CRANFIELD UNIVERSITY

AARON FERIA ALANIS

INVERSE DESIGN OF TRANSONIC/SUPERSONIC AEROFOILS BASED ON DEEP NEURAL NETWORKS

SCHOOL OF AEROSPACE, TRANSPORT AND MANUFACTURING
Computational Engineering Sciences

MSc by Research
Academic Year: 2018–2019

Supervisor: Dr Antonios F. Antoniadis
December 2019

CRANFIELD UNIVERSITY

SCHOOL OF AEROSPACE, TRANSPORT AND MANUFACTURING
Computational Engineering Sciences

MSc by Research

Academic Year: 2018–2019

AARON FERIA ALANIS

Inverse Design of Transonic/Supersonic Aerofoils Based on Deep Neural Networks

Supervisor: Dr Antonios F. Antoniadis
December 2019

This thesis is submitted in partial fulfilment of the requirements for the degree of MSc by Research.

Abstract

Transonic and supersonic aerofoil inverse design for different flight conditions is carried out using Deep Neural Networks (DNN). DNN are combined with a comprehensive and complete database of aerodynamic data and aerofoil geometry parameters to form the pillars of a surrogate inverse aerodynamic design tool. The framework of this research starts with the aerofoil parameterisation. The Class/Shape Transformation functions (CST) was selected for the parameterisation process due to its high accuracy and flexibility when describing complex shapes. An automated mesh technique is created and implemented to discretise the flow domain. The aerodynamic computations are performed for 395 aerofoils. Spatial discretisation is accomplished with the Jameson-Schmidt-Turkel (JST) scheme and convergence is reached by the backward Euler implicit numerical scheme. Data are collected and managed with the CST parameters for all aerofoils and their respective aerodynamic characteristics from the CFD solver. The Deep Neural Network is then trained, validated using cross-validation and evaluated against CFD data. An extensive investigation of the effect from different DNN configurations takes place in this research. Within this thesis, different case studies are presented for different numbers of design objectives. For the inverse design process the NACA 66-206 aerofoil was selected as the baseline aerofoil, to reduce the aerodynamic drag coefficient while maintaining or improving the lift coefficient, to obtain a superior lift/drag ratio compared with the baseline aerofoil. The framework of this thesis have proved to output aerofoil designs with an improved lift/drag ratio in comparison with the baseline aerofoil.

Keywords

Deep Neural Networks, Class function/Shape function Transformation, Computational Fluid Dynamics, Euler equations, Transonic/Supersonic flight.

Contents

List of Figures

List of Tables

List of Abbreviations

ANN	Artificial Neural Networks
DNN	Deep Neural Network
CNN)	Convolutional Neural Network
SOM	Self Organising Map
FFBP	Feed-Forward Back Propagation
CFD	Computational Fluid Dynamics
RANS	Reynolds-Averaged NavierStokes
DVM	Discrete Vortex Method
HPC	High Performance Computer
NACA	National Advisory Committee for Aeronautics
NASA	National Aeronautics and Space Administration
SSBJ	Supersonic Business Jets
GA	Genetic Algorithms
NSGA-II	Non-dominated Sorting Genetic Algorithm
MOGA	Multi-Objective Genetic Algorithm
MIGA	Multi-Island Genetic Algorithm
EA	Evolutionary Algorithms
PSO	Particle Swarm Optimisations
ALPSO	Augmented Lagrange multiplier Particle Swarm Optimiser
CST	Class/Shape Transformation
PDEs	Partial Differential Equations

MSE	Mean Square Error
EIIDO	Experience-Independent Inverse Design Optimisation
CAD	Computer-Aided Design
NURBS	Non-Uniform Rational Basis-Splines

Acknowledgements

I would candidly like to thank my supervisor Dr. Antonios F. Antoniadis for his valuable help and advice during this project. I would also like to thank for all the support and facilities I received from the staff of Cranfield University and especially from the School of Aerospace, Transport and Manufacturing. I would like to thank the Mexican National Council for Science and Technology CONACYT for providing the necessary funding for me to undertake this project. All my office mates that accompanied me during the year merit to be thanked. And finally, I would like to thank God, my parents Yuri Feria and Bellanira Alanis, my sisters Bellanira Feria and Paulina Irela Feria, and all my friends for their unconditional and perpetual support, shelter, and love that made me who I am today.

Chapter 1

Introduction

Supersonic aircraft have seen plenty of interest and accelerated development progress since the first flight of the Bell X-1 in 1947. Although, the usage of supersonic aircraft was mostly restricted to military applications during the 20th century, some projects advanced the progress of supersonic aircraft for civilian transportation during the 1960s. Different research projects pioneered the application of supersonic flight for civilian transport, such as the Russian Tu-144, and the retired French/British Concorde which was in service until 2003.

Since the last flight of the Concord in 2003, there have not been any supersonic aircraft for civilian transportation. However, the present demand of faster and more efficient air transportation systems in a globalised world, where customers such as heads of state and executives attributes extensive value to time, has resulted in dedicated research in supersonic flight for civilian applications. Recent research on the matter is conducted by organisations and companies around the world working on Supersonic Business Jet (SSBJ) concepts. Among the different SSBJ concepts, the more current, such as the Spike S-512, Aerion AS-2, and HyperMach SonicStar, are summarised by Sun et al. [1] from Cranfield University.

