frac{\partial^2 \hat{\phi}}{\partial x^2} \tag{2.15}$$

The non-linear right-hand side term is not neglected in the linearisation and requires to be solved by numerical methods during the computation [135].

2.2.4 Computational Fluid Dynamics

As it is stated by Anderson [138]: *"Computational fluid dynamics (CFD) is the art of replacing the integrals or the partial derivatives in these equations with discretised algebraic forms, which in turn are solved to obtain numbers for the flow field values at discrete points in time and/or space"*.
CFD codes are indeed structured around the numerical algorithms that can tackle fluid flow problems [139]. It is known that the Navier-Stokes governing equations are not analytically solved, therefore it has been necessary to implement algorithms able to address the designer's needs. Many complex aerodynamic flow fields which had never been approached before, have been therefore solved deriving the aerodynamic coefficient with an acceptable accuracy.

Nowadays, the results computed by a CFD code are the best performances that can be collected after the experimental results. In the designing process, CFD has become a vital component due to the several advantages over the experiment-based approach. Firstly, studying the systems, where controlled experiments are difficult or impossible to perform has become possible. Secondly, once the requirement of high-computing hardware is accomplished, the tool has an unlimited level of detail of results [140].
During the pre-processing phase, the computational domain is opportunely studied in order to be discretised during the grid-generation step. Conventionally each element of the grid (or mesh) is called cell (or control volume or element). After that, both the physics and the boundary conditions of the phenomenon are studied to be modelled in the code. It was estimated that more than 50% of the time spent in industry on a CFD project is devoted to the definition of the domain geometry and mesh generation [139]. This significant portion of project time underscores the critical importance of these preparatory stages in ensuring the accuracy and reliability of CFD results. Given the substantial influence of assumptions on the outcomes, it becomes imperative to validate CFD results whenever possible, either through comparison with experimental data or by seeking analytical solutions. This validation process helps establish the credibility and trustworthiness of the CFD simulations in practical applications [139].

Core solvers are implemented with several numerical solution techniques: the finite difference approach (FD), the finite element or fine volume method (FVM) and the spectral modes. In essence, each of these aforementioned numerical algorithms involves integrating the governing equations across all control volumes by discretising the integrals into a linear system of algebraic equations. While CFD codes adeptly discretise spatial and temporal aspects of various flow phenomena, encompassing convection, diffusion, and source terms, the ultimate solution is achieved through an iterative process aimed at mitigating the inherent errors arising from discretisation. ultimately, during the post-processing step, the simulation solution can be examined and assessed through flow visualisation and quantitative data analysis [139, 140].

Chapter 3

Methodology

This chapter outlines in detail the methodology and the assumptions of the proposed data-driven optimisation framework. In Section 3.1 a complete overview of the framework is provided, justifying the implementation of each tool and highlighting the advantages. Next, in Section 3.2 the construction of the database used to train the SM is detailed, explaining the assumptions and the parameterisations made in this stage. Section 3.3 discusses the chosen aerofoil shape parameterisation, detailing its advantages and disadvantages and emphasising the implementation of the geometric constraints for the specific application. In Section 3.4 the architecture of the developed DL SM is streamlined, with a comparison with the state-of-the-art. Finally, the optimisation schemes chosen for the multi-objective and multi-point optimisations are presented in Section 3.5.

3.1 Framework Overview

The aim of this work is to develop a surrogate-based optimisation framework to address the challenges of the aerodynamic shape optimisation of a morphing aerofoil in subsonic and transonic regimes. The optimisation methodology is comprehensive of a DL model for the prediction of the aerodynamics, as it has been demonstrated to provide accurate and fast data-driven solutions [40, 50, 51, 56, 58, 59, 63]. As stated in Section 1.3.4, ASO of morphing aerofoils has attracted the interest of the researchers. This interest revolves around the potential enhancements in terms of efficiency and greening objectives, specifically addressing the reduction of pollution impact, achievable through the utilisation of morphing architectural designs. In the ASO state-of-the-art, the common approach is to construct a model to directly map the parameter space, which is explored by the optimisation scheme, to the aerodynamic coefficients of a given aerofoil. Nevertheless, reviewing recent publications it becomes evident that data-driven approaches to aerodynamic analysis and optimisation of morphing aerofoils are consistently constrained by a limited design space. Moreover, several studies are predominantly focused on a single flight regime [59, 108], as highlighted in Table 1.1.

In addition, recent studies have developed a data-based framework for the optimisation of aerofoils across a wide range of Mach and Reynolds numbers [55, 83]. However, it is important to note that the

implemented methodologies do not consider the constraint of a wing box embedded within the wing section. Ensuring the conservation of the constrained wing box is a critical consideration in the design of an aircraft wing. The wing box indeed serves as a crucial component, guaranteeing structural integrity, resistance to torsion, fuel storage capacity, and responsiveness to aeroelastic effects [11, 13, 94, 95].

To address these limitations, this work presents a DL-based approach to construct a SM that receives the raw data of the aerofoil geometry to predict its aerodynamic coefficients, C_l, C_d and C_m. Given the vast number of publications related to this topic, the SM approach involves customising a pre-existing code originally developed by Moin et al. [59]. The developers initially trained the neural network on an extensive dataset, exclusively comprising NACA-series aerofoils, within the low-subsonic incompressible regime, covering Mach numbers ranging from 0.1 to 0.3. Considering the remarkable performance of the model, the author of this thesis has decided to adapt the pre-distributed multi-layer perceptron and further train it for a broader collection of aerofoil shapes. By doing so, we aim to extend its applicability across a wider spectrum of Mach and Reynolds numbers. Among different available DL models, this architecture was chosen for several reasons. Various algorithms and architectures have been proposed by the research community to address the issue of low-budget aerodynamics evaluations. Among these models, the MLP developed by Du et al. [40] and the work presented by Junior et al. [83] have proven great efficiency and accuracy. However, despite their differences, these proposals share a common trend. Specifically, in addition to the limited diversity of aerofoils used for training, each approach separates the computation of the C_l and of the C_d into different architectures. Segregating the computations of the two coefficients would not be advantageous in terms of the costs and time required for building the SM and accomplishing the training.

To build effective and representative data-based solutions it is essential to ensure accuracy and efficiency of the SM. This requires considering only feasible geometries, avoiding abnormal shapes as stated in Section 1.3.2. To achieve this, the UIUC library [61] and the NASA supercritical SC(2) library [141] are considered to create a dataset of realistic shapes that can be used to train the ML tool. The libraries provide a vast collection of aerofoils from subsonic to transonic applications, covering the desired design space and providing a diverse spectrum of shape features.

To ensure the conservation of the wing box, a free-form deformation technique is employed. This technique effectively constrains the degrees of freedom (DoF) of each point within the lattice box in which the aerofoil is embedded. Design variables obtained with FFD are superior in clear physical meaning and feature for design operation [142], as detailed in Section 1.3.2. Moreover, it is widely recognised that the FFD approach has been outperformed by data-driven geometric parameterisation techniques, such as the GAN developed by Chen et al. [63]. Nonetheless, these advanced data-driven methods do not permit constraints on specific parts of the shape, making them unsuitable for our specific purposes.

The proposed framework for real-time aerofoil shape optimisation comprises three primary modules. The module ”3: Surrogate model (MLP)” implements an MLP, which maps the geometry of an aerofoil to its corresponding coefficients in subsonic and transonic conditions, without recurring to expensive CFD

simulations. The module "2: Geometry parameterisation (FFD + sampling)" involves the FFD technique, which embeds the aerofoil within a bounding box, enabling the representation of the aerofoil geometry in a low-dimensional space and ensuring adherence to the desired constraints of the wing-torsion box. These initial two modules operate in tandem with the optimisation solver, which takes advantage of their combined effort to explore the parameter space using a GA. Figure 3.1 presents an overview of the proposed workflow.

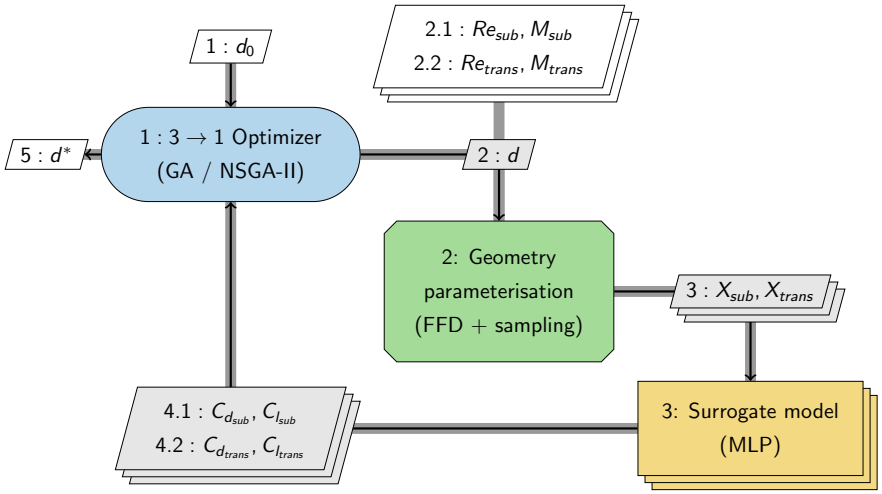


Figure 3.1: DL-based framework for ASO. The process and data dependencies are illustrated via the Extended Design Structure Matrix [143]. The diagonal nodes in the diagram represent process components, while the off-diagonal nodes indicate the data transferred between them. The off-diagonal white nodes are the secondary input of the flowchart. Thick grey lines denote the data flow, and black lines indicate the process flow.

This thesis work aims to contribute significantly to the exploration of the capabilities of the existing data-driven approach for aerodynamic shape optimisation in morphing aerofoil, addressing the limitations of prior methods. Although the key modules implemented are not built entirely from scratch, they were trained and fine-tuned to attain optimal performance within the proposed framework.

3.2 Database Creation

To achieve the aerodynamic optimisation of a morphing aerofoil, this work has developed a data-driven framework. Since the DL model chosen is trained via supervised learning for regression purposes, a database of input and output pairs is required. Nevertheless, to the best of the author's knowledge, all the available open-source database are not suitable for the selected application, due to the restrictions of both the considered design space and the simulated flight range. Therefore, in order to achieve the optimisation tasks, a high-fidelity database with geometry and aerodynamic coefficient pairs was created using aerofoil geometries suitable for both subsonic and transonic conditions. Furthermore, for each aerofoil shape, a broad range of flight conditions was simulated to explore the full spectrum of the SM capabilities, encompassing its limits and potentials.

The process of building the dataset begins with the pre-processing of the UIUC Aerofoil Coordinates

Database [61] and NASA SC(2) Database [141]. Both libraries provide a collection of real aerofoils, defined by a set of (x_i, y_i) coordinates, However, there is a disparity in the chordwise locations from which these points are taken for each aerofoil. This lack of uniformity in data format can result in inconsistencies. Therefore a standardisation process is required to establish a common distribution of aerofoil coordinates among all aerofoils. Furthermore, a mesh convergence study is undertaken to define a consistent discretisation of the computational domain used in the CFD simulations. The ultimate grid selection for generating the database is the outcome of a careful balance between computational resources, uniformity, and accuracy, ensuring consistency across all considered flight regimes. Finally, upon reaching the computational budget and collecting a discrete and relatively large dataset, a sampling tool is employed to condense each aerofoil geometry for each flight condition within the database file into a set of strategically selected data points.

Three open-source software tools are employed to generate the training and validation set for the SM, *pyHyp* [144, 145] for meshing generation, *ADflow* [146] for CFD simulation and *prefoil*[1] for sampling. These tools are favoured for their user-friendliness and thorough documentation. Indeed, each program comes equipped with a *Python* API, which proves invaluable in automating the process of simulating hundreds of thousands of samples. While *ADflow* is compatible only with three-dimensional meshes, it was selected as the optimal tool to work in tandem with *pyHyp*. The decision was driven by the fact that *pyHyp* is a hyperbolic volume mesher that extrudes structured surface meshes into volume meshes in a matter of seconds. *ADflow* requires a CFD Generated Notation System (CGNS) [147] file which is completely supported by the specifications of *pyHyp*. Additionally, based on the author's experience, *ADflow* has demonstrated superior performance, particularly in the context of aerodynamic optimisation, ensuring stability and improved conditioning [40, 50, 51, 81, 83].

3.2.1 Aerofoils Preprocessing

The preprocessing procedure adopted in this thesis is the one proposed by Li et al. [51]. The standardisation process involves the following steps to ensure consistency and uniformity of the shape format:

a) the original aerofoil surface is interpolated by a B-spline curve;

b) the trailing edge is sharpened, ensuring a zero thickness;

c) the leading edge is defined as the farthest point from the trailing edge;

d) the chord is normalised to one, scaling, therefore, the aerofoil;

e) the chord plane is rotated to achieve zero AoA;

f) the point distribution scheme for the aerofoil is standardised using a half-cosine spacing distribution with N points for each surface:

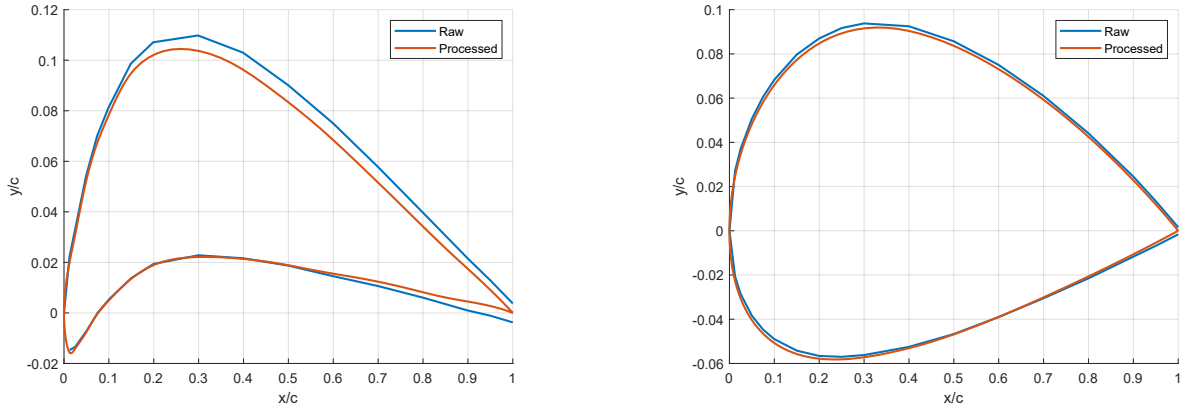
$$x_i = \frac{1}{2}\left\{\cos\left[\frac{2\pi(i-1)}{N-1}\right] + 1\right\}, \quad i = 1, 2, ..., N. \tag{3.1}$$

[1] https://github.com/mdolab/prefoil

g) the aerofoil surfaces are smoothed using a Laplacian smoothing algorithm:

$$y_i^{(k+1} = \tau(x_i)\left(y_{i-1}^{(k+1)} + y_{i+1}^{(k+1)}\right) + \left(1 - \tau(x_i)\right)y_i^{(k)}, \tag{3.2}$$

where τ is a weighting function that increases the aerofoil camber at each point [51].



(a) Gottingen 207.

(b) NACA 2415.

Figure 3.2: Effects of preprocessing procedure [51] on example aerofoils: Gottingen 207 (a) and NACA 2415 (b). The aerofoils are not plotted with an aspect ratio of 1:1 to emphasise the differences.

The procedure ensures a high-quality geometry aerofoil, which is necessary to ensure precise results in CFD simulations. Steps a) and b) are involved to guarantee the suitability of the aerofoil shape for mesh generation. Step c) identifies the leading edge, which is necessary for the normalisation and rotation tasks indicated in steps d) and e). Even though the aerodynamic coefficients are not a function of the chord, except for the moment coefficient, the normalisation task remains important. Normalisation serves for comparing results and managing the variety of different aerofoil shapes involved in the analysis. Furthermore, the AoA is excluded from the aerofoil geometry since it is considered as the incidence of the incoming flow with the chord plane during the CFD simulations. The implementation of AoA in the dataset creation is discussed in Sec 3.2.2, where the flight conditions are set up in detail.

In this context, a cosine function is chosen over a uniform distribution as it is stated in step f). The former has been chosen because it can better capture the shape features, especially near the edges. Given that the leading edge significantly influences the drag, while the trailing edge is crucial for understanding wake behaviour, the cosine distribution is preferred as it can also provide more relevant data for the simulation model [51, 148]. Indeed, using a half-cosine distribution yields a higher concentration of points near the edge, while the uniform distribution might lead to imprecision during the simulation because it does not accurately inspect the shape in these two critical zones. Moreover, the number of points chosen for the standardisation is crucial in the computation of the transonic flight regime. A precise deception of the impinging point of the lambda shock is necessary to ensure the accuracy of the simulation. Further elaboration and specific details are provided in Section 3.2.4.

Lastly, the aerofoil surfaces undergo streamlining through the smoothing of the y-coordinates, in step g). This is done to prevent spikes in the velocity distribution during CFD simulations. The process is

considered complete when the difference between the final and initial iterations of the y-coordinates, denoted as $\|y(k) - y(0)\|/\|y(0)\|$, is more than a threshold of 0.3% [51]. Two examples of the preprocessing procedure are presented in Figure 3.7. On both considered aerofoils, the sharpening of the trailing edge, the normalisation of the chord, and the surface smoothing are clearly the most important steps to improve the aerofoil geometry.

3.2.2 Computational Fluid Dynamics Simulations Setup

Thermodynamic Setup

The main purpose of this thesis work is to optimise by allowing the vertical displacements of a morphing aerofoil with a data-based framework. Morphing architecture is usually heavier than its non-morphing counterparts. This increased weight can be attributed to several factors, such as the requirement of additional mechanisms and actuators to facilitate the shape-changing abilities. Additionally, the inherent need for structural compliance in specific areas of the aircraft contributes to its increased weight. Morphing vehicles are therefore preferred in those missions with multiple varied flight phases, where the non-morphing solution will be highly compromised in design. In order to ensure practical applicability, a hypothetical mission profile has been established prior to dataset creation. Drawing the mission profile serves as the foundation for defining the thermodynamic setup and accurately parameterising the flight condition variables.

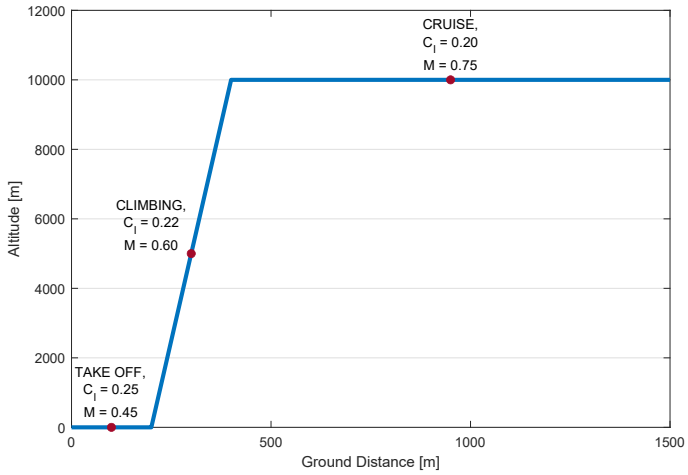


Figure 3.3: Hypothetical Aircraft Mission Profile: from Sea Level to High Altitude Cruise

In the initial stages of the project, the objective was to optimise an aerofoil from takeoff to the cruise altitude. Successively, the optimisation mission profile, detailed in Section 4.1.2, has been streamlined compared to the initial hypothetical scenario, due to the limited performance of the aerodynamic surrogate model for some flight conditions. Nevertheless, at this point of the thesis work, we address the issue assuming the mission profile represented in Figure 3.3.

Developing an accurate and high-fidelity dataset is crucial for ensuring the optimal performance of the surrogate model. Drawing upon the state-of-art, SM-based ASO and available the open-source

dataset [50, 55, 59], each simulated shape is represented in the dataset by the Reynolds number, the Mach number and the AoA. To represent both subsonic and transonic regimes, a wide range of Mach numbers is examined, from 0.35 to 0.85 with a step of 0.1.

To enhance the SM, we opted to keep the Reynolds number constant with respect to altitude, despite its sensitivity to changes in Mach number. The Reynolds number is the ratio of inertial forces to viscous forces,

$$Re = \frac{V_\infty \cdot c}{\nu}, \tag{3.3}$$

where $V_\infty = a \cdot M$ is the free stream velocity, c is the aerofoil chord and $\nu = \mu/\rho$ is the kinematic viscosity [134]. The Reynolds number indicates how the viscous effects are influenced compared to the inertial effects resulting from the fluid motion. Depending on its values, the flow over the aerofoil exhibits different characteristics that need to be represented in the dataset. The dependency of the Reynolds number on the Mach number is sketched in Figure 3.4, with respect to the altitude. Nevertheless, for the sake of dataset consistency and adhering to the imposed computational budget, a Reynolds number value was assumed, computed with a fixed Mach number of 0.6. Moreover, keeping the Mach number constant allows us to isolate the effect of altitude variations on the dataset. So in conclusion, in the dataset, a Reynolds number equal to [1.4 x 10^7, 8.7 x 10^6, 5.1 x 10^6] is simulated for each Mach number.

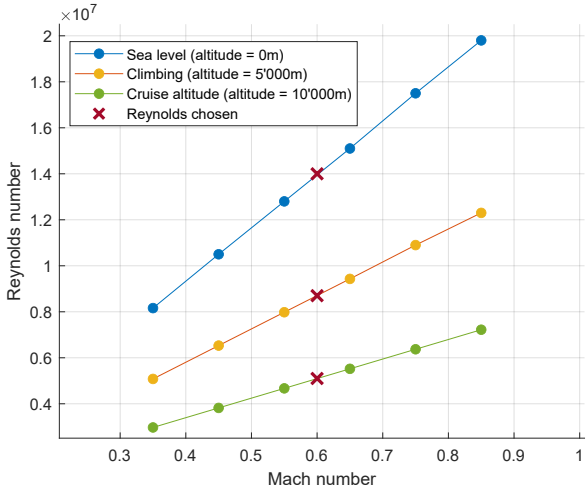


Figure 3.4: Variation of Reynolds number with respect to Mach number at a fixed altitude

Table 3.1: Designed flight condition setups for each aerofoil geometry

Parameter	Range	Total
Reynolds	1.4 x 10^7 - 8.7 x 10^6 - 5.1 x 10^6	3
Mach	0.35 - 0.45 - 0.55 - 0.65 - 0.75 - 0.85	6
AoA	0-1-2-3-4-5-6	7
Simulations for aerofoil		126

Lastly, the AoA is adjusted with the typical mission profile of commercial aircraft, which accounts for varying incidences relative to the free stream based on different mission phases and lift requirements.

Each aerofoil was simulated at AoAs between 0° and 6° with a step of 1°, achieved by altering the AoA of the incoming flow. The parameters are summarised in Table 3.1.

Numerical Setup

All CFD simulations used in the data generation are performed with *ADflow* software. *ADflow* is a finite volume structured multiblok and over-set mesh solver [146], which has the option to solve the steady RANS equations [31]. Considering the wide range of flight conditions and their varying physics in terms of viscosity and friction, RANS equations are the best choice for capturing relevant phenomena in both subsonic and transonic regimes [134]. The Spallart-Allmaras (SA) is chosen as the turbulence model since it is the recommended option for external flows [149]. Moreover, the solver implements the approximate Newtown-Krylov (ANK) algorithm that has been demonstrated to be sufficiently robust as a globalisation scheme for the full Newtown-Krilov (NK) solver [31]. The ANK robustness ensures the convergence of the SA turbulence model used in this application. To compute all the samples of the dataset, we choose a 5-th order decay in total residuals as a convergence criterion.

The boundary conditions of the problem consider the aerofoil surfaces with no-slip conditions. The outermost face of the extruded mesh is set up as a far field, which is automatically handled by *ADflow* consistently with the free stream flow direction. The surfaces of cells with unattached edges have symmetrical conditions. Furthermore, due to the requirement of *ADflow* for a 3D mesh even for 2D applications, the CFD simulation of an aerofoil necessitates the imposition of a symmetry boundary condition in the artificial ”span-wise” direction.