One of the main challenges for transonic and supersonic flight is the reduction of the supersonic boom and the improvement of the aerodynamic efficiency at transonic

and supersonic speeds. The aerodynamic performance of an aircraft is directly related with the lift-to-drag ratio (L/D). This is of great importance, due to its impact on the overall performance, such is the case of the range capabilities of the aircraft and the power consumption requirements, which are sensitive to the lift-to-drag ratio. A more efficient lift-to-drag ratio not only benefits economically the operation of supersonic aircraft, but also reduces the power consumption of an aircraft, which is directly linked to a reduction of the fuel utilisation, and to a reduction on the environmental pollution [2].

The main contributors to the lift force in the aircraft structure are the wings. Thus, the aerodynamic performance is directly linked to the design of the aircraft wings and their different configurations. In essence, the lift and drag forces are generated due only due two sources: the pressure distribution and the shear stresses distribution over the wing [3]. These basic factors are a function of the wing's cross section geometry known as the aerofoil. Thereby, the efficiency is dependent on the design of the aerofoil. The importance of the aerofoil in flight efficiency analysis is not to be underestimated, the pressure distribution around it affects the stall speed, cruise speed, thrust required, turning performance and other characteristics of aircraft flight dynamics.

1.1 The influence of the aerofoil shape in transonic and supersonic aerodynamics

The lift (L) and drag (D) aerodynamic forces are directly linked to the behaviour of the flow around a body. The aerodynamic forces of a body subjected to a flow are owing to the pressure distribution on the body's surface, and the shear stress distribution over the body [3]. The pressure acts normal to the aerofoil surface and the shear stresses acts tangentially to the body. The lift is the component of the resultant force on the body that is normal to the flow direction and the drag is the component in the flow direction. The pressure distribution around the aerofoil, and the aerodynamic forces by extension, are primarily functions of the mean line shape [4]. The mean line, or camber mean line, is

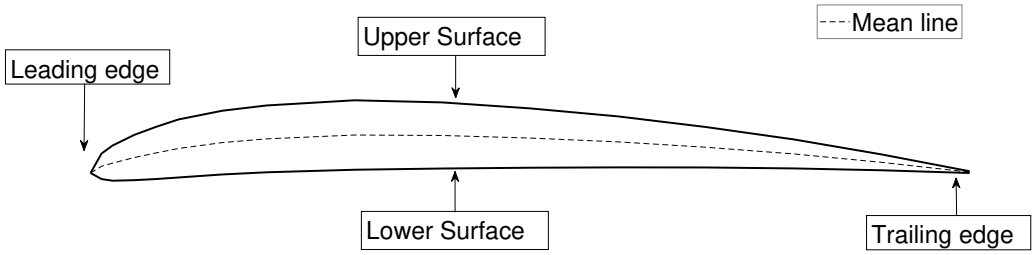

Figure 1.1: Aerofoil geometry components.

the line created by the points located halfway between the upper and lower surfaces of the aerofoil, as shown in Figure 1.1.

The aerofoil physical configuration, such as the shape of the mean line and the aerofoil's thickness, plays a crucial role on the aerodynamic aspects of the flow field around it, especially when flying at transonic and supersonic speeds. As an aeroplane reaches Mach 1, the flow can be accelerated up to beyond the speed of sound over certain parts of the aircraft, as for example, at the point of greatest camber over the surface of an aerofoil, a shock wave may form at this high-velocity point [5].

A shock wave is a narrow region where a drastic change in the flow properties takes place. Shock waves are compressibility-related discontinuities of the flow field that are related to the Mach number. Across the shock wave there is a discontinuous increase in the flow pressure, density, temperature, and entropy; along with a decrease in the flow velocity and Mach number. The generation of shock waves is directly related to the increase in the drag force, this extra amount of drag is known as the wave drag. The wave drag is inherently linked to the increase in entropy caused by the compressibility-related discontinuities and consequently loss of energy. The strength of a shock wave is a function of the ratio of the properties change across the shock wave [3].

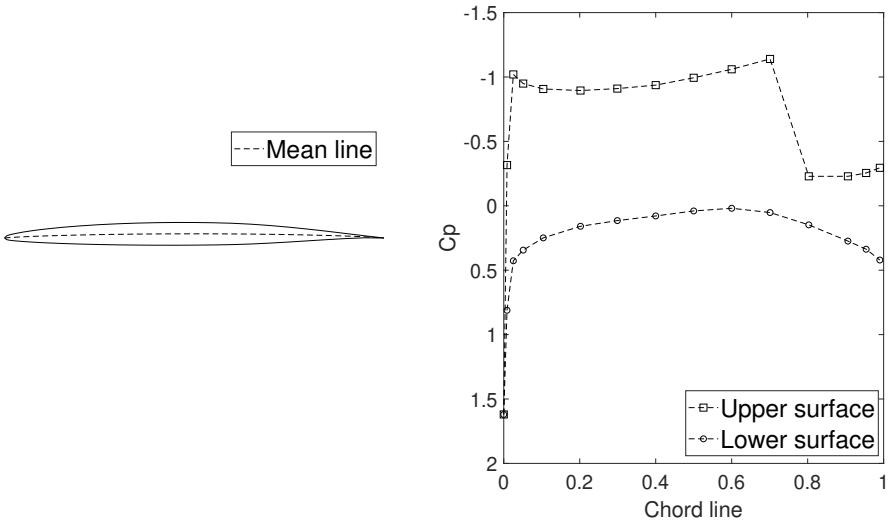

Figure 1.2: NACA 66-206 shape influence on the pressure coefficient distribution at Mach 0.8 and AoA of 2°.

When conventional aerofoils with rounded upper surface are subjected to flow velocities near the speed of sound, the air flowing on the top surface is accelerated and becomes supersonic. This accelerated flow creates a shock wave on the upper surface of the aerofoil, separating the boundary layer from the surface and creating turbulence. This phenomenon occurs at a certain Mach number, which varies with different types of aerofoils, and which is known as the critical Mach number. It is in the interest of the designer to generate aerofoil shape designs that have a high critical Mach number. Aerofoils designed to efficiently fly at transonic and supersonic regimes are usually flatter on the top surface and, in the case of transonic aerofoils, rounded on the bottom surface and have a substantial aft camber to delay and reduce the shock waves. Also, a general trait of aerofoils that have high critical Mach numbers is a slimness shape [5].

A visual example of the influence of the aerofoil shape configuration in high-speed flight is portrayed in Figures 1.2 and 1.3. Here the plots of the NACA 66-206 and the NACA 0012 and their respective pressure distribution at Mach 0.8 and an angle of attack (AoA) of of 2° are presented. Note that the flow is accelerates continuously on the upper surface of the NACA 0012 into the discontinuous increase of pressure at a point about

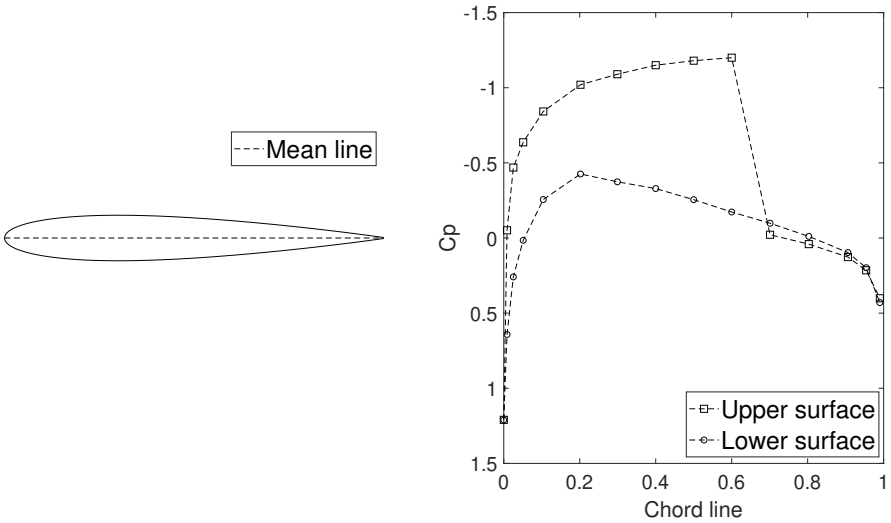

Figure 1.3: NACA 0012 shape influence on the pressure coefficient distribution at Mach 0.8 and AoA of 2°.

60% of the chord line. This as a result of a shock wave generated at that point. In contrast, the increase in pressure of the NACA 66-206 is much smaller, and it is located at a point about 70% of the chord line. The shock wave on the NACA 66-206 is much weaker than the shock generated in the NACA 0012, which decreases the drag force generated by the shock. Also, the pressure distribution over the NACA 66-206 is more uniform across the chord line, providing more lift force. These improvements of the aerofoil shape to increase the efficiency at high-speeds allows the aircraft to fly faster with less power required, which is related with a reduction in the fuel consumption.

1.2 Motivation From a Historical Standpoint

A brief historical review of the developments on transonic and supersonic aerofoil shape design, along with the motivation and importance of generating efficient aerofoil aerodynamic design optimisation methods, are described in this section. This historical review is by no means complete, however it gives a sense of the incentive of the research on transonic and supersonic flight from an historical point of view.

1.2.1 Early stages in high-speed aerodynamic design and the proliferation of interest in supersonic flows around aerofoils

From the laborious work of building and testing aerofoil models by the Wright brothers [6], through the aerodynamic theories generated by Ludwig Prandtl [7], and the interest of developing efficient aircraft driven by the First World War, to the extensive wind tunnel tests performed by the National Advisory Committee of Aeronautics–NACA (which later became NASA) during the 1920s [8, 9, 10], the attention in aerodynamic design gained momentum in the early 20th century.

Supersonic flows and the formation of shock waves were realised as far back as 1907, at the begining of Ludwig Prandtl's work in supersonic aerodynamics, when Prandtl visualised and photographed shock waves formation inside nozzles on different bodies [3]. The flight speeds of the earliest aircraft of the first two decades of the 20th century, were low enough to have null influence by the compressibility effects on the aircraft's airframe. However, by the end of the First World War, the design of some aircraft engines led to propeller diameters large enough to experiment tip speeds as fast as the speed of sound, leading to thrust loss and an increment in blade drag. This phenomenon was first noticed and analysed by British researchers in 1919 and then examined in further detail by F. W. Caldwell and E. N. Fales from the U.S. Army's Engineering Division. Caldwell and Fales designed the first high-speed wind tunnel to analyse this occurrence [10]. They performed the first wind tunnel experiments using high-speed flow over a static aerofoil. Thus, the awareness of high-speeds during the 1920s was mostly concentrated on engine blades.

During the 1930s, the focus of aerodynamics at high-speeds switched from propeller applications to the implementation on the airframe of the aircraft itself. The aerodynamic design concept for transonic and supersonic regimes for airframe applications was consolidated in Autumn 1935. On September 30th, the fifth Volta Conference was taking place in Rome, Italy. Outstanding aerodynamicists from around the world such as Ludwig Prandtl, Adolf Busemann, and Sir Geoffrey Ingram Taylor among others converged, by invitation only, to discuss the subject of High Velocities in Aviation. Effects

of compressibility at high-speeds was the centre of the discussion. During the fifth Volta conference, Eastman Jacobs presented the results of a series of wind tunnel tests using the Schlieren photography technique for visualisation [11]. These results showed a visible transonic flow field over standard NACA aerofoils and they were presented in public for the first time in history. The fifth Volta Conference marked the genesis for the upcoming proliferation of interest in high-speed aerodynamics applied to aircraft airframes.