3.2.3 Mesh Convergence Study

As stated in Section 2.2.4, CFD simulations depend on the FVM, which entails the need for spatial discretisation of the domain. Therefore, the domain needs to be partitioned into small and not overlapping cells, which all together form a grid. The discretising grid, often assessed as mesh, plays a significant role in capturing the flow physics and obtaining meaningful results since the RANS equations are numerically solved in each volume. For this reason, assessing the impact of the grid on the results is crucial to ensure accuracy and reliability of the CFD simulation.

Several types of uncertainties and errors can cause CFD simulation results to differ from the corresponding experimental reference. discretisation errors are those errors that occur from the representation of the governing flow equations as algebraic expressions in the discrete domain. For a consistent numerical method, it is expected that the error introduced by the spatial discretisation approaches zero as the number of grid points increases and the size of the grid spacing tends to zero [139]. As the mesh is refined, it is expected that the numerical solution approaches the continuum solution, becoming less sensitive to the grid spacing and reaching the so-called grid convergence. Nevertheless, refining the mesh increases the amount of computational power needed to perform the CFD simulations since the solver needs to handle a larger system of equations. Therefore, a mesh convergence study needs to be performed in order to determine the optimal level of discretisation that yields the best balance between

achieving accuracy in results and managing computational costs. In this study, the primary objective was to identify a standardised mesh that can effectively simulate different aerodynamic geometries and flight scenarios. To achieve this, multiple factors are taken into account including mesh-related error, solution time and convergence robustness across a range of different flight conditions.

As it was aforementioned in Section 3.2, *pyHyp* employs a hyperbolic volume mesh marching technique to extrude the volume meshes from the surface meshes. The hyperbolic mesh generation ensures both the orthogonality of the mesh and the cell volume specification [144]. Firstly for each aerofoil geometry, once the coordinates are provided, *pyHyp* mesher automatically generates a surface mesh with only one cell in the perpendicular direction to the aerofoil. This step is necessary since *ADflow* solver supports only three-dimensional meshes. Secondly, the surface mesh obtained is then expanded in successive layers until it reaches a distance that no longer has a significant impact on simulation results. Additionally *pyHyp* enhances hyperbolic mesh generation for high-quality meshes by adding spatially variable smoothing coefficients, metric correction procedures and local treatments of sever convex corner [144]. Once the final volume mesh is generated, it is saved in a CGNS file format that is directly used by the *ADflow* solver.

The most influential parameters for meshing that are directly defined by the user are:

- number of nodes in off-wall direction;
- thickness of first off-wall cell layer;
- marching distance between the first layer and the outermost edge.

An iterative and detailed study is conducted to properly select each parameter. The thickness of the first off-wall layer attached to the aerofoil wall is crucial for capturing the viscous effects in a turbulent flow. The guidelines specify that the thickness has to be lower than the estimated wall distance, $y = y^+\mu/(\rho u_\tau)$, where y^+ is the dimensionless wall distance, μ is the dynamic viscosity and u_τ is the friction velocity which is estimated with the Schlichting skin friction correlation [150]. For CFD simulations using the SA turbulence model, y^+ is desired to be equal or lower than 1 [149]. In this thesis, we establish the optimal first-layer thickness of the grid by carefully considering the trade-off between wall distance calculations under each atmospheric condition included in the mission profile and varying Mach numbers.

Generally, the distance between the first layer and the last layer is progressively increased until no significant changes in results are observed. For the specific case study, the marching distance does not affect significantly the results since the energy generated near the aerofoil is dissipated before it reaches the far field for restrained values of the parameter, both for the subsonic regime and transonic regime.

Differently, the number of nodes in the off-wall direction has a great influence on the accuracy of the simulation. Moreover, to ensure grid convergence in both subsonic and transonic flows, it has been necessary to vary the number of coordinates used to define the aerofoil geometry. Within *pyHyp*, the mesh generation process initiates by extruding the volume mesh from the surface mesh, which is initially constructed based on the provided coordinates. Augmenting the number of nodes in the

streamwise direction and reducing the size spacing is crucial especially in the transonic regime, in order to capture the location of the impinging point of the lambda shock. Furthermore, refining the initial curve is beneficial for achieving convergence, not only in the transonic regime but also in the subsonic case. Therefore, the aerofoil pre-processing procedure explained in Section 3.2.1 has played a pivotal role in ensuring the success of the mesh convergence study.

With reference to the flight mission sketched in Figure 3.3, the mesh convergence study has been conducted for the take-off, climbing and cruise phases. Given the impracticality of conducting a comprehensive study for every individual flight condition within the dataset, multiple assessments are made to compute the aerodynamic coefficients of the NACA 0012 aerofoil at an AoA of 2°, focusing on the flowing three flight scenarios:

Table 3.2: Flight scenario for mesh convergence study

Mach number	Reynolds number	Temperature [K]	Density [ρ]	μ [kgm^{-1}s^{-1}]	u_τ [m/s]
0.45	1.4 x 10^7	288.15	1.225	1.79 x 10^{-5}	6.181
0.60	8.7 x 10^6	255.65	0.736	1.63 x 10^{-5}	6.603
0.75	5.1 x 10^6	233.50	0.4135	1.46 x 10^{-5}	7.961

The mesh resolution is iteratively coarsened with a constant ratio, r, of 2.7839. Given the multiple parameters for mesh generation and considering that the pre-processing code allows defining only the number of points rather than their spacing, the indicated grid ratio has been calculated as the effective grid ratio,

$$r_{effective} = \left(\frac{N_1}{N_2}\right)^{(1/D)}, \tag{3.4}$$

where N_i is the total number of grid points used for the grid and D is the characteristic dimension flow domain [151]. Following the Richardson Extrapolation technique (RE) [152], the discretisation error is estimated by extrapolating the solution of a grid with a fictitious grid spacing of the order of zero:

$$\psi_{exact} = \frac{r^s \psi_1 - \psi_2}{r^s - 1}, \tag{3.5}$$

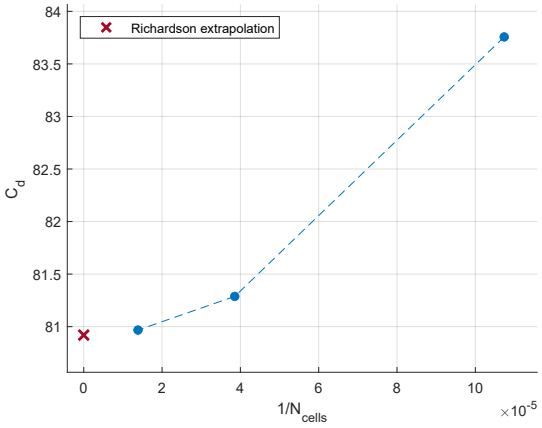
where ψ_i is the grid solution value of the i-th most refined grid, r is the grid refinement ratio and s is the rate convergence index:

$$s = \frac{\ln\left|\frac{\psi_3 - \psi_2}{\psi_2 - \psi_1}\right|}{\ln r}. \tag{3.6}$$

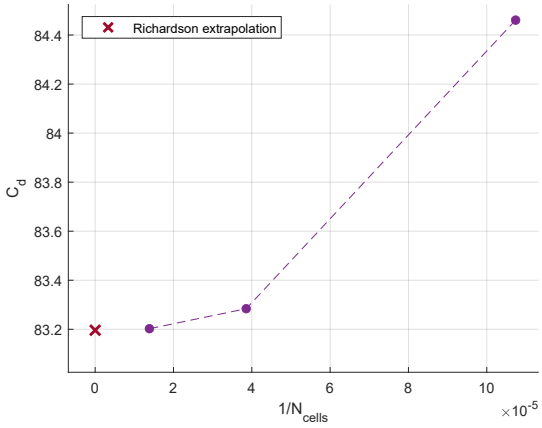
The results of the grid convergence study conducted at each flight condition are presented in Table 3.3 - 3.5 and Figures 3.5. For each Mach number, the drag coefficient C_d of the multiple configurations is compared with the Richardson extrapolation. The CPU time taken to generate the mesh and to perform the CFD simulation is measured from a workstation provided with 2 *Intel Core i9-11900* processors in parallel.

The study reveals that the values of C_d converge with the increasing number of cells respectively, for each flight condition, at:

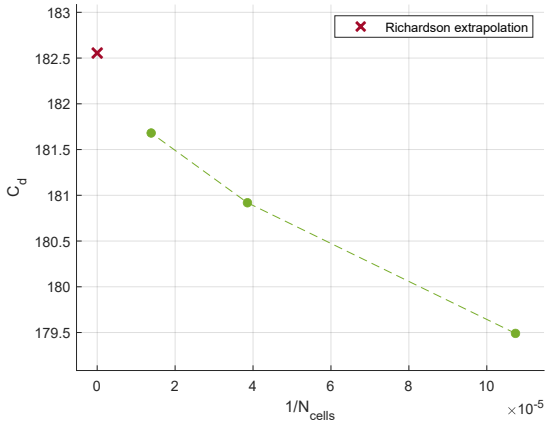
- a ratio of 1.9950 for $M = 0.45$;
- a ratio of 2.4492 for $M = 0.60$;
- a ratio of 0.6132 for $M = 0.75$.



(a) Mesh convergence study for $M = 0.45$, $Re = 1.4 \times 10^7$ with order 1.9950.



(b) Mesh convergence study for $M = 0.60$, $Re = 8.7 \times 10^6$ with order 2.4492.



(c) Mesh convergence study for $M = 0.75$, $Re = 5.1 \times 10^6$ with order 0.6132.

Figure 3.5: Mesh convergence study for subsonic and transonic regime.

Table 3.3: Mesh convergence study performed with NACA 0012 at an AoA of 2°, Mach 0.45 and an altitude of 0 m.

$N_{offwall}$	N_{strean}	Number of cells	C_l counts	C_d counts	Extrapolated relative C_d error	CPU time [s]
70	133	9,314	256.35	83.76	3.506%	2.50 x 10^2
129	201	25,929	255.90	81.29	0.455%	5.36 x 10^2
180	401	72,180	252.81	80.79	0.059%	1.49 x 10^3
		Richardson extrapolation	-	80.92	-	-

Moreover, as the accuracy improves, there is a corresponding increase in CPU time. In fact for each Mach number analysed, the computational time of the finest mesh is of one order of magnitude higher

Table 3.4: Mesh convergence study performed with NACA 0012 at an AoA of 2°, Mach 0.60 and an altitude of 5,000 m.

$N_{offwall}$	N_{stream}	Number of cells	C_l counts	C_d counts	Extrapolated relative C_d error	CPU time [s]
70	133	9,314	312.11	84.46	1.520%	1.34 x 10^2
129	201	25,929	311.15	83.28	0.105%	4.28 x 10^2
180	401	72,180	307.02	83.20	0.007%	1.44 x 10^3
		Richardson extrapolation	-	83.19	-	-

Table 3.5: Mesh convergence study performed with NACA 0012 at an AoA of 2°, Mach 0.75 and an altitude of 10,000 m.

$N_{offwall}$	N_{stream}	Number of cells	C_l counts	C_d counts	Extrapolated relative C_d error	CPU time [s]
70	133	9,314	362.30	179.49	-1.679%	1.05 x 10^2
129	201	25,929	369.55	180.91	-0.896%	3.45 x 10^2
180	401	72,180	371.84	181.68	-0.479%	1.57 x 10^3
		Richardson extrapolation	-	182.55	-	-

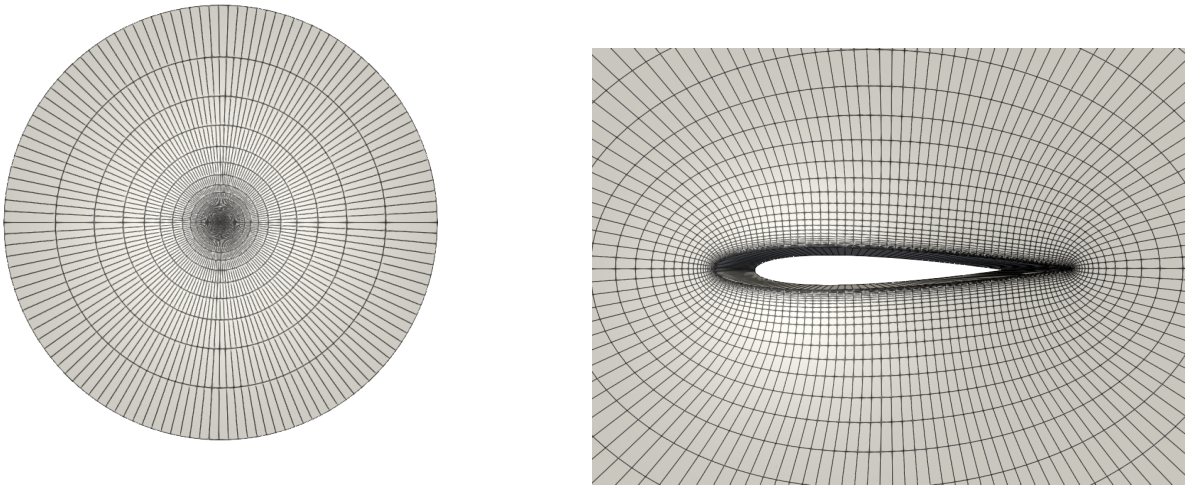
than the computational time of the two coarser meshes. Nevertheless, the discretisation error remains consistently low across all mesh configurations that are refined, with less than 2% relative C_d error in both subsonic and transonic regime simulated.

Although it is not ideal to use the same mesh configuration for such vast flight envelope and aerofoil shapes, as indicated by the lower convergence rate of the transonic case, defining the best grid for each aerofoil and each AoA, Reynolds number and Mach number would be intractable.

The decision to adopt a trade-off configuration for the mesh has been made considering several factors such as scalability of computational costs, the accuracy achieved in both coefficients across all meshes set up and the dataset size requirement of more than 140,000 simulations. Among all factors, the selection of the mesh was heavily influenced by the size of the training data. This decision aligns with data science principles outlined in Section 2.1 and the current state-of-the-art in data-driven ASO classified in Section 1.3.1, emphasising the necessity of constructing a comprehensive database to ensure the performance of the chosen SM. Hence the medium refined mesh achieves the balance between the accuracy requirements and computational costs across the varying Mach number. In conclusion, the standard mesh final parameters include a thickness of the first layer of 2.0 x 10^{-6} m, a distance between the aerofoil wall and the outermost edge of 100 aerofoil chords, a number of 128 nodes in the off-wall direction and a number of 201 nodes in the streamwise direction. Consequently, each aerofoil geometry included in the dataset is pre-processed setting a number of 201 coordinates.

Lastly, the chosen mesh configuration has been simulated for two different flight conditions distributed by the NASA website. The first case considers a NACA 0012[2] aerofoil, Mach number equal to 0.15, zero AoA, static temperature of 300 K and Reynolds number of 6 x 10^6. The second case simulates a RAE

[2]https://turbmodels.larc.nasa.gov/naca0012_val.html



(a) Mesh overview (b) Mesh detail

Figure 3.6: Two-dimensional overview (a) and a detailed close-up view (b) of the hyperbolic mesh generated using *pyHyp* [144]. The mesh is selected based on the convergence study of the NACA 0012 aerofoil

2822[3] aerofoil, Mach number equal to 0.729, AoA of 2.31 °, static temperature of 255.56 K and Reynolds number of 2.1 x 10^7. For both simulations, the extrapolated relative error of C_d with the experimental results is less than 3%, respectively, an error of 2.09% for the NACA 0012 in the subsonic regime and an error of 2.89% for the RAE 2822 in the transonic regime. While the relative error remains relatively high in the context of a purely aerodynamic study, these errors fall below the commonly accepted threshold of 5%, typically regarded as a benchmark for optimisation purposes. This comparison highlights that despite the standard mesh is not ideal for each aerofoil shape and flight condition, it serves as a well-balanced solution between accuracy and cost-effectiveness when creating a dataset.

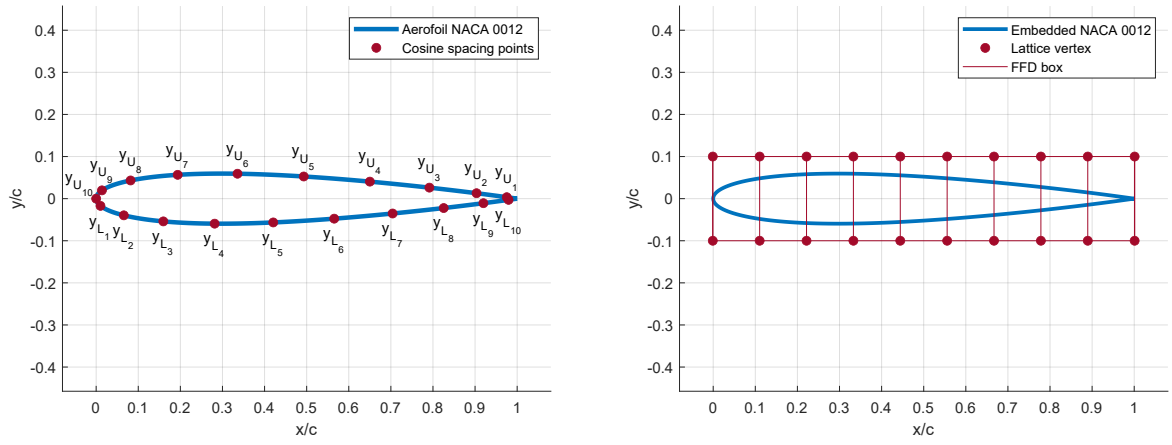
3.2.4 Aerofoil Geometry Sampling

The *prefoil* tool is a pySpline-based utility module that allows to flexibly handle aerofoil geometries. *prefoil* is an essential tool for the creation of the dataset since each aerofoil shape simulated needs to be properly represented in the dataset file. In accordance with the mesh convergence study, each aerofoil is pre-processed in 201 coordinates obtained using a cosine distribution strategy. Nevertheless, including all 201 coordinates in the dataset would significantly increase the complexity of the SM. Moreover, it could lead to challenges in terms of data management and potentially over-fitting for large datasets, without adding value to the analysis. Therefore it is necessary to define the minimum number of points for each aerofoil that better achieves the desired performance without losing essential information. However, after several attempts, the best balance choice for our application is an array containing 10 y-coordinates of the upper surface and 10 y-coordinates of the lower surface. The y-coordinates of each point are computed using a cosine distribution sampling because it effectively captures the features of the leading edge and trailing edge, as it was explained in Section 3.2.1. Given that each aerofoil is pre-processed in

[3] https://www.grc.nasa.gov/www/wind/valid/raetaf/raetaf.html

order to ensure a zero AoA, the y-coordinates for both the leading edge and the trailing edge are explicitly fixed at zero. Nevertheless, with reference to the existing dataset where the edges are discarded [59], this work has chosen to reintroduce the leading edge position. This decision was motivated by the fact that the stagnation point of an aerofoil is highly affected by the position of the leading edge, with a significant impact on the aerodynamic performance. Therefore, to enhance the representation power of the dataset and to expand the SM capabilities, including the leading edge yields to substantial benefits. Figure 3.7a presents an illustration of the sampled NACA 0012 aerofoil, which serves as a representative example within the dataset.

Lastly, once the aerofoil shape is simulated, both the y-coordinates of the geometry and the aerodynamic coefficient are stored row by row in a CSV file format. The overall flowchart of the parameterisation module is shown in Figure 3.8.



(a) Sampled aerofoil.

(b) Embendeed aerofoil inside FFD box.

Figure 3.7: Sampling output and FFD parameterisation of the NACA 0012 aerofoil.
a) 20-point sampled representation of the NACA 0012 aerofoil generated with the cosine distribution implemented in *prefoil*. b)Aerofoil embedded inside an FFD box created using *pyGeo* library. The FFD box consists of 10 evenly spaced points along the x-direction, with y-coordinates of 0.1 for the upper points and of -0.1 for the lower points.

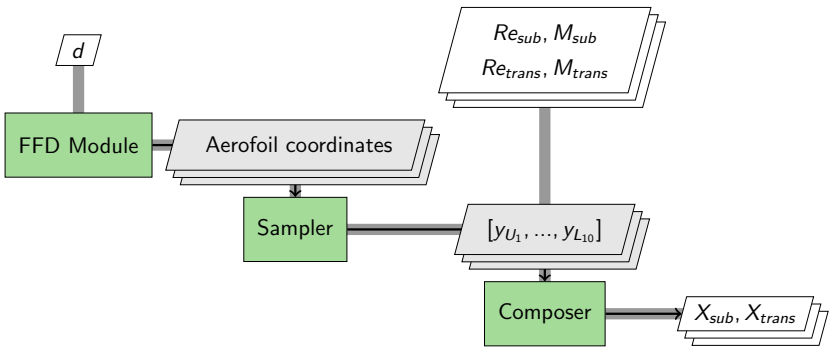


Figure 3.8: Geometry parameterisation Module: FFD Box update to assemble evaluated data for the SM. The process and data dependencies are illustrated via the Extended Design Structure Matrix [143]

3.3 Aerofoil Shape Parameterisation

As already mentioned in Section 3.1, the parametric method adopted is the FFD technique [67]. *pyGeo* is a package [153] tailored for shape manipulation, designed primarily for applications in aerodynamic and multidisciplinary optimisation (MDO). It is integrated within the MACH-Aero[4] framework and can serve both as a standalone tool and as support for high-fidelity MDO problems. *pyGeo* completely incorporates FFD methods, which simplifies the configuration of design variables and design constraints as it is detailed in Section 1.3.2. The library is favoured not only for its robust functionality but also for its user-friendly documentation and online tutorials. Moreover, its integration with other components of MACH-Aero makes it a coherent choice with the tools employed for the creation of the dataset.

The FFD method has been extensively used in the parametric modelling of two-dimensional and three-dimensional aerodynamic shapes. The main idea is embedding the entire reference geometry in a parameterised volume. Once embedded, high-level modifications made to the FFD lattice indirectly modify the included object. In *pyGeo*, B-spline volumes serve as the foundation for geometry manipulation. This approach enables the manipulation of complex geometries using a relatively small set of design variables, ensuring homogeneity in geometry control. To delve deeper, we recommend reading the article by Kenway et al. [142].

In reference to our application, an FFD box is generated with a dimension of 10 x variables, which are uniformly distributed along the chord of the aerofoil. Although *pyGeo* provides the option to implement a non-parallel configuration of the FFD box around the aerofoil, we choose to fulfil the traditional box configuration in accordance with the methodology established by Junior et al. in their work [83].

Since *pyGeo* is compatible only with a three-dimensional FFD box, the one generated around the morphing aerofoil comprises a total of 40 control points. As it was implemented for *pyHyp* a fictitious span of one unit is implemented. However, it is important to note that a code has been customised to guarantee that the displacements of the control points in one plane perpendicular to the aerofoil are mirrored in the other plane. This symmetry ensures precise and consistent control over the geometry in both planes, despite the inherent three-dimensional nature of the software. Hence, the individual control points, also referred to as local design variables in *pyGeo*, are moved to obtain local shape modifications of the baseline aerofoil. Due to the capability of *pyGeo* of controlling individually each control volume and the underlined approach of modifying the shapes with respect to the stored unperturbed baseline geometry, the aforementioned geometric constraints are imposed in this phase. Specifically, the vertical displacement of only four points closer to the leading edge and four points closer to the trailing edge, for a total of 8 points, is allowed. This selection of control points ensures the conservation of the structural integrity of the wing-box without affecting its internal geometry. Figure 3.7b includes a visualisation of FFD-box with the NACA 0012 aerofoil as embedded geometry. Figure 4.11 highlights the different degrees of freedom of each lattice vertex.

[4] https://github.com/mdolab/MACH-Aero/tree/main

3.4 Aerodynamic Coefficient Surrogate Modelling

The optimisation workflow developed to optimise a morphing aerofoil relies on the performance of the SM. Given the real-time constraints, as mentioned in Section 1.2, developing a surrogate model to adopt in lieu of the highly expensive physics-based simulations during the optimisation process is imperative. Due to the iterative nature of the process, the SM must be fast, efficient and accurate enough to guide the optimisation algorithm towards the optimal shape. Moreover, as explained in Chapter 1, one crucial factor of a surrogate-based ASO is developing and training a SM capable of understanding the underlying physics of the problem. The state-of-the-art research has widely explored the advantages of using a DL-based model to map an aerofoil shape to its corresponding aerodynamic coefficients.