By 1940, it was properly acknowledged the discontinuous effects on the pressure across shock waves, which are responsible of the flow separation. During the 1940s, the work by Antonio Ferri (which was interrupted in Guidonia, Italy and then completed at Langley NACA facility at wartime), despite the fact that the validity of the "semi-open" tunnel technique applied by Ferri was unsure, it revealed that about Mach 0.9 there was a notable recovery in lift. These results where then validated in 1947 from wind tunnel experiments at Mach 1, successfully obtaining for the first time the pressure distribution around an aerofoil at sonic speed, and conclusively proving that the shock waves moves to the trailing edge at flow regimes close to Mach 1 [12].

Although that the discontinuities generated by the shock waves at transonic and supersonic speeds were already well known during the 1940s and 1950s, the early stages of aerodynamic design process were still a slow iterative trial and error procedure which involved a lot of experience of designers, and a large amount of wind tunnel experiments and flight tests to determine the aerodynamic characteristics of an aircraft. For instance, and among a high number of cases where wind tunnel experiments were used for early stages in transonic/supersonic aerodynamic design, and enclosed by a research program aimed to the investigation of aerodynamic behaviour in the transonic and low supersonic regimes. The NACA modified the supersonic aircraft Bell X-1's wing. Several wind tunnel experiments were performed to obtain the aerodynamic data that lead to the proposed wing [13]. The need at that time of efficient tools to obtain the characteristics of the flow field around wings and aerofoils along with the rise of the computer era, led to the development of computational algorithms to assist the process of flow analysis, such as

Computational Fluid Dynamic (CFD) codes.

1.2.2 The introduction of CFD in the aerodynamic design process

The introduction of CFD in the field of aerodynamic design was made in the 1960s, and ever since, there have been an accelerated progress in this field. During the 1960s and early 1970s, the computational codes used for simulations at subsonic and supersonic conditions involved linearised fluxes [14]. However, even with the introduction of CFD in the design process, due to the small computational capacity, and the limited fidelity of the computational codes due to their early stages, high amount of wind tunnel experiments and testing was still required in the aerodynamic design process. Such is the case of the Boeing 747, which required more than 15,000 hours of wind tunnel experiments in 8 different locations [15].

During the 1970s and early 1980s, linear supersonic and subsonic flows calculations around complex geometries could be modelled by using panel methods. Thus, by the early 1980s, CFD was already established as a useful tool for aerodynamic design concerning attached flows. Therefore, the implementation of CFD for aerodynamic design was limited to commercial transport applications [16]. The linear constraint was removed during the late 1980s with the introduction of non-linear potential flow codes, along with the Euler-based codes [14].

Furthermore, the computational simulations based on the Navier–Stokes equations started to be used frequently during the 1990s. During this period, the use of CFD for aerodynamic design in parallel with wind tunnel experiments experienced intensive activity. During the first years of the 1990s more than 19,000 CFD runs were made in the aerodynamic design process of the Boeing 777, reducing considerably the amount of wind tunnel experiments in the design process [17]. Then, since the application of CFD in aerodynamic design, computational methods and experimental methods have been used in parallel by aircraft designers for aerodynamic design, and also to improve existing designs at highly non-linear flow regimes such as the transonic and the supersonic.

1.2.3 The need for automatic computational methods for aerodynamic design

Although CFD capabilities to predict highly non-linear flow field properties (such as the case of shock waves generation in supersonic fluxes), have seen substantial improvements since its conception to a high degree of fidelity, CFD by itself does not optimise completely the process of aerodynamic shape design. The use of CFD to analyse many different shape layouts has proved highly valuable, however it still lacks to guarantee the best possible design configuration. Thus, to obtain a optimal design, the main objective of the application of computational simulations should not be just to enhance the capabilities to analyse determined geometries, but to be able to determine automatically the optimum shape configuration for a specific application. Since the genesis of the implementation of CFD for aerospace applications, researchers have been working on the formulation of computational methods that optimises the CFD-based design process.

An additional driver to the pursue of an efficient computational method to optimise the design process of supersonic aircraft, is the fact that the behaviour of the flow over a supersonic aircraft varies through all its flight operating conditions. The regime of the flow changes in general from low-speed take off and landing, to subsonic climb, through transonic acceleration, to supersonic cruise flight. A successful implementation of supersonic aircraft lays in the foundation of the degree of the performance through all the flight stages. Such is the case of the Concorde, which presented an outstanding aerodynamic performance for cruise flight at Mach 2 due to its optimised aerodynamic shape. However, at low-altitude and low-speed it presented a poor performance [18]. Thus, in general, the majority of the optimisation processes related to aerospace engineering are multi-objective and in some cases multi-disciplinary.

Aerodynamic shape design can be addressed by direct optimisation methods and inverse design methods. In the case of direct optimisation methods, optimality can be measured as the degree at which the design problem is accomplished by fulfilling specific targets and requirements. For the case of aerodynamic shape optimisation, these targets

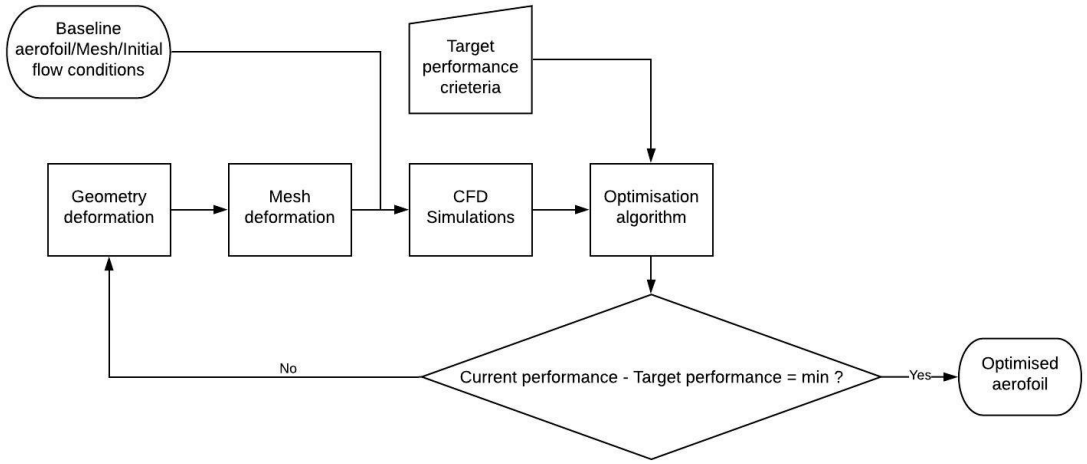

Figure 1.4: A general aerofoil aerodynamic shape design optimisation process.

can address drag minimisation, lift maximisation, an specific pressure distribution, among others. All of these requirements can be seen as a functional subjected to global minimum or maximum search. On the other hand, using inverse design methods it is possible to obtain an optimum aerodynamic shape for a prescribed set of aerodynamic characteristics, such as the target pressure distribution around the aerofoil [19].

Some of the pioneers of computational optimisations methods for shape optimisation were Hicks and Henne in the 1970s [20, 21], their work was based on numerical optimisation strategies. However, the main limitation at that time was the extensive computational burden of iteratively solve the numerical methods, calculating the flow repeatedly. During the same period of time, another pioneer approach strategy was to lay the aerodynamic design in the domain of the control systems theory [22].

A generalised direct aerodynamic design optimisation process is portrayed in Figure 1.4. Aerofoil shape design usually starts with a baseline aerofoil which is defined at the beginning of the process. A proper grid generation technique along with the boundary conditions are then selected based on the flow regime. The flow field is analysed using a CFD code (Euler, Navier–Stokes equations, etc.), then the performance of the objective function (lift maximisation, drag minimisation, L/D maximisation, among others) is evaluated by an optimisation algorithm.

Presently, the optimisation algorithms used for direct aerodynamic design can be driven by both gradient-based algorithms and gradient-free algorithms. Gradient-based search methods, usually employ a process that approximate the objective function over a sub-region of the design space. One of the main characteristics of gradient-based methods is that they rely on the gradient information to iteratively find the direction to a local optimum [23]. In addition, in a non-continuous design space, gradient-based algorithms can get get stuck in a local optimal point. On the other hand, with the improvements of the current computational capabilities, gradient-free heuristic methods have taken a place in the optimisation methods spotlight. By introducing heuristic optimisation methods, it is possible to perform multi-point search in the design space, which allows to find a global optimum of a functional.

The non linearities of the design space usually increases with the number of objectives in a multi-objective optimisation. Heuristic methods offers a robust way to find globally optimal solutions. According with a review on the state-of-the-art in aerodynamic shape optimisation methods by Skinner [24], Particle Swarm Optimisations (PSO) [25] and Genetic Algorithms (GA) [26] are generally the most used of gradient-free aerodynamic design optimisations methods. PSO and GA are also categorised as global optimisation methods due their ability to find a global optima in the design space. Heuristic optimisation methods present an advantage due to their robustness and their search ability in a complex discontinuous topology. However, a drawback of gradient-free algorithms such as GA is their high computational cost.

The drawbacks associated with the computational cost of gradient-based and gradient-free design optimisation methods can be alleviated by the use of surrogate models. The objective of a surrogate model is to establish an approximate function which is computationally cheap to evaluate. Surrogate models are data-based methods, and are often constructed by a set of training data which is not linearly linked with the output response [27]. Reference [27] makes the use of different surrogate models such as: Radial Basis Functions (RBS) [28], Kringin [29], Least Squares [30] and Quadratic Approximations

[30]. Bouhlel et al [27] developed a surrogate modelling toolbox based on the surrogates models mentioned before to enhance numerical gradient-based optimisations. In addition, surrogate models can be coupled as well with gradient-free optimisation methods to reduce the computational cost associated with population based methodologies, such as GA and PSO in aerodynamic optimisations [24].

Nowadays, recent research and development in machine learning and data-driven models, allows the use of complex surrogate methods such as Artificial Neural Networks (ANN) [31] for inverse aerodynamic design. Surrogate models such as neural networks, can be described as a nonlinear data-based process that defines a function which links the design variables to the output responses. The links between the design variables and the outputs of a neural network are a function of the finite data used to train the model [31]. In this sense, ANN can learn the links between the design variables which can be a function of geometry parameters, and the aerodynamic response of such geometries. This makes ANN suitable for inverse aerodynamic design as it will be reviewed in Chapter 2. In addition, in the field of CFD and in the aerospace industry the collection of simulation data is increasing everyday. Aircraft manufacturers can access to their own databases which are composed from experimental and simulated data. This with the increasing capacity of ANN makes favourable the usage of such methods for aerodynamic design.