Therefore, this thesis work aims to study and explore the potential of artificial neural networks present in the literature, with the goal of expanding their range of practical applications. As explained in Section 1.3.1, MLPs and CNNs have been shown to be the most suitable and flexible architectures to perform input-output mapping, without prior knowledge of the physical system. Despite the superiority of CNN over MLPs in terms of both efficiency and accuracy in visual tasks [58], MLPs demonstrate higher adaptability in understanding dataset structured around coordinate-based information. For example, Junior et al. [83] use two fully connected DNNs to approximate separately C_l and C_d. The input data of each neural network is indeed a vector composed of 20 elements that uniquely define the aerofoil shape and 3 neurons are respectively the Reynolds number, the Mach number and the AoA. The two trained networks perform an accurate prediction of both coefficients with an absolute mean error less than 2.35% for C_d, addressing the accuracy and cost requirements. Moin et al. [59] implements a simple ANN architecture, ensuring that the architecture learns the aerofoil geometric design space using normalised 2D coordinates instead of aerofoil design parameters. Representing each flight condition with the Reynolds number, Mach number and AoA makes the MLP suitable for optimising morphing aerofoils, as it excels in identifying patterns within a vast array of flight conditions. Du et al. [55] propose separate MLPs for predicting the aerodynamics in the range of $M = [0.3, 0.6]$ and $M = (0.6, 0.7]$, combined with a BSpline-GAN for the data-driven geometry parameterisation. In this case, Reynolds number, Mach number and AoA are given as input parameters, whereas the aerofoil coordinates are substituted by the BSplineGAN variables. Although the latter approach does not involve the direct representation of aerofoil coordinates, its performance underscores the adaptability of MLPs in learning from different data parameterisations.

In a prior publication by the same authors [154], there was a hypothesis presented suggesting that CNN could lead to better performance. However, after several researches in the field we have concluded that, while CNN is perfectly suitable for understanding large differences between geometric features within the dataset, it might not be the most recommended choice for the optimisation of a morphing aerofoil where the inherent displacements are relatively subtle. Consequently, we selected an MLP for the proposed optimisation framework, as it is well-suited for the outlined coordinate-based design space.

Hence, due to the remarkable performance of the architecture developed by Moin et al. [59] in the low-incompressible subsonic regime and its open-source availability, the author of this thesis tailors the MLP on a new dataset entirely built from scratch with a massive variety of aerofoil shapes and

a consistent variance in the flight range represented. The SM represents the aerofoil geometry with twenty vertical point coordinates, which are sampled using the procedure detailed in Section 3.2.4. In addition to the set of coordinates, the model takes also the Reynolds number, the Mach number and the AoA as input. Consequently, since the network will be used to compute simultaneously several aerofoils, the input to the MLP is a comprehensive matrix denoted as $\mathbf{X} \in \mathbb{R}^{m \times 23}$, where m is the dimension of the collection of shapes. The i-th input row is defined as follows:

$$\mathbf{X}^i = \Big[y_{U_1}, \quad \ldots, \quad y_{U_{10}}, \quad y_{L_1}, \quad \ldots, \quad y_{L_{10}}, \quad Re, \quad M, \quad \alpha \Big], \tag{3.7}$$

where $[y_{U_1}, \ldots, y_{L_{10}}]$ are the y-coordinates representing the i-th aerofoil, as plotted in Figure 3.7a.

This approach enables the network to capture the aerodynamic characteristics of the aerofoil, even with minimal geometric information. As it was done by the original developers, in this work the network has been tested for different setups of geometric representation of the aerofoil. It was observed that, for a fixed architecture, the performance of the neural network does not significantly improve with the increase of the input representation. We conclude therefore that parameterising the dataset is of great advantage for the MLP, which can compress the information in a lower-design space and effectively understand the peculiar features of the input.

Moreover, the developed SM performs simultaneous computations for all three aerodynamic coefficients. This stands in contrast to the conventional approach, where surrogate models typically predict one coefficient at a time. This traditional approach is not only time-consuming since it requires a double training time and hyper-parameter tuning, but it also disregards the potential interdependence among the output variables. In contrast, a multitask NN enhance a parallel multiple variables regression, making the training process more efficient. Given that Junior et al. [83] already conducted a morphing aerofoil optimisation using two parallel models, we opted to investigate the functionality of a single surrogate model for the same application.

Lastly, the SMs are conventionally trained for one regime at a time, a practice that is time-consuming and limited in terms of achieving generalisation. Recognising that this approach has been extensively explored in the literature [50, 51, 55, 83], we decided to use the same surrogate model for both transonic and subsonic regime. Our aim is to determine if it is possible to overstep the limitation associated with ANN in terms of regression and generalisation.

Architecture

The MLP used in this thesis differs from the original proposed by Moin et al. [59] in terms of both the width and depth of the hidden layer. As a matter of fact, it was necessary to conduct a study encompassing various configurations in order to identify the most suitable architecture for our specific dataset. Nevertheless, we decided to use the same optimisation scheme, layers type and sequential building approach as the original architecture. Moreover, the output configuration is conserved comprising the three aerodynamic coefficients, C_l, C_d and C_m, even though the moment coefficient is not exploited for the optimisation problem. Leveraging these similarities enables us to expedite a direct performance

comparison with the original model.

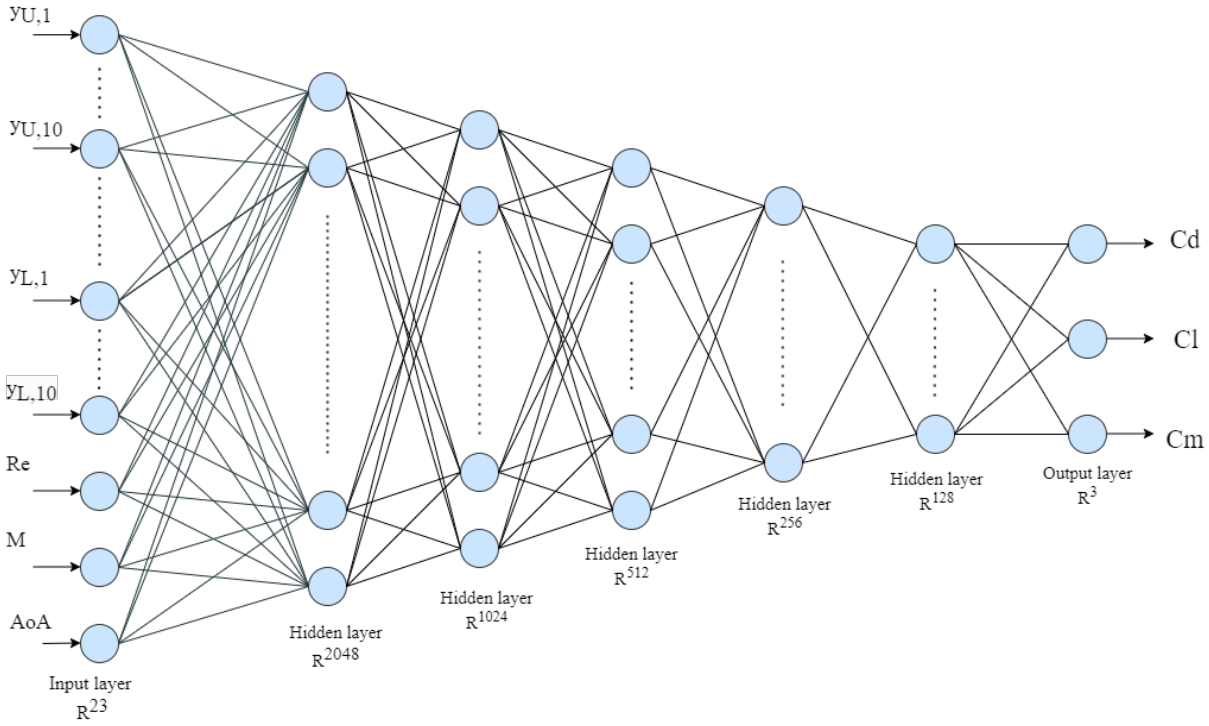


Figure 3.9: DL-based SM architecture resulting from the sensitivity analysis on width and depth.

The elements involved in building this network are presented as follows:

- **Dense Layers** are chosen as building blocks for the NN. These layers are characterised by full connectivity, meaning that each neuron in the layer is connected to each neuron in the previous and subsequent layers. The complete interconnection allows for the flow of information between all neurons, making the network capable of capturing complex relationships in the training data and approximating non-linear functions. Additionally, dense layers are suitable for extracting meaningful patterns and features from the input dataset, facilitating representation learning. Lastly, their intrinsic flexibility makes them a versatile choice for different architecture, allowing for the achievement of high levels of generalisations.

- **Rectified Linear Unit** is used as the activation function to incorporate non-linearity into the network. ReLU is applied to each layer due to its computational simplicity and its effectiveness in handling vanishing gradients.

- **Sequential approach** is employed to assemble the NN. The model consists of stucking layers sequentially, making it a straightforward way to build the neural network. The network is designed using Keras API in *Python*.

- **MSE** is kept as the loss function, whereas **RMSE** and **R-squared** are chosen as metrics to measure the performance of the networks. RMSE measures the average magnitude of the errors in

the predicted values, while R-squared statistically measures how the predicted values match the variance in the target data.

- **Adam optimiser** is the optimisation scheme used with a learning rate of 0.005. According to the original architecture, the first and second moments are maintained at 0.9 and 0.999 respectively

For more details regarding activation functions, optimisation schemes and loss functions we send back the reader to Chapter 2, where each topic is theoretically detailed.

In conclusion, this work proposes a DL-based model capable of mapping low-dimension geometric features of aerofoil into multiple aerodynamic coefficients. The peculiar difference with the existing networks built for aerodynamic surrogate modelling lies in the dataset training which covers both subsonic and transonic regimes and encompasses a diverse range of aerofoil shapes. The proposed solution seeks to expand the applicability of the neural network ensuring both robustness and efficiency to handle multiple flow conditions and representation schemes. This approach also aims to reduce training time while enhancing the network's ability to generalise and make accurate predictions.

3.5 Optimisation Scheme

To complete the optimisation framework, it is crucial to choose the optimisation scheme most suitable to navigate the design space leading towards the maximisation or minimisation of the objective function represented by the SM. The choice of the optimisation scheme is driven by a trade-off of exploitation and exploration. On the one hand, exploration is the act of exploring the environment to find out information about it. On the other hand, exploitation refers to the act of exploiting the already experienced environment and learning in order to maximise the return. Both components need to be fine-tuned to ensure that the optimisation algorithm reaches different promising regions of the design space and searches for the optimal solution within the given regions [38, 89]. Moreover, the choice of the optimisation scheme is also guided by the availability of the gradients of both objective and constraint functions. As discussed in Section 1.3.3, a gradient-based optimisation algorithm reveals more advantages for those applications where the gradients are easily obtained. Indeed, in estimating the direction of the optimisation process, gradient-based algorithms are more efficient and faster. In this study, considering the unavailability of the derivatives of the objective functions, a gradient-free optimisation scheme is the recommended scheme that best matches with the SM. However, it is noteworthy that gradient-free optimisation schemes tend to be more time-consuming and require a higher number of iterations to achieve convergence. Nevertheless, in the context of this work, where the design space is relatively small and the SM operates quickly, the drawbacks associated with these algorithms are mitigated.

For this reason, the GA [8] and the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [155] are the preferred algorithms for this study. The objective is to optimise a morphing aerofoil using both algorithms, facilitating an informed comparison of the results. This choice was necessitated by the fact that, to the best of the authors' knowledge, configuring the same optimisation scheme with the constraint of the wing box cannot be automatically achieved within commonly distributed open-source frameworks

such as MACH-Aero or SU2 [156]. As a result, a dual optimisation approach is employed to showcase the capabilities of the proposed optimisation framework. Both GA and NSGA-II are evolutionary algorithms and are defined as *population-based* methods since they start the optimisation with a set of design points (the population) rather than a single starting point [38]. Nevertheless, they differ in their specific applications and objectives.

GAs are mainly used for single-objective optimisation (SOO) tasks. In these scenarios, the fitness of an individual element within the current population is based on its performance concerning the single objective function passed to the algorithm. Consequently, new candidate solutions are generated with techniques such as crossover and mutation. Hence, GAs are designed to converge towards a single solution for a given objective function, which results in an evolutionary process for the population over time. This characteristic makes GAs well-suited for iteratively exploring the design space.

NSGA-II is specifically designed for multi-objective optimisation (MOO) problems, in which multiple conflicting objective functions need to be optimised simultaneously. Indeed, NSGA-II does not converge to a single optimal solution but it finds a set of solutions that represent the best trade-off between the conflicting objectives, forming the so-called Pareto front [157]. In MOO a set of non-dominated solutions are derived, meaning within this set there are no other solutions that outperform them in all objectives while maintaining the same level of performance in one objective. The Pareto front is therefore the collection of those trade-off solutions where improving one objective function requires sacrificing the other. Hence, upon completing the optimisation process, these algorithms provide information regarding the Pareto dominance and crowding distance. The information guides the user in the subsequent decision-making process. While in NSGA-II the evolutionary process of the population is still made with crossover and mutation techniques, the algorithm employs also the elitism strategy. It aims to maintain the best solutions within the current generation found so far to preserve diversity in the next generation and prevent premature convergence. Due to the increasing complexity of the algorithm, NSGA-II is more computationally expensive.

In this thesis work, The NSGA-II is used to find the set of solutions that optimise the morphing aerofoil throughout its profile mission. This results in a collection of distinct shapes that the aerofoil can assume to minimise drag under specified flight conditions while ensuring the prescribed lift constraints. Instead, the GA is used to identify the singular configuration that optimally enhances the aerofoil performance across the total mission profile. Lastly, it is important to note that both GA and NSGA-II are most suited to unconstrained optimisation problems. However, it is possible to impose constraints by adding a penalty function to each objective function, extending therefore the capability of the optimisation scheme to handle a constrained problem [158, 159]. The penalty functions can be applied in an additive or multiplicative manner, with the former being more prevalent. Due to the real-time computation of the aerodynamic coefficient, this work has computed a dynamic penalty function for infeasible solutions. Indeed, the lift coefficient of each individual within the population is computed and compared with the prescribed lift constraint, creating an array of penalties. The penalty values are therefore added to the drag coefficients, allowing the optimisation schemes to incorporate the lift constraint into the optimisation process. The mathematical formulation of each penalty function is outlined in Section 4.2.1.

Chapter 4

Results

This chapter offers a comprehensive analysis of the results obtained from the proposed optimisation framework within the context of ASO, highlighting its advantages and disadvantages. In Section 4.1 a complete overview of the development of the DL-model is provided, including an in-depth analysis of the collected training dataset, as discussed in Section 4.1.1. In Section 4.1.2 a complete comparison of different architectures is outlined, ultimately leading to the training of the proposed SM and an evaluation of its performance. In the final part of this chapter, Section 4.2 assesses the performance of the SM in the context of ASO of a baseline aerofoil in morphing configuration. This includes the definition of the final proposed framework and a validation framework, using different optimisation schemes. Finally, the results obtained for the drag minimisation of a lifting surface in a constrained optimisation procedure are detailed in Section 4.2.1.

4.1 Aerodynamic Deep Learning Model Development

4.1.1 Dataset Splitting

After completing the preprocessing procedure detailed in Section 3.2.1, from over 1,625 aerofoils sourced from the UIUC library and NASA SC(2) library, a total of 1,284 geometries are assessed as suitable for training and validating the aerodynamic coefficients surrogate model. The remaining geometries are discarded:

- Missing data caused failure in the preprocessing stage [51];
- The shape does not respect the requirement of maximum thickness of $(t/c)_{max} = 0.25$ prescribed. Indeed. the thickness constraint is imposed to limit the design space while maintaining a wide range of shape possibilities. The design space, obtained as the envelope of the 1,284 shapes covered within the dataset, is depicted in Figure 4.1;
- The CFD simulations failed to converge, indicating that the shape may not be suitable for the specified flight conditions. As explained in Section 3.2.2, a 5-th order decay in total residuals is imposed

as a converge criterion. However, due to the broad range of shapes, achieving the 5-th order convergence may not always be guaranteed by *ADflow*, which is configured to keep simulating the shape until convergence or until the maximum number of iterations is reached. Therefore, during the building of the dataset, the total energy residual, density residual and turbulence residual are assessed simulation by simulation. As a result, any samples that do not meet the prescribed convergence threshold are eliminated.

As stated in Chapter 3, the commitment of this work is exploring the applicability of the existing DL model for the prediction of aerodynamic performance. Therefore a vast range of shapes are considered, increasing the variety of geometric features included in the dataset. The aforementioned assumption regarding the shapes is, in this phase, established to mitigate the variance across the samples and optimise the performance of the surrogate model.

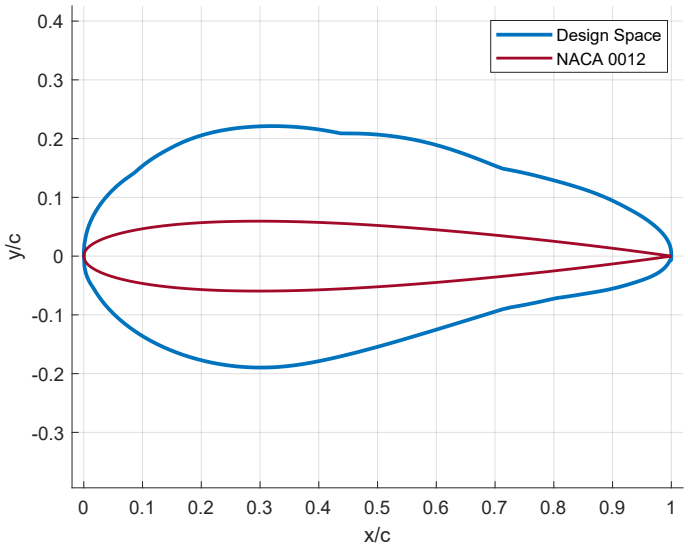


Figure 4.1: Design space covered within the dataset. The plot includes also the NACA 0012 aerofoil used as the baseline for the optimisation process.

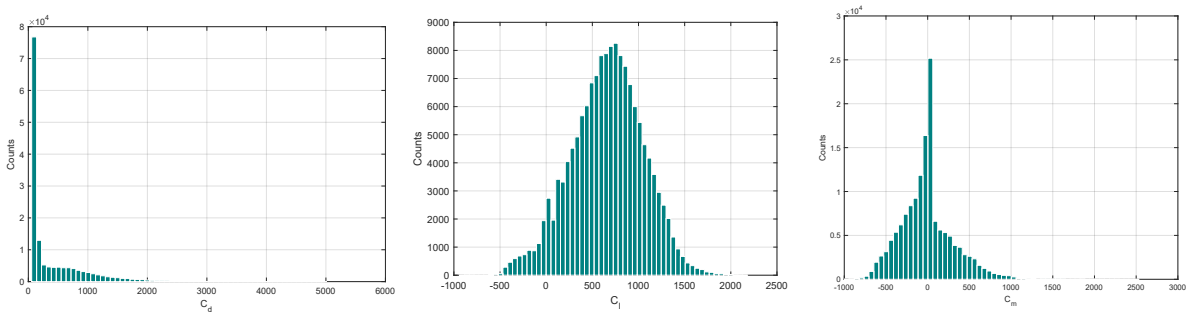
In conclusion, CFD simulations were conducted on each geometry for each flight condition scheduled in Table 3.2. This involves simulating each Reynolds number for every Mach number, which, in turn, are simulated for AoAs included in the range of 0° - 6°. Before training the surrogate model, the raw dataset undergoes thorough reformatting, cleaning, and final postprocessing phase. This step is crucial as it serves to eliminate any duplicates in the dataset. Additionally, the samples are randomly shuffled in order to improve the overall model quality and enhance its predictive performance. By doing so, the dataset attains a higher level of uniformity, with the removal of sparse samples that could potentially interfere with the training process. As a result, we gathered a dataset comprising 148,345 samples. All the simulations are conducted using an *Intel Core i9-11900* processor, configured in a parallel multi-processor setup. With an average CPU time of 43s for each simulation, when accounting for the discarded samples, the computation of the overall dataset took more than 73 CPU days.

To ensure an even representation of the subsonic and transonic regime, the distribution of Mach numbers in the dataset aligns with the percentages outlined in Table 4.1. The relatively higher occur-

rence of Mach numbers below 0.55 is a result of the difficulties encountered by the CFD simulator to attain convergence within the transonic regime. Figure 4.2 depicts the distributions of the obtained aerodynamic coefficients, showing the higher concentration of data points below 1000 C_d counts. The C_m coefficient shows a peak concentration around 0 counts, whereas C_l demonstrates a distribution that approximates a Gaussian curve.

Table 4.1: Distribution of Mach numbers in the dataset.

Mach number	Samples	Percentage of the total
0.35	26,045	17.56%
0.45	26,096	17.59%
0.55	26,266	17.71%
0.65	23,186	15.63%
0.75	23,439	15.80%
0.85	23,313	15.72%
Total samples	148,345	



(a) C_d distribution. (b) C_l distribution. (c) C_m distribution.

Figure 4.2: Distribution of aerodynamic coefficients, C_d (a), C_l (b), and C_m (c), in the database under subsonic and transonic flight conditions simulated.

Furthermore, a fraction of the data is reserved for testing, to safeguard the model against overfitting the training data and enable it to learn the underlying functions. Indeed the model hyperparameters are fine-tuned by evaluating the performance of the trained MLP on the test dataset. In this way, we have enhanced the generalisation capability of the architecture on unseen samples. Moreover, the training of the model is monitored with validation data to determine the best time to stop training using early stop techniques. The learning algorithm during the training has access to both the training and the validation sets. The former is used to estimate the gradients for updating the network parameters, while the latter is used to evaluate model performance on unseen data during the training phase. According to the approach of Moin et al. [59], the dataset is split into training, validation, and test sets with a ratio of 70:15:15, comprising a total of 10,842, 22,251, and 22,251 samples respectively. The partition of the dataset is made to ensure that each sample point is represented only in one set, avoiding the overfitting

issue.

In this work, the splitting of the dataset is based on flight conditions rather than aerofoil, as the primary objective is to enable the mapping of the aerodynamic coefficients across different flight regimes. To evaluate the performance of the model on handling variation in unseen geometry shapes, we have stored three simulated aerofoils which are not included in any sets. The comparison results, between the predicted coefficients generated by the model and the coefficient computed through CFD simulations, are presented in Section 4.1.2.

4.1.2 Surrogate Model Training

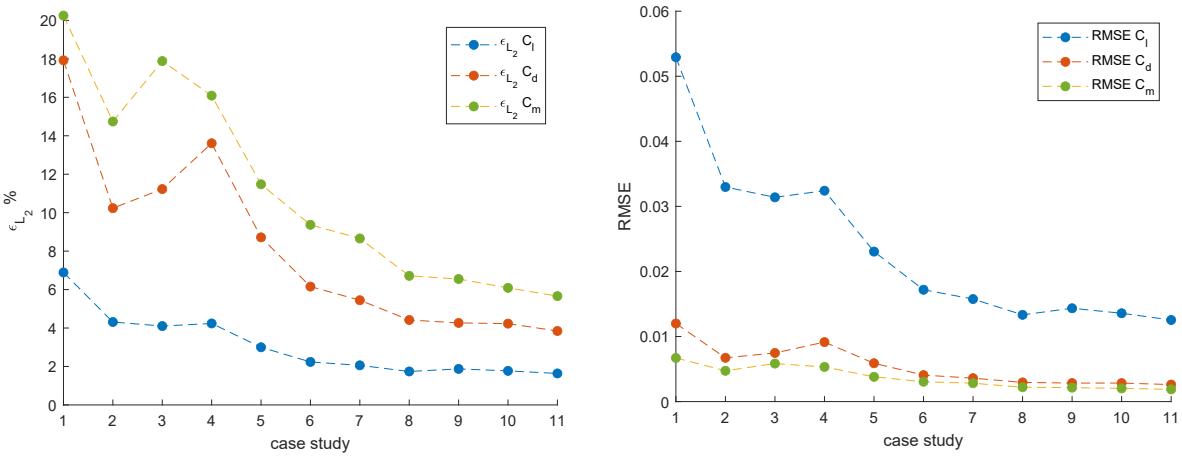
According to the original model, the loss function chosen is the MSE. Given the use of high-fidelity simulations within this task, instances of outliers are minimal. Consequently, MSE is a well-suited choice, optimising the convergence process for regression-based learning. Additionally, the batch normalisation technique is implemented as the regularisation method to prevent the MLP from overfitting the training data. The introduction of batch normalisation is essential in improving the convergence of the model since the activations of intermediate layers are redistributed during the training process due to the change in network parameters. This phenomenon also referred to as internal covariate shift, leads to potential instability due to the difficulty encountered by the internal layer to learn from the previous one [124]. As mentioned in Section 2.1.2, gradient descent stands out as the prevailing training algorithm used in deep learning models. Nevertheless, this type of algorithm can be susceptible to the complexity and the challenges associated with the learning process. Therefore, implementing a batch normalisation technique results in beneficial mitigation of internal covariate shift due to the normalisation of the activations of each layer within a mini-batch during the training.

To further enhance the architecture and training procedure of the SM, a hyperparameter tuning approach is implemented. This process has become imperative because the guidelines derived from the initial model may not guarantee the same level of performance when applied to our specific dataset. It is important to notice that we decided to conserve the Adam algorithm as the learning optimiser. To the best of the author's knowledge, adopting the Adam optimiser for training a regression MLP is a common choice since it adapts the learning rate for each activation individually, which is especially efficient when dealing with complex and high-dimensional data [93]. The dual advantages of Adam optimiser, namely its adaptability of learning rate and low memory requirements, make it well-suited for efficiently training large neural networks within a reasonable time frame. Hence, also aimed to maintain continuity with state-of-the-art models, we assumed that it would perform successfully for our application.

Fixing the learning algorithm with a learning rate of 0.0005, multiple configurations are explored to identify the optimal model with superior generalisation capabilities on our dataset. Hence, systematic parameter studies and experiments are conducted to assess the influence of specific hyperparameters on the model performance. The parameters that are tuned include the width and the depth of the architecture, referred to as the number of hidden layers and the size of each layer, the batch size, and the number of epochs used for training. Each study involves varying one hyperparameter while keeping the

other constant and analysing its effect on the model accuracy and convergence rate. To ensure the reproducibility of the results and mitigate errors stemming from dataset shuffling and random variance, the experiments are conducted using the same full dataset and the same random seed [42, 30]. Moreover, each setup is iterated five times to ensure the robustness and generalisability of the outcomes. Each model is trained using parallel contributions of the training set and the validation set, as explained in Section 4.1.1, with the subsequent evaluation on the test dataset. The model that performs the best is chosen based on evaluation metrics, including the loss function and computational efficiency. In addition to the evaluation metrics used by Moin et al. [59], this thesis has post-processed the outcomes of the tuning process also comparing the dispersion of the relative $\mathcal{L}^2$ error, outlined in Equation 1.5. To gain a more comprehensive insight into the performance, individual evaluation metrics are computed for each coefficient C_l, C_d, and C_m. This approach allows us to independently assess their performance, leading to a more refined analysis of their respective outcomes.

Firstly, a study is conducted to determine the complexity of the MLP that best exploits the simulateddesign space and finds the mapping function between the geometry samples and the aerodynamic coefficients. The study explores the depth and the width of the network by varying the number and the length of each hidden layer. Figure 4.3 illustrates the progression of the metrics concerning the case study. Table 4.2 provides a comprehensive analysis of different architectures, comparing the RMSE loss function, the relative $\mathcal{L}^2$ error, and the overall number of hyperparameters. Each layer undergoes batch normalisation, with the activation function set as Leaky ReLU. All models are trained using early stopping with a patience of 50 epochs.



(a) $\epsilon_{\mathcal{L}^2}$. (b) RMSE.

Figure 4.3: Evaluation metrics trends for each aerodynamic coefficient C_l, C_d and C_m with the increasing complexity of the network architecture: a) ϵ_{L_2} plotted in percentage values and b) RMSE.

The study unveils a noteworthy trend: as the number of layers increases, there is a consistent decrease in both the relative $\mathcal{L}^2$ error and RMSE for each modelled coefficient. Furthermore, having a uniform architecture with layers decreasing in the number of neurons, especially with smaller layers

Table 4.2: Test set accuracy and number of learnable parameters for different setups of the MLP.

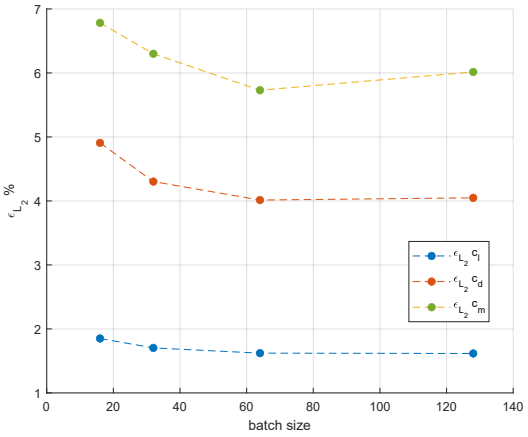
Case	Network	$\epsilon_{\mathcal{L}^2}$ [%]			RMSE [‰]			Number of
study	architecture	C_l	C_d	C_m	C_l	C_d	C_m	parameters
1	64, 3	6.8912	17.924	20.251	52.925	11.997	6.708	1,731
2	64, 32, 3	4.3130	10.233	14.743	32.983	6.729	4.735	3,713
3	64, 32, 16, 3	4.1022	11.228	17.886	31.394	7.471	5.843	4,195
4	64, 32, 16, 8, 3	4.2382	13.609	16.087	32.412	9.144	5.330	4,307
5	128, 64, 32, 3	2.9992	8.7166	11.477	23.053	5.894	3.805	13,507
6	256, 128, 64, 3	2.2361	6.1561	9.3687	17.189	4.069	3.052	47,491
7	512, 256, 128, 3	2.0591	5.4463	8.6578	15.758	3.602	2.825	176,899
8	1024, 512, 256, 3	1.7405	4.4180	6.7143	13.333	2.945	2.205	681,975
9	2048, 1024, 512, 3	1.8735	4.2657	6.5492	14.342	2.847	2.140	1,615,335
10	2048, 1024, 512, 256, 3	1.7718	4.2295	6.0890	13.575	2.845	2.032	2,673,667
11	2048, 1024, 512, 256, 128, 3	1.6381	3.8470	5.6609	12.540	2.614	1.882	2,804,227

closer to the output is more effective for many reasons. For instance, the hierarchical feature extraction is facilitated with the initial layers that capture high-level features while subsequent layers refine and extract more details. Additionally, smaller layers close to the output layer reduce the number of model parameters, preventing overfitting and enhancing better generalisation. Lastly, during the training, gradients tend to propagate smoothly through the network, resulting in faster convergence and stability [52]. Consequently, with respect to the metrics trends, the architecture characterised by its higher complexity, [2048, 1024, 512, 256, 128, 3], stands out as the selected option. Despite its complexity, this architecture efficiently accommodates the allocated computational budget for training and evaluation, all while exerting minimal impact on the development of the surrogate model. Once the architecture of the MLP had been settled, we proceeded with fine-tuning the batch size and the number of training epochs as stopping criteria. Table 4.3 presents the evaluation metrics derived from training sessions conducted with varying batch sizes. In this case, the training procedure is set up on an early stopping with a patience of 50 epochs. Considering that the performance results are relatively similar across all case studies, as also illustrated in Figure 4.4, it is worth noting that training with a batch size of 64 yields a slight improvement in the relative $\mathcal{L}^2$ error and RMSE. However, when factoring in the higher training time required for this batch size and the marginal difference in performance compared to the 128 batch size, this work opts for configuring the architecture with the larger batch size that was tested.

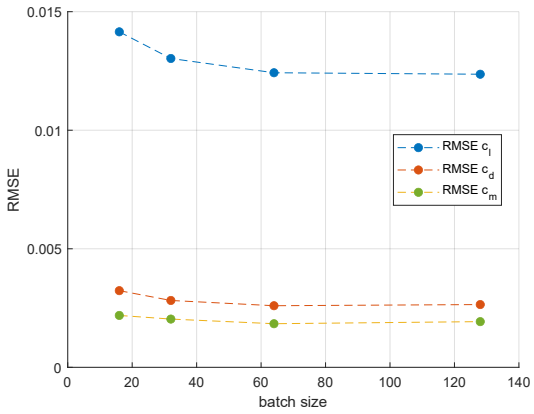
Table 4.3: Test set accuracy and number of learnable parameters for different setups of batch size.

Case	Batch	$\epsilon_{\mathcal{L}^2}$ [%]			RMSE [‰]			Training
study	size	C_l	C_d	C_m	C_l	C_d	C_m	CPU time (s)
1	16	1.8506	4.9075	6.7828	14.149	3.233	2.188	2.62 x 10^4
2	32	1.7034	4.3024	6.3001	13.026	2.820	2.035	2.45 x 10^4
3	64	1.6215	4.0137	5.7300	12.426	2.598	1.839	1.95 x 10^4
4	128	1.6163	4.0484	6.0153	12.361	2.647	1.927	0.78 x 10^4

In conclusion, an investigation into the impact of varying the number of training epochs was conducted. The accuracy, assessed in terms of relative $\mathcal{L}^2$ error and RMSE, showed only marginal differ-



(a) $\epsilon_{\mathcal{L}^2}$.



(b) RMSE.

Figure 4.4: Evaluation metrics trends for each aerodynamic coefficient C_l, C_d and C_m with the increasing batch size: a) ϵ_{L_2} plotted in percentage values and b) RMSE.

ences with increasing epoch thresholds, ultimately reaching its peak performance at 50 epochs as it is highlighted in Table 4.4. Moreover, the choice of 50 is also motivated to align with the computational constraints imposed, taking into account both the R^2 score as a measure of accuracy and the MSE loss function computed on the training and the validation sets.

Table 4.4: Test set accuracy and number of learnable parameters for different training epochs.

Case study	Epochs	$\epsilon_{\mathcal{L}^2}$ [%]			RMSE [‰]		
		C_l	C_d	C_m	C_l	C_d	C_m
1	10	2.0694	4.9964	7.6050	20.694	49.964	76.050
2	50	1.6381	3.8470	5.6609	12.540	2.614	1.882
3	90	1.7760	4.2938	6.6121	13.592	2.805	2.118

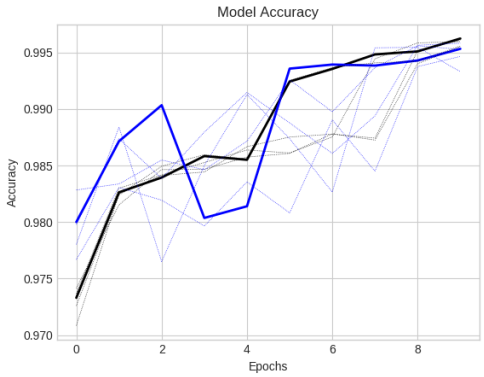
Furthermore, as depicted in Figure 4.5, it is evident that, except for the configuration with 10 epochs, each case study converged to stop training at around 30 epochs, indicating that extending the number of epochs and runs did not significantly alter the resulting performance of the MLP. Indeed, with the increase in the number of epochs, the accuracy reached a consistent asymptotic value of $R^2 = 0.9978$ for training and $R^2 = 0.9972$ for validation. This behaviour is also mirrored in the trends of the MSE. To assess the performance of the regression model on previously unseen data, the R^2 score is employed as a key metric. R^2 quantifies the proportion of variance in the dependent variable (the variables we want to predict) that can be explained by the independent variables (the variables used to make predictions).

The R^2 formula is expressed as follows:

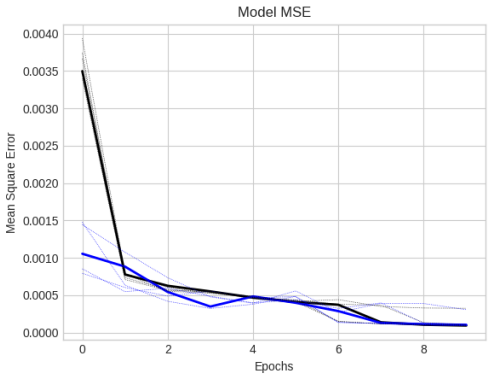
$$R^2 = 1 - \frac{\sum_{i=1}^{N}(\widetilde{\mathrm{y}}_i - \mathrm{y}_i)^2}{\sum_{i=1}^{N}(\widetilde{\mathrm{y}}_i - \bar{\mathrm{y}})^2}. \tag{4.1}$$

$\widetilde{\mathrm{y}}_i$ represents the predicted coefficient for the i-th sample as estimated by the SM. y_i represents the

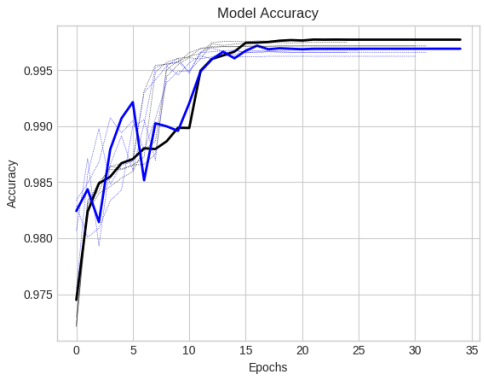
actual coefficient for the same i-th sample as computed by the CFD. In this context, $\bar{y}$ represents the mean value of the actual coefficients y_i across the dataset $\bar{y} = \frac{\sum_{i=1}^{N} y_i}{N}$. Values of R^2 close to 1 indicate a strong relationship between the variables and a good fit between the model and the data, suggesting the accuracy and robustness of the regression model.



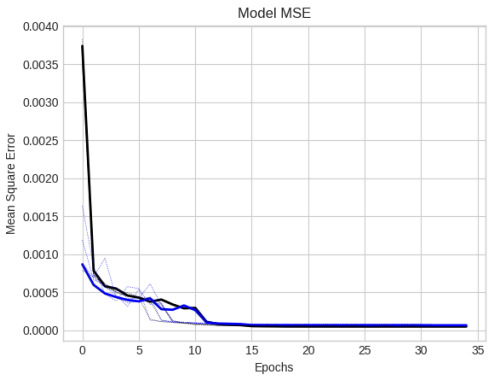
(a) Accuracy for 10 epochs.



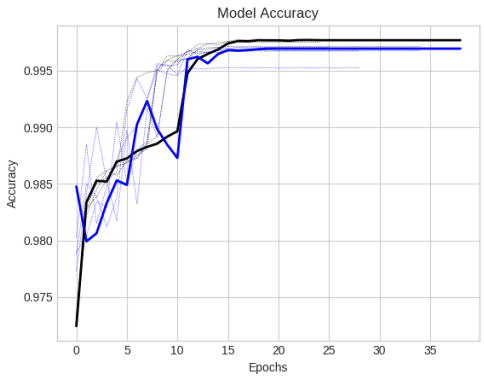
(b) MSE losses for 10 epochs.



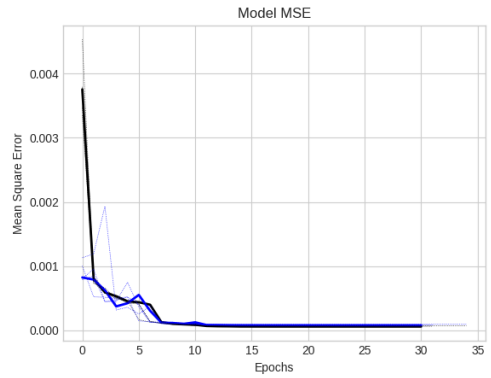
(c) Accuracy for 50 epochs.



(d) MSE losses for 50 epochs.



(e) Accuracy for 90 epochs.



(f) MSE losses for 90 epochs.

Figure 4.5: Model accuracy and Model MSE across increasing training epochs. The black line represents the training set, while the blue line corresponds to the validation set.

Bearing in mind the outcomes of previous studies, the tailored MLP used for the aerodynamic coefficients prediction is characterised by the final architecture depicted in Figure 3.9. This remaining section provides a more detailed analysis of the accuracy of the proposed network. Table 4.5 presents an

overview of the results obtained with the final SM, along with a comparison with state-of-the-art results from Table 1.1.

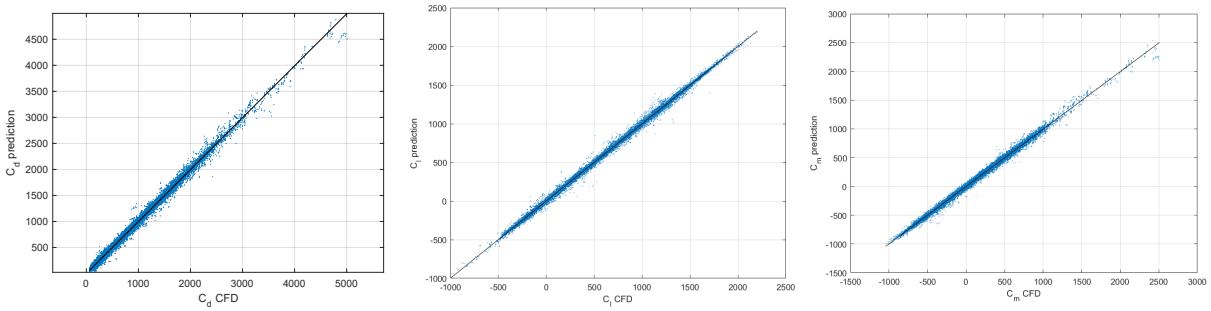
Table 4.5: Aerodynamic coefficient modeling results on the test dataset, benchmarking against state-of-the-art findings in interactive aerofoil shape optimisation. This table serves as an extension to Table 1.1.

Study	Model	Application	Flight Regime	Training Samples	Modelling Variables	Coefficients	$\epsilon_{\mathcal{L}^2}$ [%]	RMSE [‰]
Li et al. [51]	GE-KPLS	Aerofoil	Subsonic	81,000	16	C_d, C_l	0.26, 0.15	-
			Transonic	32,400	10		0.83, 0.40	-
Li et al. [50]	MLP	Wing	Transonic	135,108	57	C_d, C_l , C_m	0.35, 0.20, 0.36	-
Zhang et al. [58]	CNN	Aerofoil	Transonic	1,600	2,403	C_l	-	70.71
Du et al. [55]	MLP	Aerofoil	Subsonic	45,696	29	C_d, C_l	2.34, 2.26	12.90, 2.77
			Transonic	39,505	29		2.87, 4.65	16.13, 8.76
Moin et al. [59]	MLP	Aerofoil	Subsonic	454,675	23	C_d, C_l, C_m	2.17, 1.74, 1.64	7.88, 14.20, 4.21
Proposed	MLP	Aerofoil	Subsonic and Transonic	147,345	23	C_d, C_l, C_m	3.84, 1.64, 5.66	2.61, 12.54, 1.88

The performance are computed with reference to the evaluation test set obtained from the splitting of the entire database. At first glance, the proposed model presents results on par with the state-of-the-art, achieving relative $\mathcal{L}^2$ errors of 5.66% for the C_m and below 3.84 % for C_l and C_d. This suggests that the model effectively exploits the design space and maps the lift and the drag coefficients while respecting the generalisation requirements imposed. Nevertheless, a substantial drop in performance is noted in the mapping of the third coefficient, suggesting that the MLP trained on the new dataset may not fully capture all the relevant features of this specific parameter. Regarding the RMSE in the prediction of C_l, C_d and C_m the proposed model has a lower error than the state-of-the-art model. This suggests that, within the flight regimes of interest and considering the variability in the design training space, the proposed model offers a more accurate approximation of the aerodynamic functions, resulting in significantly reduced relative errors.

In this work, the accuracy of the model on previously unseen samples is evaluated by visually comparing its predictions with the test data. The study is therefore presented in Figure 4.6.

The accuracy and robustness requirements are assessed by comparing evaluation metrics and loss functions across multiple network setups. In the remaining section, a detailed analysis of the performance of the proposed model when applied to unseen aerofoil geometry is provided. The first objective of this work is to examine the state-of-the-art existing DL-based surrogate model in order to accurately predict the aerodynamic performance of both best and worst-performing aerofoils. The current approach in the literature often restricts the training stage to a reduced design space, focusing on the most feasible aerofoils. While this approach is effective in guiding the optimisation process toward well-performing regions, it becomes less convenient when the need arises to apply the model to aerofoils with features different from those found within the uniform design space. For this reason, this thesis work has created a dataset encompassing over 1,200 distinct aerofoil shapes. However, it is important to note that the partitioning of the training, validation, and test sets is not based on the individual aerofoil shapes.



(a) C_d regression with $R^2 = 0.998967$. (b) C_l regression with $R^2 = 0.997256$. (c) C_m regression with $R^2 = 0.996257$.

Figure 4.6: Regression analysis results for the aerodynamic coefficients, C_d (a), C_l (b), and C_m (c). The black line denotes the ideal regression model, where the predicted values match the CFD-computed values perfectly.

Instead, the partitioning is conducted with a focus on flight regimes, ensuring an equitable distribution of aerofoil features within each set. Nevertheless, given that the ultimate application involves optimising a morphing aerofoil, it is crucial to evaluate the model performance on previously unseen geometries. This evaluation is essential for gaining insights into how well the model can capture the non-linear coefficients for shapes that have never been simulated before. Figure 4.7 provides a detailed analysis of the prediction of aerofoil coefficients for NACA 0012, RAE 2822 and SC(2) 1095 aerofoils at varying AoAs. Given that each aerofoil in the dataset is simulated for 126 different flight conditions, for visualisation purposes Figure 4.7 presents the predictions for three different combinations of Reynolds and Mach number, accordingly with the flight regimes parameterisation conducted in Section 3.2.2. Therefore, in the figure are plotted respectively:

- Top: evaluation of NACA 0012 for $Re = 1.4 \times 10^7$ and $M = 0.45$
- Middle: evaluation of RAE 2822 for $Re = 8.7 \times 10^6$ and $M = 0.65$
- Bottom: evaluation of SC(2) 1095 for $Re = 5.1 \times 10^6$ and $M = 0.85$

A second phase of performance evaluation is outlined in Table 4.6, categorising absolute errors between the high-fidelity values and the predicted values for each aerofoil shape. The absolute error $\mathcal{L}^0$ is calculated as follows:

$$\epsilon_{\mathcal{L}^0}^{(i)} = |\widetilde{\mathrm{y}}_i - \mathrm{y}_i|, \tag{4.2}$$

where $\widetilde{\mathrm{y}}_i$ represents the SM-predicted coefficient for the i-th sample, and y_i represents the CFD-simulated coefficient for the same i-th sample. In this context, an average $\mathcal{L}^0$ error is computed for each Mach number, while keeping the Reynolds number constant and varying AoAs. The rationale behind selecting the $\mathcal{L}^0$ error lies in its suitability for scenarios with a limited number of data points involved in the evaluation.

Upon comparing the $\mathcal{L}^0$ distribution between the Mach number and coefficient plots for varying AoAs, it becomes evident that the DL-based SM accurately predicts the point-wise distribution. On the one

Table 4.6: $\mathcal{L}^0$ distribution across different Mach numbers, considering parameterised Reynolds numbers. The averages are computed while varying the AoA within a range of 0° to 6° in 1° increments. It is important to note that the error is presented as a percentage value.