The popularity of ANN is constantly increasing due to the current improvement in the computational performance and the advantages of improved hardware such as graphical processors. Nowadays, it is possible to obtain more complex neural networks architectures (e. g. with larger number of layers) that enhance the learning capabilities of the network, bringing the possibility to further improve the process and accuracy of aerodynamic inverse design. The improved ANN with a large number of layers is known as a Deep Neural Network (DNN). A mathematical description of ANN and DNN is presented in Chapter 2. It is worth mentioning that, recently another type of neural network called Convolutional Neural Network (CNN) [32] have been widely used, mostly developed for applications in computer vision. CNN differs from ANN and DNN in the way that the

neurons within the structure are connected internally. The number of trainable parameters in CNN are cutted down by sharing weights between the neurons, which makes it more suitable than ANN for image recognition applications, this due to the large datasets used in this kind of studies. However, the study of CNN is out of the scope for this thesis.

Based on the literature review in Chapter 2 it seems that the usage of surrogate models such as ANN is increasing in the field of aerodynamic design, this due to the advances in computational performance and the availability of complete and comprehensive databases. A detailed review and discussion of the currently most used computational optimisation methods is presented in the next chapter of this thesis.

1.3 Methodology

Within this thesis the implementation of an inverse aerodynamic design by the use of Deep Neural Networks (DNN) is carried out. This research is intended to inverse design aerofoils for different flight conditions e. g. different transonic and supersonic flow regime and angles of attack. The process of the implementation of DNN for inverse design in this thesis is portrayed in Figure 1.5.

A database is constructed by 395 aerofoils using the CST parameterisation scheme [33] for every aerofoil included in the database. The aerofoil geometries where obtained from the UIUC Airfoil Database [34] from the University of Illinois. The aerofoils were selected among the complete UIUC database taking into account their geometry and aerodynamic characteristics. Aerofoils designed for low velocities, gliding, and thick aerofoils were left out to construct the database for this research. Thus, only the aerofoils for high velocities were taking into consideration in the databse. An automated mesh process takes place for every aerofoil to perform CFD simulations in order to obtain the aerodynamic lift and drag coefficients which are included in the database. The free stream conditions are specified for the different flight conditions. A DNN is then trained with a dataset composed with the information from the database and the trained DNN is used as a function

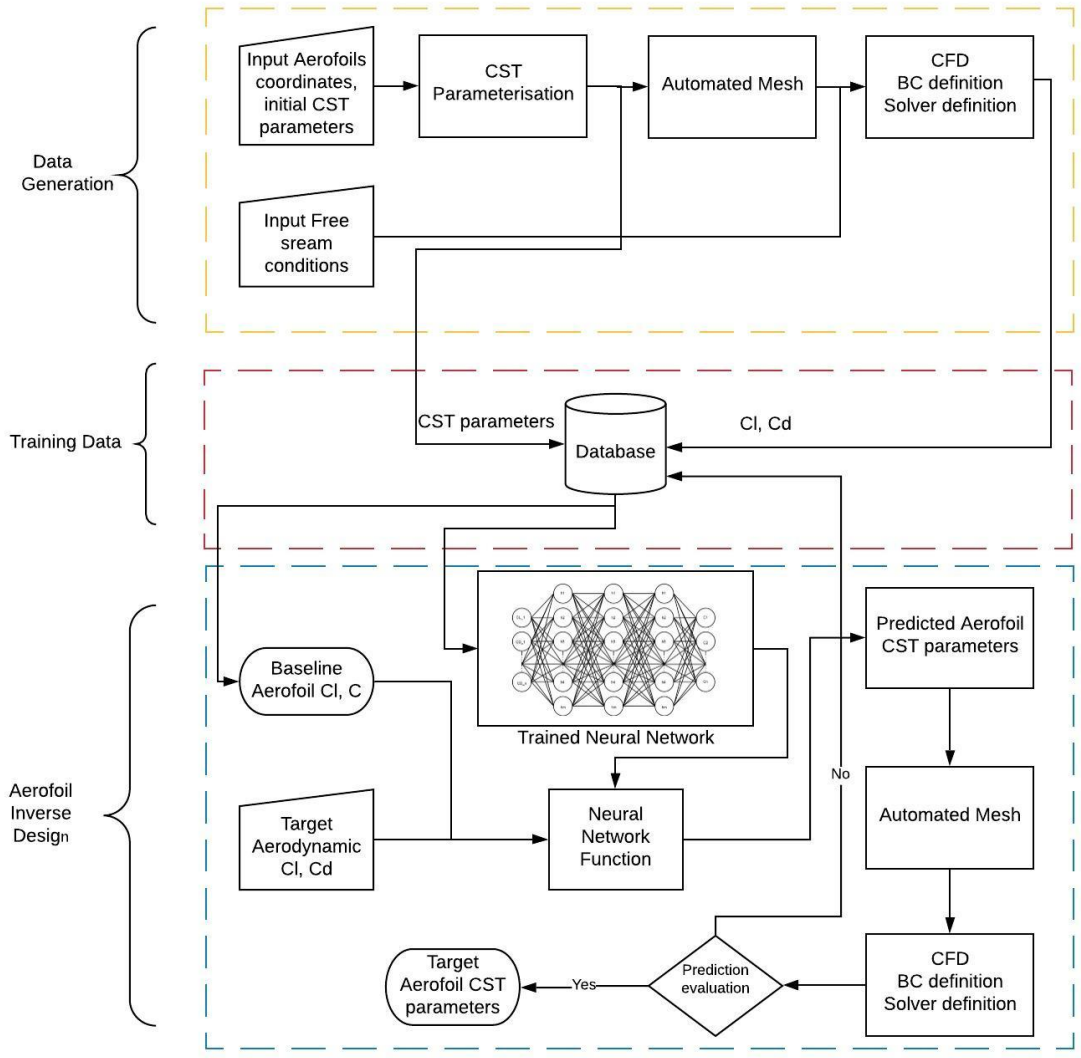

Figure 1.5: Schematic diagram of the aerofoil inverse design process using Deep Neural Networks.