Mach number	NACA 0012 - $Re = 1.4 \times 10^7$			RAE 2822 - $Re = 8.7 \times 10^6$			SC(2) 1095 - $Re = 5.1 \times 10^6$		
	C_l	C_d	C_m	C_l	C_d	C_m	C_l	C_d	C_m
0.35	0.16%	0.02%	0.05%	0.20%	0.05%	0.11%	0.30%	0.06%	0.06%
0.45	0.33%	0.05%	0.05%	0.38%	0.05%	0.08%	0.38%	0.08%	0.05%
0.55	0.91%	0.11%	0.10%	0.34%	0.12%	0.11%	0.72%	0.10%	0.07%
0.65	0.67%	0.18%	0.10%	0.47%	0.12%	0.08%	0.90%	0.18%	0.15%
0.75	0.30%	0.18%	0.12%	0.43%	0.10%	0.09%	0.79%	0.15%	0.17%
0.85	0.57%	0.34%	0.27%	0.84%	0.43%	0.17%	1.45%	0.10%	0.15%

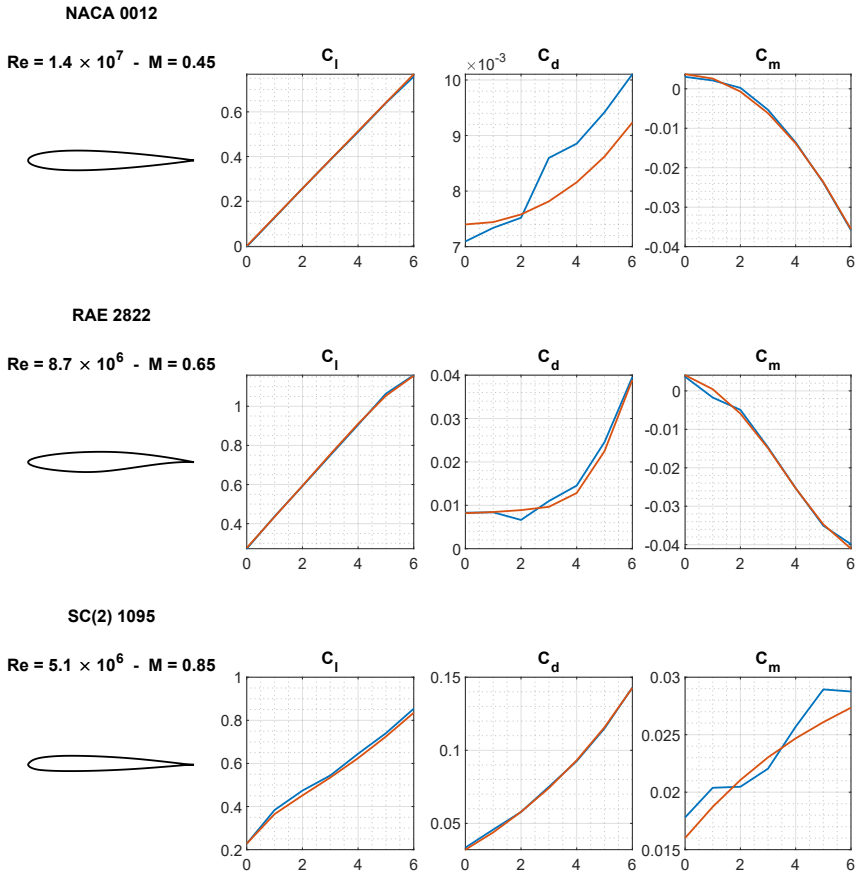


Figure 4.7: Comparison of evaluated aerodynamic coefficients (blue line) with the values obtained via CFD simulations (orange line) at varying AoA. Each sub-plot line is referred to the aerofoil geometry plotted on the left-hand side and the reference flight conditions are reported above the aerofoil shape.

hand, the results show that the SM performs well in predicting the C_l coefficient, as the SM curve closely matches the CFD curve. This level of accuracy is consistent across all the simulated Reynolds and Mach numbers, with an average error rate of less than 2% for each shape tested. The distribution of average $\mathcal{L}^0$ error slightly increases with the Mach number, which is predictable based on the trend of the lift coefficient appearing more linear at lower AoA and Mach numbers. On the other hand, the prediction of drag and moment coefficient find higher discrepancy across the flight conditions. Although the point-wise prediction of both C_d and C_m is highly accurate, with $\mathcal{L}^0$ values lower than 0.50%, the

plots of the coefficients' trend against the AoA show a higher discrepancy across the flight conditions. In contrast to previous studies of Du et al. [55] and Li et al. [51], the C_d function is better approximate for higher Mach numbers, meanwhile, the SM loses performance for lower subsonic regimes. Mirrored behaviour is registered in the C_m prediction, which is worst-evaluated with the increasing of the Mach number.

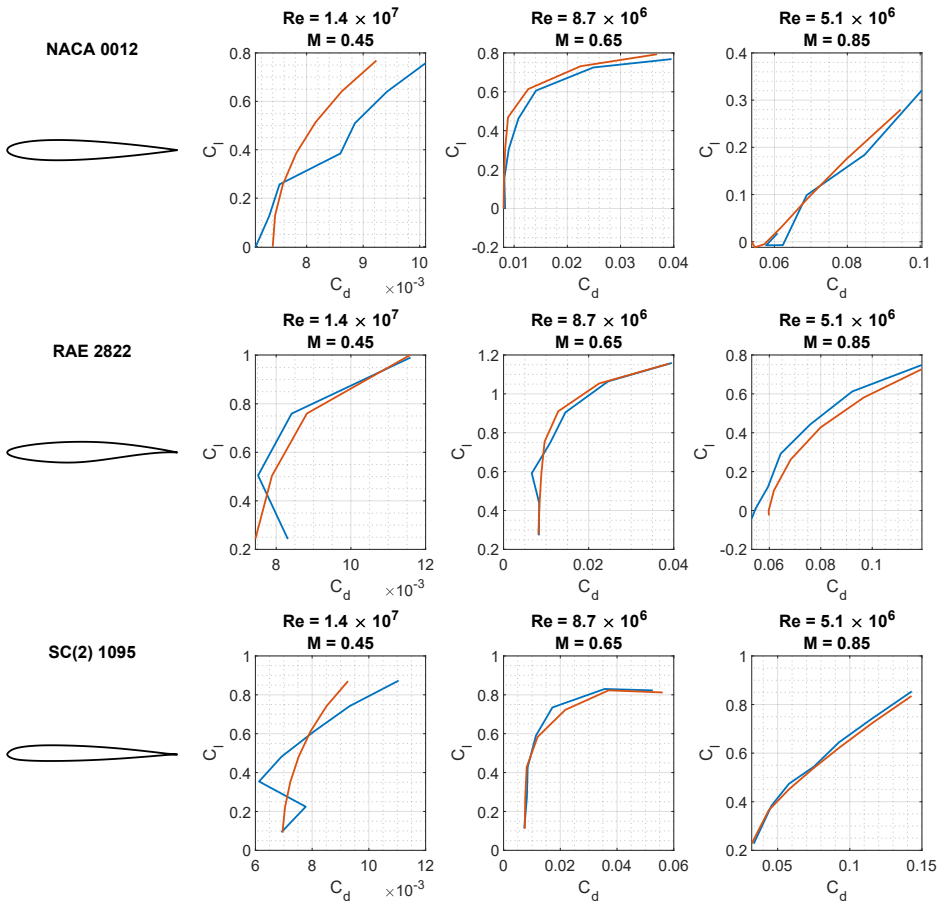
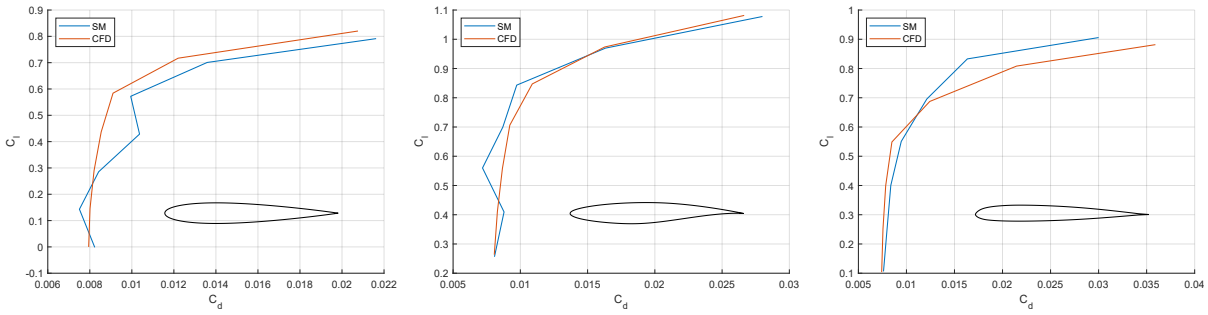


Figure 4.8: Comparison of predicted aerodynamic polar with the surrogate model (blue line) and the polar obtained via CFD simulations (orange line). The aerofoil shape is parameterised by rows while the Reynolds and Mach numbers are held constant in columns.

In addition, to complete the evaluation of the performance of the model on unseen geometries, the aerodynamic drag polar for each aerofoil at different Reynolds and Mach numbers are presented in Figure 4.8. In order to ensure continuity with the previous study, on each row the polar for decreasing Reynolds number and increasing Mach number is plotted, showing an intersectional pattern of performance across flight regimes and aerofoil shapes. Firstly, from the comparison of the polar for different shapes at parameterised Reynolds number, it is possible to affirm that the SM performs consistently for each Reynolds number represented in the dataset, demonstrating that the Reynolds variation does not affect significantly the learning of the underlined physics of the aerodynamic coefficients. Secondly, observing the results of Figures 4.7-4.8, it is evident that the Mach number and AoA are the most influen-

tial flight parameters. Although each coefficient is well approximated for each flight condition, registering relatively low absolute errors for each regime, the SM is not always able to understand the underlying relation between C_l and C_d within the dataset. Regardless of the aerofoil shape simulated, the aerodynamic drag polar for $M = 0.45$ is poorly understood from the SM, since the trend of the C_d coefficient is largely misinterpreted. On the other hand, the physics relationship between lift and drag is mostly depicted for Mach numbers higher than 0.65. This suggests that the MLP architecture proposed may not accurately represent all the phenomena present at low flow velocities. This might be due to the fact that the SM does not depict all the viscous effects within the dataset, which are increasingly relevant for low Mach numbers. Moreover, C_d values are not uniformly distributed across the dataset as they are for C_l. This nonuniform distribution can make it challenging for the network to generalise. Indeed, when training regression DL models, a uniform distribution of the dataset enhances performance and helps the network detect high and low-level features in the samples [52].

Lastly, taking into consideration the poor performance for Mach number included in the range of [0.35 - 0.55], we decided to conduct a study for a flight condition not included in the dataset, in order to understand the true limits of the SM. Therefore, considering the low influence of the Reynolds number and the higher variance of the Mach number, we decided to investigate the predicted polar for $M = 0.60$ and $Re = 8.7 \times 10^6$, based on the climb phase of the hypothetical mission profile schemed in Figure 3.3. In this last case study, the $\mathcal{L}^0$ error between the SM prediction and the CFD simulation of C_l and C_d is lower than 0.76% for each simulated AoA. The results in Figure 4.9 demonstrate that each aerofoil tested effectively captures the drag polar, with the RAE 2822 aerofoil performing the best. The comparison indicates that the MLP has the ability to predict Mach numbers higher than 0.55 with reasonable accuracy, even though the velocity is not represented in the built dataset. This indicates that the SM has reasonably learned the effect of high Mach numbers on aerodynamic coefficients.



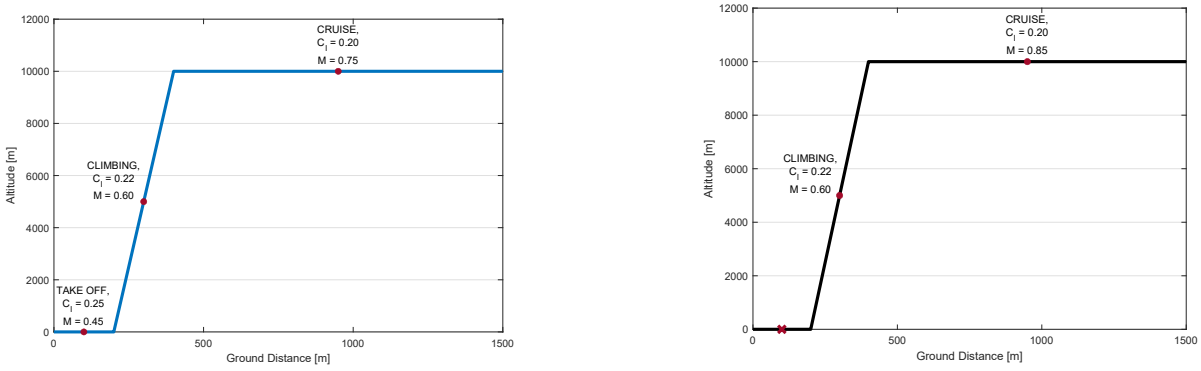
(a) NACA 0012 drag polar. (b) RAE 2822 drag polar. (c) SC(2) 1095 drag polar.

Figure 4.9: Drag polar of NACA 0012, RAE 2822 and SC(2) 1095 for Re = 8.7 x 10^6 and M = 0.60. For visualisation purposes, the geometry of the aerofoil corresponding to each drag polar is added to the plots with a 1:1 aspect ratio.

In conclusion, the DL-based surrogate model, trained on a dataset comprising a broad range of flight regimes and geometries, consistently predicts the aerodynamic polar for each aerofoil shape tested. The uniform performance for different shapes represents the first achievement of this research. Indeed, this thesis aims to explore the potential of a DL model in the context of ASO of a morphing aerofoil, with-

out restricting the variability of the training shapes or the flight regimes of interest. Therefore, this first outcome exhibits the efficiency of the SM in distinguishing different geometry features and consistently mapping each shape to its aerodynamic coefficients. Different results are obtained when comparing the performance of the model along the flight envelope. When the flight condition setups are separately and independently evaluated, the absolute errors between the CFD-simulated coefficients and the SM-predicted coefficients are consistently low, in accordance with the high R^2 scores obtained. Nevertheless, across the flight conditions simulated, the model differently performs within the subsonic and transonic regimes. While the SM accurately depict the underlying relation between lift and drag for Mach numbers higher than 0.60, it loses performance for the prediction of drag coefficient for lower Mach numbers. This factor suggests that the network does not depict all the patterns within the low-subsonic regimes. Using a higher granularity of the AoA for those flight conditions could improve the prediction of the aerodynamic coefficients since it is known that the MLP mainly relies on the fidelity and the size of the dataset [34, 52, 59]. However, this would increase the computational budget of the simulations for the database creation.

Consequently, in light of the last results, we decided to reduce the reference mission profile on which



(a) Hypothetical mission profile.

(b) Effective optimisation mission profile.

Figure 4.10: Comparison of the hypothetical mission profile used for the dataset creation and the effective mission profile used for the set up of the aerofoil optimisation

the baseline aerofoil will be optimised. The poor performance for low subsonic Mach numbers may represent a major setback since the SM is used for aerofoil optimisation and it needs to drive the optimiser algorithm towards the optimal solution discarding the unfeasible shape configurations. Given that 90% of the time is spent in the climbing and cruising phase for a typical aircraft mission profile, in this study the take-off phase is discarded, as depicted in Figure 4.10. Referring to the climbing phase, both the climb altitude of 5,000m and Mach number of 0.60 are preserved. Furthermore, for a fixed cruise altitude of 10,000m, we have increased the highest Mach number from 0.75 to 0.85. By doing so, the optimiser is better guided toward a well-performing region, leading to a clearer differentiation between the climb phase, where is highly unlikely the presence of a lambda shock, and the cruise phase where the transonic regime is encouraged.

4.2 Aerodynamic Shape Optimisation

In order to optimise ASO using the data-based method, the optimisation framework depicted in Figure 3.1 is implemented. In the previous section, we evaluated the effectiveness of the surrogate model that is used to fulfil the aerodynamic computations. The objective of this section is to assess the capabilities of the data-driven model and the parameterisation technique when combined with an optimisation scheme, e.g. the multi-point optimisation with the GA algorithm and the multi-objective optimisation with the NSGA-II algorithm.

In our case study, implementing validation results obtained through a gradient and CFD-based approach is unfeasible. This is due to the fact that, to the best of the author's knowledge, constraining the wing torsion box in MACH-Aero is not compatible with the default constraints employed by the gradient-based optimiser SLSQP [91]. In addition, although the open-source framework SU2 allows the definition of the DoF of each FFD vertex, the meshes which are extruded via *pyHyp* and used for the training of the dataset are not compatible with the CFD-solver included in the platform. Due to the time-consuming process of setting up the SU2 framework and creating the mesh, we chose not to present a gradient-based optimisation for our specific application. Nevertheless, in order to validate the results obtained through the proposed gradient-free surrogate-based method, both multi-objective and multi-point optimisation schemes are employed. Furthermore, a post-processing evaluation of the pressure coefficient, C_p, coupled with C_l and C_d coefficients, is conducted on each resulting optimal shape using *ADflow*. By comparing the two different optimisation schemes, a valuable decision-making resource is supplied to the users, aiding them in determining the feasibility of a morphing architecture tailored to the mission profile of interest. Moreover, the results are simulated with the numerical setup presented in Sections 3.2.2 and 3.2.3, giving a measure of the generalisation capabilities of the overall optimisation framework.

The GA and NSGA-II algorithms are implemented with the *pymoo* framework [160]. Similar to the training process of the aerodynamic SM, manual hyperparameter-tuning is conducted to maximise the performance of both optimisation algorithms. This fine-tuning process is critical because the genetic algorithms involve significant stochasticity due to the crossover and mutation operations. To achieve consistent and optimal results within a real-time optimisation framework, multiple configurations of GA and NSGA-II are respectively tested on the multi-point and multi-objective optimisation problems mathematically proposed in Section 4.2.1. Due to the large size of the parameter space and the consequent computational costs, it is not feasible to perform a complete sampling. Therefore, we conducted individual testing of each critical parameter of GA and NSGA-II to obtain the final setup used in this study. In order to ensure reproducibility, we used a fixed random seed for iterative optimisation of each parameter, similar to the training of the SM.

Table 4.7 highlights the parametric study conducted for both optimisers. In both algorithms, the crossover operator and the mutation operator are identical with slight differences between probabilities and distribution indices. In this thesis work, the first optimisation conducted has been the multi-objective optimisation, therefore the choice of the operators is made with reference to NSGA-II *pymoo* documentation [161], while the GA is accordingly tuned to have comparable results. The NSGA-II algorithm

implemented in *pymoo* uses by default the Simulated Binary Crossover (SBX) [162] as the crossover operator with a probability of 0.5 and a distribution index of 15 [161]. The SBX operator is usually recommended for applications which deal with optimisation problems involving real-valued variables since it smoothly combines the parent variables in the offspring. Moreover, the SBX crossover provides a balance between exploration and exploitation, thereby facilitating the search for the global minimum and escaping local minima. Hence, also influenced by the fact that NSGA-II employs Rank parent selection, we chose to maintain the default SBX operator and fine-tune the probability and distribution index to ensure convergence. The crossover probability is gradually increased from 0.5 to 1.0 to involve more parents in the crossover process. The relatively high probability encourages the exploration of the solution space as it increases the diversity of the offspring, avoiding also the premature convergence of the algorithm. Additionally, the distribution index, *'eta'*, is tuned between the value of [5, 20] in order to study its influence on the solutions. Given the performance achieved by the SM and considering that the objective functions evaluated by the optimisers are exactly derived by the combination of C_l and C_d predicted, we decided to ensure a higher value of the distribution index in order to favour exploration of the design space rather than the exploitation of the current population. Based on multiple runs, we have discovered that selecting high values of *'eta'* for SBX and an average probability value can promote a good compromise between exploring and exploiting the current population. This approach favours diversity of the population in further generations and avoids earlier convergence to unfeasible results. As explained in Section 3.5, genetic algorithms generate new candidates with the cooperation of crossover and mutation. In order to balance exploration and exploitation and improve the approximations of Pareto-optimal solutions, we chose to complement SBX with the Polynomial Mutation operator [162]. This combination enables easier navigation of the solution space. The mutation probability was tuned between 0.0 and 1.0, with higher probabilities resulting in larger perturbations in the solution.

Table 4.7: NSGAII and GA parametric study.

Hyperparameter	Value Range	Final Value	
		NSGA-II	GA
Population size	[90, 180]	100	100
Number of generations	[10, 400]	200	100
Parent Selection type	-	Rank	Rank
Crossover operator	-	SBX	SBX
Crossover probability	[0.5, 1.0]	0.5	0.5
Crossover distribution index	-	15	20
Mutation operator	-	Polynomial Mutation	Polynomial Mutation
Mutation rate	[0.1, 1.0]	0.5	0.4

Furthermore, since there is no direct mapping between the database geometries and the FFD parameterisation, it is not possible to use the same geometries used in the SM database as the initial

population. Therefore the Latin Hypercube Sampling (LHS) [163] is used to randomly sample across the design space for population initialisation ensuring an efficient exploration of the design space with a considerable reduction of computational costs. As detailed in Section 4.2.1, both multi-objective and multi-point optimisation problems are used to minimise aerodynamic drag. Both problems optimised 10 design variables, including the displacement of the FFD points and the AoAs. It is important to note that in our specific optimisation framework, both NSGA-II and GA consider the perturbation of each geometric design variable instead of the effective position. To determine the upper and lower boundaries of LHS sampling in each optimisation problem, we assess multiple configurations within the specified value range:

- The geometric design variables have a lower limit, denoted by $\boldsymbol{d}_l^{(i)}$, which is set between -0.025 and -0.01. The upper limit, $\boldsymbol{d}_u^{(i)}$, is determined by testing values between 0.01 and 0.02. In the final setup, $\boldsymbol{d}_l^{(i)}$ is set to -0.01 and $\boldsymbol{d}_u^{(i)}$ is set to 0.01, where i ranges from 1 to 8.

- To ensure that lift constraints were met, we used the higher range of AoAs provided in the SM training dataset, where $\boldsymbol{d}_l^{(i)} = 0$ and $\boldsymbol{d}_u^{(i)} = 6$ for $i \in [9, 10]$.

4.2.1 Aerodynamic Drag Minimisation

To validate the ASO methodology of a morphing aerofoil from the subsonic to the transonic regime, two optimisation case studies are conducted on the NACA 0012 aerofoil. Firstly, the multi-objective optimisation problem focuses on minimising the drag of the baseline aerofoil for each phase of the mission profile, depicted in Figure 4.10b. Hence, the objective functions in this case are respectively:

- Minimisation of the drag coefficient, $C_{d_{sub}}$, during the climb phase for the aerofoil shape at a freestream Mach number of 0.60, Reynolds number equal to 8.7 x 10^6 and lift coefficient target of $C^*_{l_{sub}} = 0.22$;

- Minimisation of the drag coefficient, $C_{d_{trans}}$, during the cruise phase for the aerofoil shape at a freestream Mach number of 0.85, Reynolds number equal to 5.1 x 10^6 and lift coefficient target of $C^*_{l_{trans}} = 0.20$.

Both functions are optimised with respect to 10 design variables comprising:

- The unconstrained FFD box vertices, $\boldsymbol{y} = [y_1, ..., y_8]$, depicted in Figure 4.11. As explained in Section 3.1, the geometric constraints are ensured through the geometry parameterization of the aerofoil, since the FFD technique defines the DoF of each vertex constituent within the volume box. Therefore, referring to an aerofoil whose chord is normalised to one, each FFD vertex whose x-coordinate is included in the range of [0.25, 0.75] has been fixed. Meanwhile, the eight left points, including the LE and TE, are allowed to move vertically. Their displacement is determined by the optimiser, which iteratively defines the perturbation of each DV through the optimisation process.

- Two angles of attack one for the climbing phase, α_{sub}, and one for the cruise phase, α_{trans}.

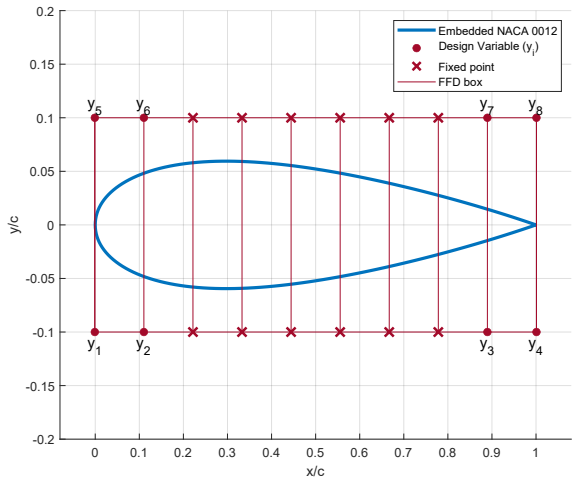


Figure 4.11: Displaying of the NACA 0012 aerofoil enclosed within the FFD box. The fixed points constrained during the optimisation task are marked with an 'x', meanwhile the free-to-move design variables are marked with a 'o'.

In summary, assuming that $\boldsymbol{f}_1 = C_{d_{sub}}$ and $\boldsymbol{f}_2 = C_{d_{trans}}$, the multi-objective problem can be expressed as:

$$\begin{aligned} \text{Minimise} \quad & \boldsymbol{f}(\boldsymbol{d}) = \left[\boldsymbol{f}_1(\boldsymbol{y}, \alpha_{sub}), \boldsymbol{f}_2(\boldsymbol{y}, \alpha_{trans})\right], \\ \text{w.r.t.} \quad & \boldsymbol{d} = \left[\boldsymbol{y}, \alpha_{sub}, \alpha_{trans}\right], \\ \text{subject to} \quad & C_{l_{sub}}(\boldsymbol{y}, \alpha_{sub}) = C_{l^*_{sub}}(\boldsymbol{y}, \alpha_{sub}) \\ & C_{l_{trans}}(\boldsymbol{y}, \alpha_{trans}) = C_{l^*_{trans}}(\boldsymbol{y}, \alpha_{trans}) \,. \end{aligned} \tag{4.3}$$

Unlike the previous problem, the multi-point optimisation aims to minimise the drag of the baseline aerofoil throughout the entire mission profile. Although $C_{d_{sub}}$ and $C_{d_{trans}}$, design variables and constraints are defined as in the multi-objective optimisation, the multi-point optimisation differs in the definition of the objective function. In this case, only one objective function, $\boldsymbol{g}$, is obtained by weighting the two drag coefficients linearly to enhance a singular shape configuration that optimises both flight conditions with respect to the chosen weights. Therefore, considering that the cruise phase has a greater impact on the overall flight mission, the multipoint problem is defined as follows:

$$\begin{aligned} \text{Minimise} \quad & \boldsymbol{g}(\boldsymbol{d}) = 0.1 \cdot C_{d_{sub}}(\boldsymbol{y}, \alpha_{sub}) + 0.9 \cdot C_{d_{trans}}(\boldsymbol{y}, \alpha_{trans}), \\ \text{w.r.t.} \quad & \boldsymbol{d} = \left[\boldsymbol{y}, \alpha_{sub}, \alpha_{trans}\right], \\ \text{subject to} \quad & C_{l_{sub}}(\boldsymbol{y}, \alpha_{sub}) = C_{l^*_{sub}}(\boldsymbol{y}, \alpha_{sub}) \\ & C_{l_{trans}}(\boldsymbol{y}, \alpha_{trans}) = C_{l^*_{trans}}(\boldsymbol{y}, \alpha_{trans}) \,. \end{aligned} \tag{4.4}$$

The main objective of this work is to develop an iterative framework for the ASO of a morphing aerofoil. In order to ensure that the real-time optimisation procedure can be achieved, the inference time of both optimisation algorithms is evaluated. As explained in Section 3.2.4, from each i-th element

of the population, the corresponding aerofoil shape has to be sampled and evaluated twice by the SM, once for the climbing flight condition and once for the cruise flight condition. The SM is used to measure the CPU time required for inference, using an *Intel Core i9-11900* processor. During one optimisation routine, the data-driven model is set up to make simultaneous predictions of several samples, taking 0.92ms of CPU time to predict C_l and C_d coefficients of 200 samples for one flight condition. Therefore, one round of optimisation, including sampling and SM-based predictions, for each shape of a population of 100 using NSGA-II or GA takes about 32 seconds. As a result, a multi-objective optimisation with 200 generations and a population size of 100 demands about 1.77 hours of CPU time to complete. Likewise, a multi-point optimisation of 100 generations and a population size of 100 demands about 0.89 hours of CPU time to complete. Although the SM performs within a competitive time frame, the complete process of sampling and preparation of the shapes for the evaluation is more time-consuming. This is a consequence of using *Python* for developing the optimisation framework, which results in slower performance due to the required CSV file input from the SM. Nevertheless, by rapidly evaluating the multiple aerofoil shapes during the optimisation routine, this approach enables an efficient process, meeting the lack of implementation within the open-source codes distributed and addressing the request for interactive optimisation.

As explained in Section 3.5, the genetic algorithms are population-based frameworks. Therefore the SM is called upon to evaluate the objective functions, $[C^{(k)}_{d_{sub}}, C^{(k)}_{d_{trans}}]$, and the constraint functions, $[C^{(k)}_{l_{sub}}, C^{(k)}_{l_{trans}}]$, for i-th element of the current population, with $k \in [1, ..., N_{pop}]$. The population size, N_{pop}, is tuned and set up during the first stage of the optimisation development. Furthermore, when solving an optimisation problem using genetic algorithms, a penalty function has to be employed to handle the constraints. Consequently, for each objective function defined in Problems 4.3 - 4.4, ad additive penalty function is computed for each i-th element of the current population as follows:

- In the context of the NSGA-II two penalty functions, defined as:

$$\mathcal{P}^{(k)}_{sub} = \beta_{sub} \cdot \sqrt{\left(C^{(k)}_{l_{sub}} - C^{(k)}_{l^*_{sub}}\right)^2}, \tag{4.5}$$

$$\mathcal{P}^{(k)}_{trans} = \beta_{trans} \cdot \sqrt{\left(C^{(k)}_{l_{trans}} - C^{(k)}_{l^*_{trans}}\right)^2}, \tag{4.6}$$

  are separately computed for each k-element of the current population. Therefore the optimiser handles with two objective functions, obtained as $\widehat{\boldsymbol{f_1}} = C_{d_{sub}} + \mathcal{P}_{sub}$ and $\widehat{\boldsymbol{f_2}} = C_{d_{trans}} + \mathcal{P}_{trans}$.

- Regarding the multi-point optimisation driven by the GA, a single penalty function is formulated as follows:

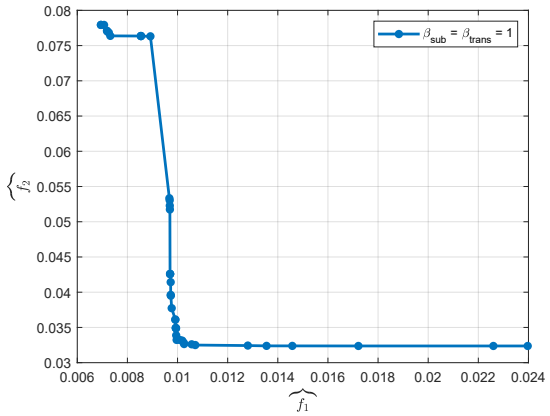
$$\mathcal{P}^{(k)}_{GA} = \beta_{GA} \cdot \left(\sqrt{\left(C^{(k)}_{l_{sub}} - C^{(k)}_{l^*_{sub}}\right)^2} + \sqrt{\left(C^{(k)}_{l_{trans}} - C^{(k)}_{l^*_{trans}}\right)^2}\right), \tag{4.7}$$

  is added to the objective function defined in Problem 4.4 for each k-th member of the current population. In this case, the optimiser searches for that configuration that better minimises the objective function $\widehat{\boldsymbol{g}} = \boldsymbol{g} + \mathcal{P}_{GA}$.

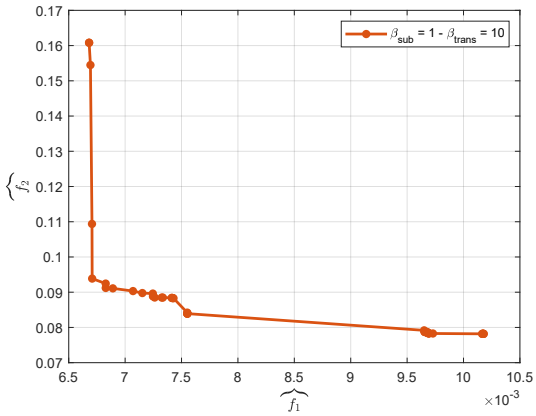
As we have conducted the hyperparameter tuning of the genetic algorithm in Section 4.2, we have assessed fine-tuning factors, namely β_{sub} and β_{trans}, to ensure a good balance discarding unfeasible

data and escaping potential loops due to the extreme restrictive penalty. Multiple experiments were conducted using the fixed setups outlined in Table 4.7 for both the NSGA-II and GA algorithms. In these experiments, we carefully examined various parameters to strike the optimal balance between computational efficiency and accuracy. Our evaluation of accuracy was based on a comparison with the post-processing results obtained through *ADflow*. Figures 4.12 compare different Pareto fronts, obtained for varying penalty function combinations.

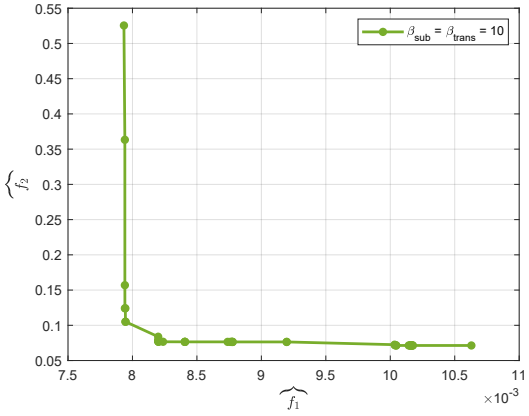
It is worth noting that low values of β in each penalty function result in a smoother and more continuous Pareto front in the context of multi-objective optimisation. Nevertheless, the SM-predicted coefficients of the Pareto solutions, computed with β values lower than 10, show a high discrepancy with the CFD performance, with relative errors higher than 50% for both lift and drag coefficients. Moreover, when using low penalty functions, there is a notable increase in the number of infeasible shapes that are collected. Consequently, both β_{sub} and β_{trans} are imposed equal to 10 to better balance the trade-off between the objective function and constraint satisfaction. Furthermore, to enhance the comparability with the results of multi-objective optimisation, the β_{GA} factor used in the multi-point optimisation, has been consistently set to 10. This standardisation ensures a consistent basis for comparison between the two optimisation approaches.



(a) $\beta_{sub} = \beta_{trans} = 1$.



(b) $\beta_{sub} = 1$, $\beta_{trans} = 10$.



(c) $\beta_{sub} = \beta_{trans} = 10$.

Figure 4.12: Pareto fronts obtained at varying penalties through NSGA-II, utilising an initial population size of 100 and 200 generations.

Lastly, to thoroughly validate the generalisation achieved by the SBO framework, we carried out a mesh convergence study on each of the optimal solutions presented. In this thesis work, the same mesh configuration used in the generation of the training dataset is employed. Conducting the study with the same mesh layouts is instrumental in verifying the fundamental assumptions made during the formulation of the methodology. As outlined in Section 3.2.3, the aerofoil solutions are subject to resampling with an increasing number of points and three refined meshes are extruded using *pyHyp*.

Table 4.8: Refined mesh setups for validation of optimised solution resulting from multi-point and multi-objective problems.

Mesh case	$\mathrm{N_{offwall}}$	$\mathrm{N_{stream}}$	Number of cells
Course	70	133	9314
Medium	129	201	25929
Fine	180	401	72180

Similar to the procedure adopted for database generation, the mesh resolution is iteratively coarsened with a constant ratio, r, of 2.78, resulting in the setup outlined in Table 4.8. Furthermore, for assessing consistency, we have specified the thickness of the first attached layer equal to 2 x 10^{-6} m and the marching distance of 100 chords.

Having defined the optimisation problems and properly assessed the penalty functions for each optimiser, the remaining part of the section is dedicated to the presentation of a comprehensive comparative analysis. This analysis seeks to provide an in-depth evaluation of the results generated by both multi-point and multi-objective optimisation strategies, thereby contributing to a formal assessment of their relative advantages and performance characteristics. For the mission profile of interest, NSGA-II has yielded a Pareto front comprising 43 elements, as depicted in Figure 4.13. Recalling what was mentioned in Section 3.5, a Pareto front represents a collection of solutions in multi-objective optimisation that best trade-offs between the conflicting objective functions. It effectively conveys a set of superior solutions where enhancing one objective comes at the cost of degrading performance in the other, within the so-called non-dominated frontier. Coherently with the underlying theoretical principles, every single element of the proposed Pareto effectively minimises both $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$, while adhering to the lift constraints $C^*_{l_{sub}} = 0.22$ and $C^*_{l_{trans}} = 0.20$. Furthermore, the relative $\mathcal{L}^1$ error is defined as follows:

$$\epsilon^{(i)}_{\mathcal{L}^1} = \left|\frac{\widetilde{\mathrm{y}}_i - \mathrm{y}_i}{\widetilde{\mathrm{y}}_i}\right|, \tag{4.8}$$

where $\widetilde{\mathrm{y}}_i$ represents the SM-predicted coefficient for the i-th sample, and y_i represents the CFD-simulated coefficient for the same i-th sample. By estimating the $\mathcal{L}^0$ distribution it is possible to analyse the errors across the aerodynamic coefficients predicted by the SM. Upon completing the multi-objective optimisation process, it becomes evident that all instances of $C_l^{(k)}$, with k in the range of 1 to 43, respect the target lift constraints exhibiting an error rate that does not surpass 0.70%. This observation signifies that NSGA-II effectively explores the design space, culminating in a convergent and highly satisfactory

solution. Moreover, the L-shaped Pareto underscores the relation between the two objective functions, suggesting that the cluster of solutions is split into two main groups. Upon thorough exploration of the solution space, one group of aerofoil outperforms all others on one objective, culminating at a flexion point. This particular scenario suggests that the optimisation process, aimed at identifying the optimal morphing configuration from the baseline, consistently generates divergent shapes. This, in turn, affirms the variation in design requirements across different segments of the mission profile. In the post-

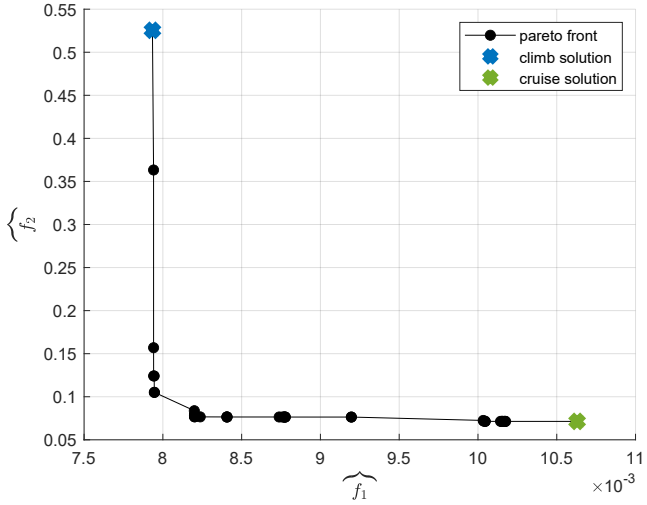


Figure 4.13: L-shaped Pareto front obtained through NSGA-II for the multi-objective optimisation problem. The two solutions that best optimise the two objective function are marked with an 'x': in blue for the one that better optimises the climb phase, and in green for the one that better optimise the cruise phase.

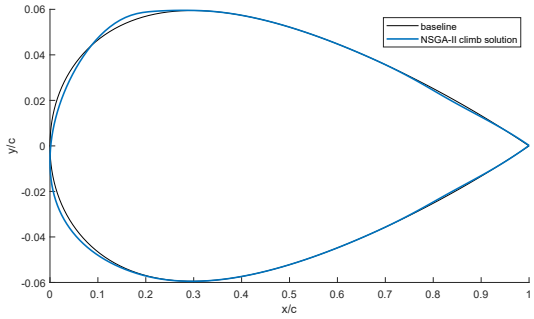
processing of NSGA-II output, we have identified the two solutions that excel in optimising drag, one for the cruise phase and one for the climb phase. Leveraging the aerodynamic coefficients predicted by the SM, it is noteworthy that these two selected airfoil shapes are positioned at the extreme ends of the Pareto front, aligning with the theoretical expectations. Table 4.9 represents the aerodynamic performance of both aerofoil geometries. The shapes are evaluated both by the SM and by *ADflow*. The CFD simulations took into account the specific Mach and Reynolds numbers defined in the mission profile, as well as the optimised values of the angle of attack, denoted as α_{sub} and α_{trans}. Through the optimisation process, we have gathered two optimal AoAs, specifically $\alpha_{sub} = 1.82°$ for the climb phase and $\alpha_{trans} = 3.75°$ for the cruise phase. The presented CFD coefficients correspond to those obtained using the medium-refined mesh, which was consistently used for every sample of the training dataset. In both flight conditions, the mesh convergence study, following the Richardons extrapolation procedure, confirms the convergence of the solution. The optimal aerofoil for the climb phases converges at a rate of $s = 1.0865$, while the cruise solution converges at a rate of $s = 0.3260$. By comparing the solvers performance, the SM consistently underestimated the drag with a relative error lower than 5% in both cases. Furthermore, the lift coefficient obtained by *ADflow* is higher than the one resulting from the optimisation. While the SM has returned values of C_l of the same magnitude as the target lift constraints in both solutions, the CFD overestimates those values. This discrepancy is more significant in the cruise

case with a relative error of 32.19%.

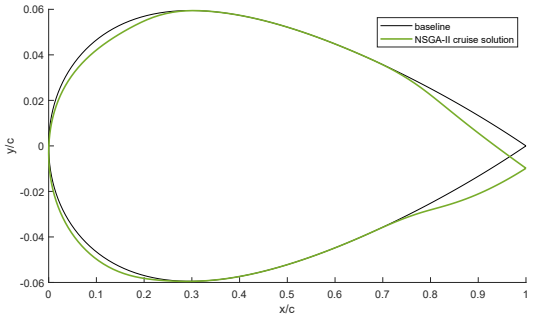
Table 4.9: SM and CFD comparison for NSGA-II solutions.

Solver	Climb		Cruise	
	C_l counts	C_d counts	C_l counts	C_d counts
SM	220.01	78.09	200.00	713.75
CFD	253.13	82.11	294.93	733.08
$\mathcal{L}^1$ error	13.08%	4.89%	32.19%	2.63%

The SBO method implementing the multi-objective optimiser NSGA-II has successfully yielded optimal shapes while adhering to the constraint of the wing box. Figure 4.14 shows the two resulting solutions, in comparison with the NACA 0012 baseline.



(a) Optimal aerofoil shape for climb phase in the context of multi-objective optimisation problem.



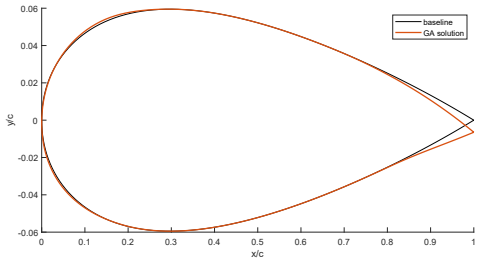
(b) Optimal aerofoil shape for cruise phase in the context of multi-objective optimisation problem.

Figure 4.14: Multi-objcetive optimal solutions for climb and cruise phases.

On the left-hand side, the morphed aerofoil shape presents alterations in the proximity of the LE rather than the TE. This modification aligns with the principles of the aerodynamic characteristic of the subsonic field. Notably, high-performing aerofoil shapes within the UIUC library often feature asymmetric geometries. This asymmetry, especially around the LE, is strategically designed to optimise the acceleration of airflow over the upper surface and to manage the stagnation point effectively. In the context of subsonic climb optimisation, the algorithms have been shown to adjust the aerofoil shape coherently with these established aerodynamic principles.

On the right-hand side, the optimal morphed aerofoil for the cruise speed is plotted. In contrast to the solution obtained for the climb phase, the algorithm has predominantly modified the shape close to the TE, with no significant shift in the position of the LE. As evident from the plot, the optimal shape displays greater camber compared to the baseline airfoil, enhancing the required lift. In the context of the transonic regime, both TE and camber play pivotal roles in shaping the drag distribution around the aerofoil. Modifying the camber indeed has a beneficial impact on the location of the lambda shock impingement. Furthermore, the inclusion of a sharp TE is designed to reduce wave drag, a key concern in transonic flight. Additionally, it is important to emphasise the significance of the airfoil thickness in this context. Generally, the maximum thickness is strategically designed to be as near as possible to the

LE, anticipating the formation of the lambda shock and, consequently, minimising the drag. However, in our study where the wing box constraint is active, the algorithm addresses this feature by reshaping the upper surface in the proximity of the LE, ensuring that the maximum thickness is located near the forward boundary of the wing box.



Solver	Climb		Cruise	
	C_l counts	C_d counts	C_l counts	C_d counts
SM	220.00	99.82	200.00	801.71
CFD	310.71	88.30	378.53	911.75
$\mathcal{L}^1$ error	29.19%	13.05%	47.16%	12.07%

Figure 4.15: Multi-point optimal solution for both climb and cruise phases.

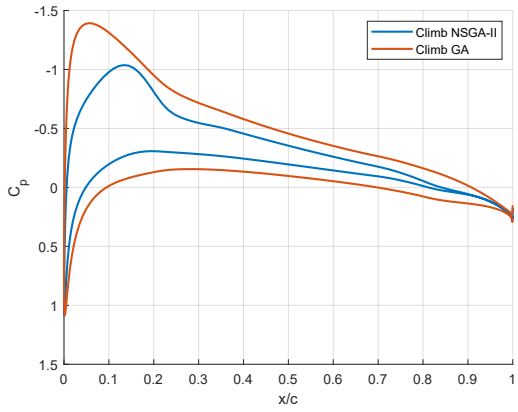
Comparable results are obtained from the multi-point optimisation gathered through the GA. In this case, only one optimal shape is acquired which best optimises the mission profile, with respect to the two resulting AoAs. The optimal aerofoil geometry with its aerodynamic performance is depicted in Figure 4.15. As it was conducted for the outputs of the multi-objective problem, the solution of the multi-point problem undergoes the grid convergence study. While the climb configuration simulated, for $\alpha_{sub} = 2.60°$, confirms the convergence of the solution with a rate of $s = 0.9206$, the cruise configuration, for $\alpha_{trans} = 4.08°$, does not achieve the asymptotic convergent trend with the refinement of the mesh. Considering the multiple factors responsible for this behaviour, we have found that the mesh used for the grid convergence study dataset does not completely dissipate the turbulent wake energy. This indicates that the mesh size of 100 chords is not suitable, but it has been necessary to augment the size in order to capture all the phenomena present in the field. Therefore, in order to present consistent results, for the specific geometry we have conducted a different mesh convergence study keeping the number of cells defined in Table 4.8, and increasing the distance between the aerofoil wall and the outermost edge from 100 to 500 chord lengths. In this way, the convergence has been assessed with a rate of $s = 0.8957$. The recent findings have highlighted a limitation in the proposed methodology, where a constant and uniform mesh is adopted for every combination of shapes and flight conditions. Although this approach successfully managed computational costs and facilitated the automated creation of the database used for training the SM, it introduced systematic errors within the samples. This occurred due to the vast variability within the design space, making it unfeasible to confirm convergence for each individual shape and flight condition incorporated in the dataset. However, despite the inevitable limitations due to the made assumptions and the limited performance of the SM, it can be affirmed that the SBO framework consistently delivers optimal solutions across the range of developed problems.

Table 4.10 compares the aerodynamic performance of optimal solutions obtained through multi-objective and multi-point optimisation. It is noteworthy that the solutions derived from the multi-objective optimisation approach exhibit superior optimisation in both objective functions. The final outputs obtained through multi-objective and multi-point optimisation are consistent, recording a similar trend in

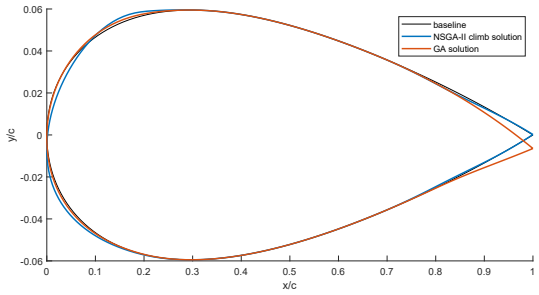
Table 4.10: Comparison of NSGA-II and GA solutions.