for aerodynamic inverse design. The aerodynamic inverse design use as inputs a baseline aerofoil and the target aerodynamic coefficients. The output of the DNN function are the predicted aerofoil CST parameters which construct the predicted aerofoil shape. This predicted aerofoil shape is then subjected to a CFD simulation in order to evaluate if its aerodynamic coefficients fits the target aerodynamic coefficients. If the aerodynamic coefficients of the predicted aerofoil shape does not fit the target aerodynamic coefficients, this new aerofoil and its aerodynamic coefficients are fed into the database and the network is retrained. This research accomplished to inverse design aerofoils that fits the aerodynamic requests for up to 16 flight conditions with fair accuracy.

It is important to mention that the inverse design framework developed for this research does not include geometrical constraints for the inverse design process, such as maximum or minimum thickness, which could result in structurally infeasible geometries and limits the framework application. As a first approach to demonstrate the capabilities of DNN to predict aerofoils shapes given a set of target aerodynamics (including different flight conditions), and for the sake of maintaining the input parameters to a minimum (only considering the aerodynamic coefficients for different flight conditions as inputs for inverse design), it was decided not to take into consideration geometrical constraints. Geometrical constraints could be contemplated as a next step of this research. However, it is also worth mentioning that as the network was trained with selected aerofoil geometries, it is not expected to obtain predicted geometries outside of the boundaries established within the dataset.

1.4 Aims and Objectives

Section 1.2.3 highlighted the need of efficient computational methods to improve the aerodynamic design process. Although there have been substantial improvement and research on the efficient implementation of numerical optimisation and heuristic optimisation methods for single and multi-objective design problems, most of the literature on the

matter performs design optimisation for a single and in some cases small flight conditions (e.g. different flight velocities and different angles of attack). Population based algorithms such as GA and PSO are more likely to fill up the need of efficient computational methods for aerodynamic design taking into consideration different flight conditions, due to their multi-point search capability. However, even with the implementation of surrogate models to enhance the process required for population based algorithms, they are still in essence an iterative problem which represents high computational burden.

Based on the literature review presented in Chapter 2, surrogate methods have been used mostly to enhance the computational performance of gradient-based and gradient-free optimisation methods. However, nowadays the popularity of machine learning and deep learning techniques such as ANN and DNN is constantly increasing, this due to the computational performance and the advantages of improved hardware, such as graphical processors. As it will be reviewed in Chapter 2, emerging surrogate technologies such as neural networks are increasingly being used for aerodynamic design, due the improvements and developments in these data-driven models. This then begs the questions of whether modern-day aerodynamic design still requires use of deterministic and/or population based optimisation algorithms, and whether current complex surrogate models can achieve to perform aerodynamic design with acceptable accuracy. Furthermore, taking into consideration that the efficient operation of a supersonic aircraft is dependent on the performance through different flight stages, covering a broad range of flight velocities and angles of attack, the question of whether it is possible to perform aerodynamic inverse design taking into consideration different flight conditions is yet to be solved.

Within this thesis, the implementation of DNN for inverse aerodynamic design is carried out. This research is intended to perform aerodynamic inverse design of aerofoils at the transonic and supersonic flow regimes and at different angles of attack. In this first approach the inverse design carried out using a DNN takes into consideration only the lift and drag coefficients of the aerofoils, and the parametric coefficients for the geometry representation. Thus, constraints regarding structural integrity and aerodynamic constraints

such as pitching moment are not considered for this early stage research, which main purpose is to establish the capabilities of DNN to predict aerofoils for an specific target set of aerodynamic lift and drag coefficients only. However, this work accomplishes the inverse design of an aerofoil that fits the aerodynamic targets for up to 16 flight conditions with fair accuracy. To summarise, the aims of this research are as follow:

1. The development of an automated system to perform CFD simulations of 395 aerofoils at different angles of attack at Mach numbers of 0.8, 1.0, 2.0 and 3.0.

2. The evaluation of different numerical geometry representation methods (parameterisation) to mathematically describe the aerofoil shape in the database.

3. The development of a system to automatically create a database including the data corresponding to the CFD simulations and the parametric representation of the aerofoil geometry.

4. Conducting a study of the influence of the DNN architecture configuration on the output prediction accuracy of the model.

5. Develop a framework based on DNN which is able to design aerofoils that fits a specific set of target aerodynamic coefficient for a given set of flight conditions.

1.5 Contribution to scientific knowledge

The contribution to scientific knowledge are summarised as follows:

- The establishment of a complete, comprehensive, and structured database containing 395 aerofoils and their respective aerodynamic coefficients at different angles of attack at Mach numbers of 0.8, 1.0, 2.0 and 3.0.

- The development of an automated 2-dimensional mesh generation code coupled with the CST parameterisation model.

- The development of a novel multi-objective DNN-based design tool which is capable of creating aerofoils for a given set of aerodynamic characteristics.