Optimiser	Climb			Cruise		
	C_l counts	C_d counts	α_{sub}	C_l counts	C_d counts	α_{trans}
NSGA-II	220.01	78.09	1.81°	200.00	713.75	3.75°
GA	220.00	99.82	2.60°	200.00	801.71	4.46°
Δ counts	-0.01	21.73	-	0.0	87.97	-

the AoAs which increase from the subsonic regime to the transonic regime. In the climb phase, the NSGA-II algorithm excels at minimising drag while maintaining similar lift performance, resulting in a significant reduction of 21.73 drag counts compared to the solution produced by the GA. Furthermore, in the cruise phase, the GA algorithm is again outperformed from NSGA-II, whose solution results in a reduction of 87.97 counts. In addition, the C_p distribution of each optimal shape is included in order to provide the reader with a comprehensive view. It has to be mentioned that the distributions illustrated in Figures 4.16-4.17 are obtained using *ADflow*, in accordance with the convergence grid study discussed.



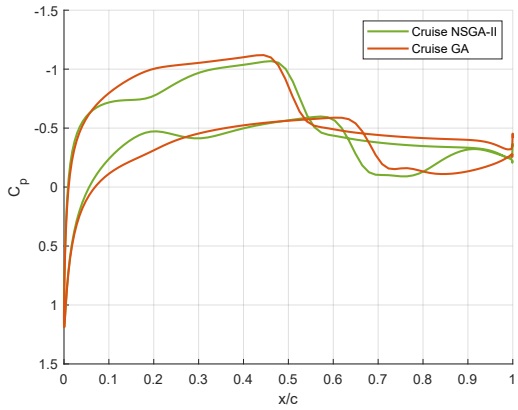
(a) C_p distribution for climb phase.



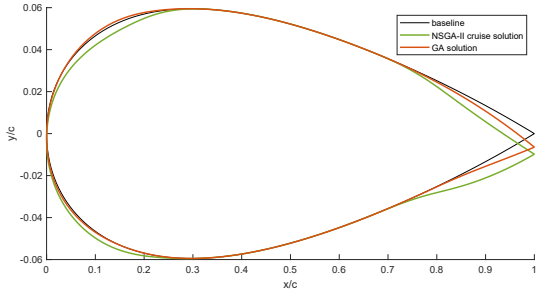
(b) NSGA-II (blue line) and GA (orange line) optimal solutions for climb phase.

Figure 4.16: Pressure coefficient distributions for climb phase in multi-objective and multi-point optimisation.

Firstly, it is important to note that the distribution of the pressure coefficient for the climb phase aligns with the lift and drag coefficients reported in Tables 4.14-4.15. In both cases, the CFD-predicted lift coefficient considerably exceeds the values computed by the SM during the optimisation. For the climb phase scenario, the C_p associated with the multi-point optimisation presents a 34% higher peak than the one derived from the multi-objective optimisation. This discrepancy totally aligns with the differences in lift coefficients: 310.71 counts for the GA output compared to 253.13 counts for the shape resulting from NSGA-II. Furthermore, the lower pressure depicted by the orange line anticipates the lower pressure for the blue lines, in accordance with the respective shapes. Indeed, while both optimisers adjust the shape near the LE to optimise drag during the climb phase, the GA solution sets the maximum thickness closer to the LE. This, coupled with the higher value of the α_{sub}, has resulted in a greater lift coefficient. Besides the different performance in the lift coefficients, both solutions present comparable values of



(a) C_p distribution for cruise phase.



(b) NSGA-II (green line) and GA (orange line) optimal solutions for cruise phase.

Figure 4.17: P pressure coefficient distributions for cruise phase in multi-objective and multi-point optimisation.

CFD-predicted drag: 88.30 counts for the GA output compared to 82.11 counts for the NSGA-II optimised aerofoil. Nevertheless, the camber of the GA profile is higher than the NSGA-II shape, the higher AoA resulting from the multi-point optimisation might account for the slightly higher drag experienced by the GA aerofoil.

For the cruise phase scenario, both aerofoils obtained through the multi-point and multi-objective optimisation present a drop of the pressure, indicating the occurrence of the lamba shock. At higher values of Mach number, the lambda shock is expected and it needs to be managed in order to reduce its effect on the performance. It is worth noting that while both solutions present the impingement of the shock at a similar position along the chord, there is a consistent difference in the CFD-predicted drag between the two airfoil shapes: 773.08 counts for NSGA-II and 911.75 counts for the GA solution. This significant reduction in the drag observed in the former case might be attributed to the higher camber of the multi-objective optimised aerofoil which typically leads to reduced drag for moderate AoA. Beyond the differences in solver performance, comparing the two pressure distributions for the cruise speed fully justifies the higher values of C_l counts for the multi-point output.

Besides the divergence resulting from the post-processing with *ADflow*, this thesis work has effectively performed a data-driven aerodynamic shape optimisation of a morphing aerofoil within a competitive time framework. In the context of the surrogate-based optimisation, the aerofoil solutions gathered with the multi-objective approach outperform the resulting shape of the multi-point optimisation. The comprehensive analysis of the performances of both outcomes has revealed that a morphing configuration effectively minimises the drag along the sketched mission profile. This statement confirms the initial assumption that the adoption of a morphing configuration enhances performance, thereby reducing costs and mitigating environmental impact. However, upon a comparison of the optimal results with the assessment provided by the CFD solver, a significant discrepancy comes to light. It becomes evident that the SBO framework ignored the aerodynamic relation between lift and drag forces for the specific application, resulting in an underestimation of the lift coefficient across all the proposed solutions. While the trend in drag minimisation remains consistent between the results obtained through SM and those

post-processed by CFD, there is a striking disparity in the lift values. The lift values obtained through SM exhibit a relative error exceeding 40% for the multi-point output. This implies a possible limitation of the SM to generalise for those perturbed shapes that are not part of the training dataset and are not included in the UIUC or NASA SC(2) library. However, the authors believe that the source of this discrepancy might lie in the output configurations of the SM. As explained in Section 3.4, the coefficients predicted by the SM align with the choices made in the network developed by Moin et al. [59]. While this approach facilitates a direct comparison of the performance between the new model and the original one, it does not fully account for correlations among the coefficients. To address this issue and enhance overall accuracy and generalisation, creating a database that directly maps the viscous drag coefficient, C_{d_0}, and the pressure drag coefficient, C_{d_p}, might be profitable. This adjustment might allow to create a correlation between lift and drag by employing the drag polar equation ($C_d = C_{d_0} + k_{\mathrm{L}} C_l^2$). Additionally, the drag can be calculated by adding the friction drag and pressure drag ($C_d = C_{d_0} + C_{d_p}$). This is a useful approach to validate the drag polar. Furthermore, by examining the errors and trends in the viscous drag, it may be possible to investigate the hypothesis that the degraded performance of the SM at low Mach numbers is linked to its limited understanding of viscous phenomena. Although this improvement was previously proposed by Moin et al. [59], in our thesis work, we opted to reserve this adjustment for future investigations, as we had already significantly increased the variability of the dataset. It is also important to account that, during the post-processing of the multi-point solution, we needed to modify the mesh to accommodate the cruise speed. This reveals an important factor that could impact future outcomes. It is worth noting that the standardised mesh has accomplished the automatic creation of the dataset while sacrificing the accuracy of some samples, due to the inability of the grid to capture all relevant aerodynamic phenomena for the broad set of shapes included in the dataset, especially in the transonic regime. Therefore, it would be preferable to mitigate this systematic error by increasing the size of the mesh. Lastly, streamlining the selection of training shapes might also be a valuable choice to couple with the expansion of the grid. As stated in Section 4.1.2, the SM has proved to efficiently generalise for different shapes achieving comparable performance with varying geometries. Nevertheless, simplifying the range of shapes to include only NACA aerofoils, RAE aerofoils, and NASA SC(2) aerofoils may prove useful in reducing the overall complexity of the design space while maintaining a significant level of variability in geometric features.

In conclusion, despite some limitations within the database, we can state that the proposed framework has effectively fulfilled the primary objectives of this research. Through comprehensive evaluations of SM performance, we have established its capability to generalise across a diverse range of shapes and its accuracy in evaluating both subsonic and transonic regimes, achieving performance on par with the state-of-the-art. Moreover, the integration of the network with the FFD technique and the genetic algorithm has addressed this gap in the literature, gathering a valuable tool for the optimisation of aerofoil in morphing configuration, resulting in successful drag reduction within the context of a multi-objective setup.

Chapter 5

Conclusions

In the last decades, there has been a growing interest among researchers in morphing airfoils. This growing attention is driven by the promising potential of morphing wings to enhance aircraft performance and achieve greening aviation. Indeed, shape morphing might offer considerable benefits in optimising mission profiles characterised by varying flight conditions. However, a significant challenge in harnessing the full potential of these architectures lies in the considerable computational costs of aerodynamic shape optimisation. The ASO process relies on costly physics-based simulations, and its consequent expenses represent a potential setback in the optimisation of real-time and many-query applications. Moreover, these limitations are pronounced when the aerodynamic design space increases its dimensionality. For these reasons, data-based approaches involving DL surrogates, represent valuable substitutes to use in lieu of the time-consuming and expensive gradient and CFD-based tools.

The main objective of this thesis was to develop a data-driven optimisation framework designed to address the ASO of a two-dimensional morphing aerofoil, striking an acceptable balance between computational efficiency and accuracy. In the presence of the large body of literature in the field, this research aimed to explore the capability of the existing DL models with the goal of finding new potential applications and evaluating the current limitations. Nowadays, all the SMs proposed in the ASO state-of-the-art are constrained in the design space and flight range, thus hindering the generalisation of these tools. Hence a SBO framework has been developed comprising a DL model for the computation of aerodynamics and the FFD technique as the parameterisation method. Those two modules are then integrated into a multi-objective and multi-point optimisation, which includes genetic algorithms in both instances.

As aerodynamic surrogate modelling, the MLP designed by Moin et al. [59] has been chosen to be tailored on a new database, comprising a broad variety of shapes and flight conditions. The original MLP has been fine-tuned in its setups in order to best fit with the new dataset, resulting in the prediction of aerodynamic performance with RMSE values of 2.61‰ for C_d, 12.54‰ for C_l and 1.88‰ for C_m. Although the pointwise distribution of each aerodynamic coefficient is well-represented, the network does not depict the underlying physics within the samples, especially for lower Mach numbers. The consequent loss of accuracy in the subsonic regime might be attributed to the difficulty of the network to

capture all the phenomena characterising the flow, including viscous effects. Furthermore, ensuring the convergence of the mesh implemented to collect data is crucial for the aerodynamic analysis. However, due to the extension of the design space, the methodology implemented in the creation of the dataset employed a standardised mesh. Although this approach is efficient in terms of computational resources, it inevitably leads to errors in the dataset with a reduction in its level of representation.

Despite the limitations of the SM, the proposed SBO framework has demonstrated its effectiveness in the optimisation of a given baseline, all while adhering to the constraints of the wing box, setting a solid foundation for future research and development. The multi-objective framework, in particular, has yielded significant improvements for the reference mission profile. It has led to a significant reduction in drag of 22 counts during the climb phase and 87.97 counts during the cruise phase when compared to the multi-point solution. Even when accounting for discrepancies arising from the CFD post-processing steps of the optimal solutions, the comparison between the two optimisation approaches clearly highlights the improved performance of a morphing configuration over a fixed suboptimal shape. This outcome holds significant promise, especially considering that the optimisation process was successfully completed within a competitive timeframe of less than 2 hours for the multi-objective case. Furthermore, the proposed methodology might present a valuable solution for addressing the current gap in implementing optimisation within distributed open-source codes for the given application. Nevertheless, the most recent findings underscore some limitations of the SM proposed. Based on the gathered results, it is evident that the errors arising from the optimisation process stem from the limited generalisation capabilities of the SM approach used for aerodynamic predictions. This limitation becomes particularly evident when comparing the CFD-simulated performance, highlighting the inability of the SM to accurately capture the relationship between lift and drag. Nevertheless, it is essential to emphasise that these errors are not a result of shortcomings within the optimisation framework itself. This is underlined by the consistent results achieved across both optimisation cases, demonstrating a notable reduction in drag with a relative error compared to CFD of less than 5% for the multi-objective approach.

5.1 Future Work

The presented thesis work has effectively explored the applicability and generalisation capabilities of data-driven models within the development of SBO. This was achieved through the training of the SM with a dataset encompassing various shape features and flight regimes. While the implemented network has proved to accurately predict individual aerodynamic performance, it falls short of achieving the desired level of generalisation due to an incomplete understanding of the underlying physics governing certain flight conditions. In order to address those limitations there are several potential extensions and developments for future research:

- Streamlining the selection of training shapes to fewer classes of shape, thereby enhancing a deeper understanding of the connection between shape variation and aerodynamics. This approach helps mitigate potential errors stemming from mesh inaccuracies across diverse shapes;

- Augmenting the mesh size in order to mitigate the systematic errors within the transonic regime, improving the wake dissipation;
- Reconfiguring the MLP for a different dataset, that directly comprises the viscous drag, C_{d_0}, and the pressure drag, Cd_p. Studying the response of the network when directly dealing with viscosity might give a deep insight into the factors contributing to performance degradation in the subsonic regime. Calculating the drag through the drag polar can improve the correlation between lift and drag, leading to fewer errors. Additionally, this technique can be verified by comparing the drag polar results with the drag obtained by combining the friction and pressure drag.

Despite the limitations of the surrogate, this work has successfully delved into optimisation reaching the conclusion that morphing architecture performs better for the aerodynamic point of view. Once the limitations of the SM have been addressed, prospective future developments could be explored:

- Investigating alternative methodologies aimed at establishing a more adaptable mesh, in alignment with contemporary trends in the literature, which focus on the development of grids capable of automatic adaptation to diverse shapes while maintaining acceptable accuracy;
- Enabling an online training of the SM in order to include morphed candidates into the dataset, thus improving a more comprehensive exploration of the design space;
- Implementing a gradient and CFD-based optimisation in SU2 in order to have a performance comparison between different softwares. Even though SU2 is not automatically adaptable to a generic baseline, accomplishing the same optimisation with a high-fidelity tool would guide the developers towards the improvement of the research.

By addressing these aspects in future work, the proposed framework can undergo further refinement and expansion. This will enable a more comprehensive optimisation of different aerodynamic shapes across a wider range of flight conditions and multidisciplinary fields.

Bibliography

[1] W. Beitz, G. Pahl, and K. Grote. *Engineering design: a systematic approach*, volume 71. Springer London, 3rd edition, 1996. doi: 10.1007/978-1-84628-319-2.

[2] N. Yüksel, H. R. Börklü, H. K. Sezer, and O. E. Canyurt. Review of artificial intelligence applications in engineering design perspective. *Engineering Applications of Artificial Intelligence*, 118: 845–875, 2023. doi: 10.1016/j.engappai.2022.105697.

[3] M. Rychener. *Expert systems for engineering design*. Elsevier, 1st edition, 1988. doi: 10.1016/B978-0-126-05110-0.X5001-0.

[4] T. J. Ross. *Fuzzy logic with engineering applications*. John Wiley & Sons, 3rd edition, 2009. doi: 10.1002/9781119994374.

[5] M. Rafiq, G. Bugmann, and D. Easterbrook. Neural network design for engineering applications. *Computers & Structures*, 79(17):1541–1552, 2001. doi: 10.1016/S0045-7949(01)00039-6.

[6] A. Patnaik and R. Mishra. Ann techniques in microwave engineering. *IEEE Microwave Magazine*, 1(1):55–60, 2000. doi: 10.1109/6668.823828.

[7] A. Baghbani, T. Choudhury, S. Costa, and J. Reiner. Application of artificial intelligence in geotechnical engineering: A state-of-the-art review. *Earth-Science Reviews*, 228:103991, 2022. doi: 10.1016/j.earscirev.2022.103991.

[8] K. Man, K. Tang, and S. Kwong. Genetic algorithms: concepts and applications [in engineering design]. *IEEE Transactions on Industrial Electronics*, 43(5):519–534, 1996. doi: 10.1109/41.538609.

[9] J. Liu, X. Kong, F. Xia, X. Bai, L. Wang, Q. Qing, and I. Lee. Artificial intelligence in the 21st century. *IEEE Access*, 6:34403–34421, 2018. doi: 10.1109/ACCESS.2018.2819688.

[10] J. Li, X. Du, and J. R. Martins. Machine learning in aerodynamic shape optimization. *Progress in Aerospace Sciences*, 134:100849, 2022. doi: 10.1016/j.paerosci.2022.100849.

[11] D. Li, S. Zhao, A. Da Ronch, J. Xiang, J. Drofelnik, Y. Li, L. Zhang, Y. Wu, M. Kintscher, H. P. Monner, A. Rudenko, S. Guo, W. Yin, J. Kirn, S. Storm, and R. D. Breuker. A review of modelling and analysis of morphing wings. *Progress in Aerospace Sciences*, 100:46–62, 2018. doi: 10.1016/j.paerosci.2018.06.002.

[12] S. Joshi, Z. Tidwell, W. Crossley, and S. Ramakrishnan. Comparison of morphing wing strategies based upon aircraft performance impacts. In *45th AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics & materials conference*, page 1722, June 2004. doi: 10.2514/6.2004-1722.

[13] S. Ameduri and A. Concilio. Morphing wings review: Aims, challenges, and current open issues of a technology. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 237(18):4112–4130, 2023. doi: 10.1177/0954406220944423.

[14] F. Afonso, J. Vale, F. Lau, and A. Suleman. Performance based multidisciplinary design optimization of morphing aircraft. *Aerospace Science and Technology*, 67:1–12, 2017. doi: 10.1016/j.ast.2017.03.029.

[15] A. Jameson. Aerodynamic design via control theory. *Journal of scientific computing*, 3:233–260, 1988. doi: 10.1007/BF01061285.

[16] G. K. Kenway, C. A. Mader, P. He, and J. R. Martins. Effective adjoint approaches for computational fluid dynamics. *Progress in Aerospace Sciences*, 110:100542, 2019. doi: 10.1016/j.paerosci.2019.05.002.

[17] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. Kevin Tucker. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, 41(1):1–28, 2005. doi: 10.1016/j.paerosci.2005.02.001.

[18] A. I. Forrester and A. J. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1):50–79, 2009. doi: 10.1016/j.paerosci.2008.11.001.

[19] J. P. Kleijnen. Kriging metamodeling in simulation: A review. *European Journal of Operational Research*, 192(3):707–716, 2009. doi: 10.1016/j.ejor.2007.10.013.

[20] M. A. Bouhlel, S. He, and J. R. Martins. Scalable gradient–enhanced artificial neural networks for airfoil shape design in the subsonic and transonic regimes. *Structural and Multidisciplinary Optimization*, 61:1363–1376, 2020. doi: 10.1007/s00158-020-02488-5.

[21] X. Yan, J. Zhu, M. Kuang, and X. Wang. Aerodynamic shape optimization using a novel optimizer based on machine learning techniques. *Aerospace Science and Technology*, 86:826–835, 2019. doi: 10.1016/j.ast.2019.02.003.

[22] M. Nemec, D. W. Zingg, and T. H. Pulliam. Multipoint and multi-objective aerodynamic shape optimization. *AIAA journal*, 42(6):1057–1065, 2004. doi: 10.2514/1.10415.

[23] A. Jameson and J. Alonso. Automatic aerodynamic optimization on distributed memory architectures. In *34th Aerospace Sciences Meeting and Exhibit*, page 409, January 1996. doi: 10.2514/6.1996-409.

[24] J. R. Martins. Aerodynamic design optimization: Challenges and perspectives. *Computers & Fluids*, 239:105391, 2022. doi: 10.1016/j.compfluid.2022.105391.

[25] X. He, J. Li, C. A. Mader, A. Yildirim, and J. R. Martins. Robust aerodynamic shape optimization—from a circle to an airfoil. *Aerospace Science and Technology*, 87:48–61, 2019. doi: 10.1016/j.ast.2019.01.051.

[26] H. P. Buckley, B. Y. Zhou, and D. W. Zingg. Airfoil optimization using practical aerodynamic design requirements. *Journal of Aircraft*, 47(5):1707–1719, 2010. doi: 10.2514/1.C000256.

[27] D. Shi-Dong, C.-H. Chen, and S. Nadarajah. Adjoint-based aerodynamic optimization of benchmark crm wing. In *35th AIAA Applied Aerodynamics Conference*, June 2017. doi: 10.2514/6.2017-3755.

[28] Z. Lyu, G. K. Kenway, and J. R. Martins. Aerodynamic shape optimization investigations of the common research model wing benchmark. *AIAA journal*, 53(4):968–985, 2015. doi: 10.2514/1.J053318.

[29] S. Xu, S. Timme, O. Mykhaskiv, and J.-D. Müller. Wing-body junction optimisation with cad-based parametrisation including a moving intersection. *Aerospace Science and Technology*, 68:543–551, 2017. doi: 10.1016/j.ast.2017.06.014.

[30] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. Mavriplis. Cfd vision 2030 study: A path to revolutionary computational aerosciences. technical report cr–2014-218178, 2014. URL `https://ntrs.nasa.gov/citations/20140003093`. Accessed: 27/10/2023.

[31] A. Yildirim, G. K. Kenway, C. A. Mader, and J. R. Martins. A jacobian-free approximate newton–krylov startup strategy for rans simulations. *Journal of Computational Physics*, 397:108741, 2019. doi: 10.1016/j.jcp.2019.06.018.

[32] C. Schillings, S. Schmidt, and V. Schulz. Efficient shape optimization for certain and uncertain aerodynamic design. *Computers & Fluids*, 46(1):78–87, 2011. doi: 10.1016/j.compfluid.2010.12.007.

[33] A. Jahangirian and A. Shahrokhi. Aerodynamic shape optimization using efficient evolutionary algorithms and unstructured cfd solver. *Computers & Fluids*, 46(1):270–276, 2011. doi: 10.1016/j.compfluid.2011.02.010.

[34] X. Zhang, F. Xie, T. Ji, Z. Zhu, and Y. Zheng. Multi-fidelity deep neural network surrogate model for aerodynamic shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 373:113485, 2021. doi: 10.1016/j.cma.2020.113485.

[35] H. Chen, L. He, W. Qian, and S. Wang. Multiple aerodynamic coefficient prediction of airfoils using a convolutional neural network. *Symmetry*, 12(4):544, 2020. doi: 10.3390/sym12040544.

[36] V. Sekar, M. Zhang, C. Shu, and B. C. Khoo. Inverse design of airfoil using a deep convolutional neural network. 57(3):993–1003, 2019. doi: 10.2514/1.J057894.

[37] R. Li, Y. Zhang, and H. Chen. Learning the aerodynamic design of supercritical airfoils through deep reinforcement learning. 59(10):3988–4001, 2021. doi: 10.2514/1.J060189.

[38] J. R. Martins and A. Ning. *Engineering design optimization*. Cambridge University Press, 1st edition, 2021. doi: 10.1017/9781108980647.

[39] W. Faller and S. Schreck. Real-time prediction of unsteady aerodynamics: Application for aircraft control and manoeuvrability enhancement. *IEEE Transactions on Neural Networks*, 6(6):1461–1468, 1995. doi: 10.1109/72.471362.

[40] E. Du, M. Sleeman, and M. Yano. Adaptive discontinuous-galerkin reduced-basis reduced-quadrature method for many-query cfd problems. In *AIAA AVIATION 2021 FORUM*, July 2021. doi: 10.2514/6.2021-2716.

[41] 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*, 2018. doi: 10.1016/j.paerosci.2017.11.003.

[42] V. Raul and L. Leifsson. Surrogate-based aerodynamic shape optimization for delaying airfoil dynamic stall using kriging regression and infill criteria. *Aerospace Science and Technology*, 111: 106555, 2021. doi: 10.1016/j.ast.2021.106555.

[43] B. Liu, H. Liang, Z.-H. Han, and G. Yang. Surrogate-based aerodynamic shape optimization of a morphing wing considering a wide mach-number range. *Aerospace Science and Technology*, 124:107557, 2022. doi: 10.1016/j.ast.2022.107557.

[44] D. E. Myers. Matrix formulation of co-kriging. *Journal of the International Association for Mathematical Geology*, 14:249–257, 1982. doi: 10.1007/BF01032887.

[45] D. E. Myers. Co-kriging -— new developments. In *Geostatistics for Natural Resources Characterization: Part 1*, pages 295–305. Springer, 1984. doi: 10.1007/978-94-009-3699-7_18.

[46] Z.-H. Han and S. Görtz. Hierarchical kriging model for variable-fidelity surrogate modeling. 50(9): 1885–1896, 2012. doi: 10.2514/1.J051354.

[47] T. Crestaux, O. Le Maıˆtre, and J.-M. Martinez. Polynomial chaos expansion for sensitivity analysis. *Reliability Engineering & System Safety*, 94(7):1161–1172, 2009. doi: 10.1016/j.ress.2008.10.008.

[48] M. G. Fernández-Godino. Review of multi-fidelity models. 2016. doi: 10.48550/arXiv.1609.07196.

[49] L. Huang, Z. Gao, and D. Zhang. Research on multi-fidelity aerodynamic optimization methods. *Chinese Journal of Aeronautics*, 26(2):279–286, 2013. doi: 10.1016/j.cja.2013.02.004.

[50] J. Li and M. Zhang. Data-based approach for wing shape design optimization. *Aerospace Science and Technology*, 112:106639, 2021. doi: 10.1016/j.ast.2021.106639.

[51] J. Li, M. A. Bouhlel, and J. R. Martins. Data-based approach for fast airfoil analysis and optimization. *AIAA Journal*, 57(2):581–596, 2019. doi: 10.2514/1.j057129.

[52] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. doi: 10.1007/s10710-017-9314-z.

[53] G. Sun and S. Wang. A review of the artificial neural network surrogate modeling in aerodynamic design. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 233(16):5863–5872, 2019. doi: 10.1177/0954410019864485.

[54] E. R. Lalonde, B. Vischschraper, G. Bitsuamlak, and K. Dai. Comparison of neural network types and architectures for generating a surrogate aerodynamic wind turbine blade model. *Journal of Wind Engineering and Industrial Aerodynamics*, 216:104696, 2021. doi: 10.1016/j.jweia.2021.104696.

[55] X. Du, P. He, and J. R. Martins. Rapid airfoil design optimization via neural networks-based parameterization and surrogate modeling. *Aerospace Science and Technology*, 113:106701, 2021. doi: 10.1016/j.ast.2021.106701.

[56] S. Bhatnagar, Y. Afshar, S. Pan, K. Duraisamy, and S. Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64:525–545, 2019. doi: 10.1007/s00466-019-01740-0.

[57] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996. doi: 10.1613/jair.301.

[58] Y. Zhang, W. J. Sung, and D. N. Mavris. Application of convolutional neural network to predict airfoil lift coefficient. In *2018 AIAA/ASCE/AHS/ASC structures, structural dynamics, and materials conference*, page 1903, January 2018. doi: 10.48550/arXiv.1712.10082.

[59] H. Moin, H. Z. I. Khan, S. Mobeen, and J. Riaz. Airfoil's aerodynamic coefficients prediction using artificial neural network. In *2022 19th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pages 175–182, September 2021. doi: 10.48550/arXiv.2109.12149.

[60] T.-t. Zhang, Z.-g. Wang, W. Huang, and L. Yan. A review of parametric approaches specific to aerodynamic design process. *Acta Astronautica*, 145:319–331, 2018. doi: 10.1016/j.actaastro.2018.02.011.

[61] M. S. Selig. *UIUC airfoil data site*. Department of Aeronautical and Astronautical Engineering University of Illinois at Urbana-Champaign, 1996. URL `https://m-selig.ae.illinois.edu/ads/coord_database.html`. Accessed: 21/10/2023.

[62] J. Li, M. Zhang, C. M. J. Tay, N. Liu, Y. Cui, S. C. Chew, and B. C. Khoo. Low-reynolds-number airfoil design optimization using deep-learning-based tailored airfoil modes. *Aerospace Science and Technology*, 121:107309, 2022. doi: 10.1016/j.ast.2021.107309.

[63] W. Chen, K. Chiu, and M. D. Fuge. Airfoil design parameterization and optimization using bézier generative adversarial networks. *AIAA journal*, 58(11):4723–4735, 2020. doi: 10.2514/1.J059317.

[64] G. Achour, W. J. Sung, O. J. Pinon-Fischer, and D. N. Mavris. Development of a conditional generative adversarial network for airfoil shape optimization. In *AIAA Scitech 2020 Forum*, page 2261, January 2020. doi: 10.2514/6.2020-2261.

[65] A. Shahrokhi and A. Jahangirian. Airfoil shape parameterization for optimum navier–stokes design with genetic algorithm. *Aerospace science and technology*, 11(6):443–450, 2007. doi: 10.1016/j.ast.2007.04.004.

[66] B. M. Kulfan. Universal parametric geometry representation method. *Journal of aircraft*, 45(1): 142–158, 2008. doi: 10.2514/1.29958.

[67] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 151–160, August 1986. doi: 10.1145/15886.15903.

[68] W. M. Hsu, J. F. Hughes, and H. Kaufman. Direct manipulation of free-form deformations. *ACM Siggraph Computer Graphics*, 26(2):177–184, 1992. doi: 10.1145/142920.134036.

[69] W. Zhang, L. Zhao, T. Gao, and S. Cai. Topology optimization with closed b-splines and boolean operations. *Computer Methods in Applied Mechanics and Engineering*, 315:652–670, 2017. doi: 10.1016/j.cma.2016.11.015.

[70] R. Derksen and T. Rogalsky. Bezier-parsec: An optimized aerofoil parameterization for design. *Advances in engineering software*, 41(7-8):923–930, 2010. doi: 10.1016/j.advengsoft.2010.05.002.

[71] R. M. Hicks and P. A. Henne. Wing design by numerical optimization. *Journal of Aircraft*, 15(7): 407–412, 1978. doi: 10.2514/3.58379.

[72] I. H. Abbott and A. E. Von Doenhoff. *Theory of wing sections: including a summary of airfoil data*. McGraw-Hill Book Company, 1st edition, 1949. ISBN 9780486605869.

[73] D. A. Masters, N. J. Taylor, T. Rendall, C. B. Allen, and D. J. Poole. Geometric comparison of aerofoil shape parameterization methods. *AIAA journal*, 55(5):1575–1589, 2017. doi: 10.2514/1.J054943.

[74] C. Lee, D. Koo, and D. W. Zingg. Comparison of b-spline surface and free-form deformation geometry control for aerodynamic optimization. *AIAA Journal*, 55(1):228–240, 2017. doi: 10.2514/1.J055102.

[75] N. P. Salunke, R. Juned Ahamad, and S. Channiwala. Airfoil parameterization techniques: A review. *American Journal of Mechanical Engineering*, 2(4):99–102, 2014. doi: 10.12691/ajme-2-4-1.

[76] V. Braibant and C. Fleury. Shape optimal design using b-splines. *Computer methods in applied mechanics and engineering*, 44(3):247–267, 1984. doi: 10.1016/0045-7825(84)90132-4.

[77] M. Ceze, M. Hayashi, and E. Volpe. A study of the cst parameterization characteristics. In *27th AIAA applied aerodynamics conference*, page 3767, June 2009. doi: 10.2514/6.2009-3767.

[78] F. Zhu and N. Qin. Intuitive class/shape function parameterization for airfoils. *AIAA journal*, 52(1): 17–25, 2014. doi: 10.2514/1.J052610.

[79] R. Duvigneau. *Adaptive parameterization using free-form deformation for aerodynamic shape optimization*. PhD thesis, INRIA, 2006. URL `https://inria.hal.science/inria-00085058v2`. Accessed: 21/10/2023.

[80] M. Zhang, N. Bartoli, A. Jungo, W. Lammen, E. Baalbergen, and M. Voskuijl. Enhancing the handling qualities analysis by collaborative aerodynamics surrogate modelling and aero-data fusion. *Progress in Aerospace Sciences*, 119:100647, 2020. doi: 10.1016/j.paerosci.2020.100647.

[81] J. Li, M. Zhang, J. R. Martins, and C. Shu. Efficient aerodynamic shape optimization with deep-learning-based geometric filtering. *AIAA journal*, 58(10):4243–4259, 2020. doi: 10.2514/1.J059254.

[82] J. Li and M. Zhang. Adjoint-free aerodynamic shape optimization of the common research model wing. *AIAA Journal*, 59(6):1990–2000, 2021. doi: 10.2514/1.J059921.

[83] J. M. M. Júnior, G. L. Halila, Y. Kim, T. Khamvilai, and K. G. Vamvoudakis. Intelligent data-driven aerodynamic analysis and optimization of morphing configurations. *Aerospace Science and Technology*, 121:107388, 2022. doi: 10.1016/j.ast.2022.107388.

[84] J. Li and M. Zhang. On deep-learning-based geometric filtering in aerodynamic shape optimization. *Aerospace Science and Technology*, 112:106603, 2021. doi: 10.1016/j.ast.2021.106603.

[85] R. Bro and A. K. Smilde. Principal component analysis. *Analytical methods*, 6(9):2812–2831, 2014. doi: 10.1039/C3AY41907J.

[86] D. J. Toal, N. W. Bressloff, A. J. Keane, and C. M. Holden. Geometric filtration using proper orthogonal decomposition for aerodynamic design optimization. *AIAA journal*, 48(5):916–928, 2010. doi: 10.2514/1.41420.

[87] H. Abdi. Singular value decomposition (svd) and generalized singular value decomposition. *Encyclopedia of measurement and statistics*, 907:912, 2007. doi: 10.4135/9781412952644.

[88] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. doi: 10.48550/arXiv.1406.2661.

[89] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer, 1st edition, 1999. doi: 10.1007/b98874.

[90] S. S. Rao. *Engineering optimization: theory and practice*. John Wiley & Sons, 2019. doi: 10.1002/9781119454816.

[91] D. Kraft. A software package for sequential quadratic programming. *Forschungsbericht-Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988. URL `https://yetanothermathprogrammingconsultant.blogspot.com/2022/02/slsqp-original-paper.html`. Accessed: 29/10/2023.

[92] S.-i. Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5): 185–196, 1993. doi: 10.1016/0925-2312(93)90006-O.

[93] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations*, May 2015. doi: 10.48550/arXiv.1412.6980.

[94] Cambridge dictionary. URL `https://dictionary.cambridge.org/`. Accessed: 24/09/2023.

[95] C. Lingling, L. Qi, G. Feng, D. Xintian, H. Yuqing, and D. Yangchen. Design, modeling, and control of morphing aircraft: A review. *Chinese Journal of Aeronautics*, 35(5):220–246, 2022. doi: 10.1016/j.cja.2021.09.013.

[96] Ö. Cevdet, E. ÖZBEK, and S. Ekici. A review on applications and effects of morphing wing technology on uavs. *International Journal of Aviation Science and Technology*, 1(01):30–40, 2021. doi: 10.23890/IJAST.vm01is01.0105.

[97] S. Barbarino, O. Bilgen, R. M. Ajaj, M. I. Friswell, and D. J. Inman. A review of morphing aircraft. *Journal of intelligent material systems and structures*, 22(9):823–877, 2011. doi: 10.1177/1045389X11414084.

[98] R. Meyer, W. Hage, D. W. Bechert, M. Schatz, T. Knacke, and F. Thiele. Separation control by self-activated movable flaps. *AIAA journal*, 45(1):191–199, 2007. doi: 10.2514/6.2004-1243.

[99] Z. Zhang, A. De Gaspari, S. Ricci, C. Song, and C. Yang. Gradient-based aerodynamic optimization of an airfoil with morphing leading and trailing edges. *Applied Sciences*, 11(4):1929, 2021. doi: 10.3390/app11041929.

[100] Z. Lyu and J. R. Martins. Aerodynamic shape optimization of an adaptive morphing trailing-edge wing. *Journal of Aircraft*, 52(6):1951–1970, 2015. doi: 10.2514/1.C033116.

[101] F. Sturm, E. Wehrle, and M. Hornung. Simplified modeling of topology-optimized compliant mechanism for multi-fidelity integration in morphing wings. In *AIAA AVIATION 2023 Forum*, page 4022, June 2023. doi: 10.2514/6.2023-4022.

[102] K. Soneda, T. Yokozeki, T. Imamura, and N. Tsushima. Multi-fidelity aeroelastic simulation of a morphing wing trailing edge. In *AIAA Scitech 2021 Forum*, page 0953, January 2021. doi: 10.2514/6.2021-0953.

[103] A. K. Kancharla and D. Roy Mahapatra. Aerodynamic pressure variation over sma wire integrated morphing aerofoil. In *49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 16th AIAA/ASME/AHS Adaptive Structures Conference, 10th*

AIAA Non-Deterministic Approaches Conference, 9th AIAA Gossamer Spacecraft Forum, 4th AIAA Multidisciplinary Design Optimization Specialists Conference, page 2044, April 2008. doi: 10.2514/6.2008-2044.

[104] S. Longtin Martel, M. Bashir, R. M. Botez, and T. Wong. A pareto multi-objective optimization of a camber morphing airfoil using non-dominated sorting genetic algorithm. In *AIAA SCITECH 2023 Forum*, page 1583, January 2023. doi: 10.2514/6.2023-1583.

[105] S. Kim, I. Kim, and D. You. Multi-condition multi-objective optimization using deep reinforcement learning. *Journal of Computational Physics*, 462:111263, 2022. doi: 10.1016/j.jcp.2022.111263.

[106] H. Namgoong, W. A. Crossley, and A. S. Lyrintzis. Aerodynamic optimization of a morphing airfoil using energy as an objective. *AIAA journal*, 45(9):2113–2124, 2007. doi: 10.2514/1.24355.

[107] D. Lee, L. F. Gonzalez, J. Periaux, and G. Bugeda. Multi-objective design optimization of morphing uav aerofoil/wing using hybridised moga. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8, June 2012. doi: 10.1109/CEC.2012.6256429.

[108] J. Fincham and M. Friswell. Aerodynamic optimisation of a camber morphing aerofoil. *Aerospace Science and technology*, 43:245–255, 2015. doi: 10.1016/j.ast.2015.02.023.

[109] T. M. Mitchell. *Machine learning*, volume 1. McGraw Hill, 2007.

[110] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539.

[111] L. E. Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009. doi: http://dx.doi.org/10.4249/scholarpedia.1883.

[112] S. Suthaharan and S. Suthaharan. Support vector machine. *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*, pages 207–235, 2016. doi: 10.1007/978-1-4899-7641-3_9.

[113] J. E. Van Engelen and H. H. Hoos. A survey on semi-supervised learning. *Machine learning*, 109 (2):373–440, 2020. doi: 10.1007/s10994-019-05855-6.

[114] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, June 2016. doi: 10.1109/CVPR.2016.90.

[115] J. B. Heaton, N. G. Polson, and J. H. Witte. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1):3–12, 2017. doi: 10.1002/asmb.2209.

[116] Y. Pan, Y. Yang, and W. Li. A deep learning trained by genetic algorithm to improve the efficiency of path planning for data collection with multi-uav. *Ieee Access*, 9:7994–8005, 2021. doi: 10.1109/ACCESS.2021.3049892.

[117] C. Barata, M. Ruela, M. Francisco, T. Mendonça, and J. S. Marques. Two systems for the detection of melanomas in dermoscopy images using texture and color features. *IEEE systems Journal*, 8 (3):965–979, 2013. doi: 10.1109/JSYST.2013.2271540.

[118] F. Murtagh. Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5-6): 183–197, 1991. doi: 10.1016/0925-2312(91)90023-5.

[119] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017. doi: 10.1109/JPROC.2017.2761740.

[120] S. Sharma, S. Sharma, and A. Athaiya. Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316, 2017. ISSN 2455-2143.

[121] A. D. Rasamoelina, F. Adjailia, and P. Sinčák. A review of activation function for artificial neural network. In *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, pages 281–286. IEEE, January 2020. doi: 10.1109/SAMI48414.2020.9108717.

[122] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis. Dying relu and initialization: Theory and numerical examples. *arXiv:1903.06733*, 2019. doi: 10.4208/cicp.OA-2020-0165.

[123] R. Rojas and R. Rojas. The backpropagation algorithm. *Neural networks: a systematic introduction*, pages 149–182, 1996. doi: 10.1007/978-3-642-61068-4_7.

[124] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv:1609.04747*, 2016. doi: 10.48550/arXiv.1609.04747.

[125] L. Bottou. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade: Second Edition*, pages 421–436. Springer, 2012. doi: 10.1007/978-3-642-35289-8_2.

[126] L. Bottou et al. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8):12, 1991. URL `https://api.semanticscholar.org/CorpusID:12410481`. Accessed: 29/10/2023.

[127] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv:1609.04836*, 2016. doi: 10.48550/arXiv.1609.04836.

[128] J. C. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. URL `https://typeset.io/papers/adaptive-subgradient-methods-for-online-learning-and-1nnwnbmnxy`. Accessed: 29/10/2023.

[129] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1): 145–151, 1999. doi: 10.1016/S0893-6080(98)00116-6.

[130] Y. Nesterov. A method for solving the convex programming problem with convergence rate o($1/k^2$). *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983. URL https://api.semanticscholar.org/CorpusID:145918791. Accessed: 29/10/2023.

[131] Q. Wang, Y. Ma, K. Zhao, and Y. Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, pages 1–26, 2020. doi: 10.1007/s40745-020-00253-5.

[132] D. McLean. *Understanding aerodynamics: arguing from the real physics*. John Wiley & Sons, 2012. doi: 10.1002/9781118454190.

[133] K. S. Patel, S. Patel, U. B. Patel, and A. Ahuja. Cfd analysis of an aerofoil. *International Journal of Engineering Research and*, 3:154–158, 2014. URL https://api.semanticscholar.org/CorpusID:111152794. Accessed: 29/10/2023.

[134] J. D. Anderson. *Fundamentals of aerodynamics*. McGraw-Hill New York, 6th edition, 2009.

[135] J. D. Anderson. *Modern compressible flow: with historical perspective*, volume 12. McGraw-Hill New York, 1990.

[136] H. Glauert. Aerodynamic theory. *The Aeronautical Journal*, 34(233):409–414, 1930. doi: 10.1017/S0368393100114397.

[137] G. Ben-Dor, O. Igra, and T. Elperin. *Handbook of shock waves, three volume set*. Elsevier, 1st edition, 2000.

[138] J. D. Anderson and J. Wendt. *Computational fluid dynamics*, volume 206. Springer, 1995. doi: 10.1007/978-3-662-11350-9.

[139] H. K. Versteeg and W. Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.

[140] C. Hirsch. *Numerical computation of internal and external flows: The fundamentals of computational fluid dynamics*. Elsevier, 2007. doi: 10.1016/B978-0-7506-6594-0.X5037-1.

[141] C. D. Harris. Nasa supercritical airfoils: A matrix of family-related airfoils. Technical report, 1990. URL https://ntrs.nasa.gov/api/citations/19900007394/downloads/19900007394.pdf. Accessed: 23/10/2023.

[142] G. K. W. Kenway, G. J. Kennedy, and J. R. R. A. Martins. A CAD-free approach to high-fidelity aerostructural optimization. September 2010. doi: 10.2514/6.2010-9231. AIAA 2010-9231.

[143] A. B. Lambe and J. R. R. A. Martins. Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes. *Structural and Multidisciplinary Optimization*, 46:273–284, 2012. doi: 10.1007/s00158-012-0763-y.

[144] N. R. Secco, G. K. Kenway, P. He, C. Mader, and J. R. Martins. Efficient mesh generation and deformation for aerodynamic shape optimization. *AIAA Journal*, 59(4):1151–1168, 2021. doi: 10.2514/1.J059491.

[145] W. M. Chan and J. L. Steger. Enhancements of a three-dimensional hyperbolic grid generation scheme. *Applied Mathematics and Computation*, 51(2-3):181–205, 1992. doi: 10.1016/0096-3003(92)90073-A.

[146] C. A. Mader, G. K. W. Kenway, A. Yildirim, and J. R. R. A. Martins. ADflow—an open-source computational fluid dynamics solver for aerodynamic and multidisciplinary optimization. *Journal of Aerospace Information Systems*, 2020. doi: 10.2514/1.I010796.

[147] D. Poirier, S. Allmaras, D. McCarthy, M. Smith, and F. Enomoto. The cgns system. In *29th AIAA, Fluid Dynamics Conference*, page 3007, June 1998. doi: 10.2514/6.1998-3007.

[148] J. Li, S. He, and J. R. Martins. Data-driven constraint approach to ensure low-speed performance in transonic aerodynamic shape optimization. *Aerospace Science and Technology*, 92:536–550, 2019. doi: 10.1016/j.ast.2019.06.008.

[149] P. Spalart and S. Allmaras. A one-equation turbulence model for aerodynamic flows. In *30th aerospace sciences meeting and exhibit*, page 439, January 1992. doi: 10.2514/6.1992-439.

[150] L. Prandtl and H. Schlichting. *The Resistance Law for Rough Plates*. David W. Taylor Model Basin, Navy Department, Washington DC, Translation 258, 1955.

[151] P. J. Roache. *Verification and validation in computational science and engineering*, volume 895. Hermosa Albuquerque, NM, 1998.

[152] Z. Zlatev, I. Dimov, I. Faragó, and Á. Havasi. *Richardson extrapolation: Practical aspects and applications*, volume 2. Walter de Gruyter GmbH & Co KG, 2017.

[153] H. M. Hajdik, A. Yildirim, N. Wu, B. J. Brelje, S. Seraj, M. Mangano, J. L. Anibal, E. Jonsson, E. J. Adler, C. A. Mader, G. K. W. Kenway, and J. R. R. A. Martins. pyGeo: A geometry package for multidisciplinary design optimization. *Journal of Open Source Software*, 8(87):5319, 2023. doi: 10.21105/joss.05319.

[154] S. Pucciarelli, F. Lau, and A. Suleman. An efficient numerical modelling for aerodynamic shape optimisation. In *Proceesings of the 2nd Thematic Conference on Multidisciplinary Design Optimization of Aerospace Systems*, pages 565–578, July 2023. URL `https://aerobest2023.idmec.tecnico.ulisboa.pt/publications/`. Accessed: 29/10/2023.

[155] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. doi: 10.1109/4235.996017.

[156] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso. Su2: An open-source suite for multiphysics simulation and design. *AIAA Journal*, 54(3):828–846, 2016. doi: 10.2514/1.J053813.

[157] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002. doi: 10.1109/4235.996017.

[158] Z. Michalewicz, D. Dasgupta, R. G. Le Riche, and M. Schoenauer. Evolutionary algorithms for constrained engineering problems. *Computers & Industrial Engineering*, 30(4):851–870, 1996. doi: 10.1016/0360-8352(96)00037-X.

[159] Ö. Yeniay. Penalty function methods for constrained optimization with genetic algorithms. *Mathematical and computational Applications*, 10(1):45–56, 2005. doi: 10.3390/mca10010045.

[160] J. Blank and K. Deb. Pymoo: Multi-objective optimization in python. *IEEE Access*, 8:89497–89509, 2020. doi: 10.1109/ACCESS.2020.2990567.

[161] J. Blank and K. Deb. Multi objective optimisation in python: *pymoo*, 2023. url: `https://pymoo.org/index.html`. Accessed: 21/10/2023.

[162] K. Deb, K. Sindhya, and T. Okabe. Self-adaptive simulated binary crossover for real-parameter optimization. In *Proceedings of the 9th annual conference on genetic and evolutionary computation*, pages 1187–1194, July 2007. doi: 10.1145/1276958.1277190.

[163] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000. doi: 10.1080/00401706.2000.10485979.