1.6 List of publications

- Aaron Feria-Alanis, Antonios F. Antoniadis, "Aerodynamic Design Optimisation of Transonic/Supersonic Aerofoils Based on Deep Neural Networks", UK Fluids Conference 2019 at University of Cambridge, Cambridge, United Kingdom, 27-29 August 2019.

- Aaron Feria-Alanis, Antonios F. Antoniadis, "Aerodynamic Inverse Design of Transonic/Supersonic Aerofoils Based on Deep Neural Networks" Aerospace Science and Technology, Paper in preparation.

1.7 Thesis Structure

The structure of this thesis is as follows:

In Chapter 2, the literature regarding direct design methods and inverse design methods is reviewed. Different approaches such as gradient-based optimisation methods, heuristic methods, enhanced optimisation methods via surrogate models and machine learning techniques. The literature with respect to parameterisation techniques is also revised.

Chapter 3 describes the mathematical representation of the aerofoil geometry. The literature for different parameterisation methods is presented and analysed according to their intuitiveness and flexibility. In addition, a comparative analysis among different parameterisation methods regarding the accuracy when reconstructing aerofoil geometries is performed.

Chapter 4 describes the methodology to solve the flow field around the aerofoils in the database for this research, involving the grid generation process and the flow assumptions.

A grid convergence study is implemented and the governing equations of the flow are described.

In Chapter 5, the description of the construction of the database is portrayed. Also, the implementation of the DNN for aerofoil design optimisation is described, including the structure of the dataset used to train the network. An evaluation of the DNN predictions is also presented, including a critical analysis of the different network's hyperparameters configuration.

In Chapter 6, the DNN configuration established in Chapter 5 is implemented for aerofoil multi-objective design optimisation, including different flight conditions. Within the implementation of the DNN, 5 case studies are presented for different number of design objectives, where the degree of the multi-objective design optimisation is based on the number of design requirements. The design optimisations performed is based on a multi-objective aerodynamic optimisations of the NACA 66-206 aerofoil to improve its aerodynamic performance.

Chapter 2

Literature Review

A detailed review of the relevant and current literature related to computational-based aerodynamic design methods is presented in the following sections of this chapter. This review states the background of computational aerodynamic design methods based on numerical methods and design methods based on a heuristic approach. A discussion of the effectiveness and the range of the applicability of each method is previewed along with a critical comparison of their capabilities. The current gap in knowledge is summarised at the end of this chapter.

2.1 Direct aerodynamic design: Numerical Optimisation methods

Numerical optimisation methods involve in general a deterministic repeating process of design, evaluation, and improvement; these methods normally employs a gradient-based optimisation process and are usually linked directly with a CFD code. Numerical methods attempt to find an optimal solution to an objective function or cost function by methodically adjusting the configurations of the design variables. The objective in a gradient-based shape design optimisation is to minimise a cost function, which is associated to the aerodynamic characteristics (lift, drag, lift/drag ratio, target pressure distribution, etc.) of

the shape itself.

Out of the different methods to obtain the gradient of an objective function, the finite differences method is the simplest. By using finite differences, the gradient elements are estimated by separately disturbing the design variables using a finite step. Then, the correlated value of the objective function is iteratively calculated using a CFD code. The numerical optimisation algorithm then uses the derivatives from the gradient elements to compute the optimal direction search. Thus, the computational cost of finite differences methods is directly related with the number of design variables, and hence becomes computationally expensive while increasing the number of design variables. The implementation of finite differences for a gradient-based numerical optimisations for aerodynamic shape design dates back to 1974. Hicks et al. [20] implemented the full potential flow equations along with the finite differences method on two-dimensional transonic aerofoils, thus the usage of this method was restricted to inviscid incompressible flows. This method was also expanded to a three-dimensional wing design [21]. Later Reuther et al. [18] implemented the finite differences method for a supersonic wing/body using three-dimensional flows solved by the Euler equations.

The derivatives of a cost function can be obtained in different ways, however, currently some of the most used methods for aerodynamic design optimisation are the adjoint methods. The implementation of adjoint methods for design optimisation has been introduced by Jameson [35, 36, 37], for design systems governed by the full potential flow equations, Euler equations, and the Navier–Stokes equations. The adjoint method is based on control theory, and its main motivation was to remove the dependency of estimating each design variable separately, such as the case of finite differences. In contrast to finite differences, by implementing the adjoint methodology, the modifications in the aerofoil shape are estimated on gradient information which is obtained by solving the adjoint problem. The adjoint methods can be divided in two categories: continuous adjoint and discrete adjoint [38, 39, 40, 41].

2.1.1 Implementation of the adjoint method in shape optimisation

Kuruvila et al. [42] implemented a numerical method for aerodynamic design optimisation by implementing a cost function and a Lagrange multiplier based on a ”One Shot” approach. The study by Kuruvila was limited to the case of a subsonic potential flow passing over an aerofoil. Kuruvila accomplished to develop an efficient numerical framework for design optimisation, which have the duality of being able to find an aerofoil shape for a prescribed potential or a target pressure coefficient. Three test cases were analysed for a shape recovery of the NACA 0012 and the FX 60126/1. An initial non-smooth geometry is subjected to the design requirements of the aerofoils mentioned before at Mach 0.0 and AoA of 0.0 °. The framework accomplished to recover the aerofoil's shape efficiently in most of the cases for a prescribed pressure distribution and a prescribed potential, however, this approach was limited to a single flight condition (e.g. a single angle of attack and a single flow velocity), and for a subsonic potential flow solution.

Anderson et al. [43] implemented the Navier–Stokes equations with the continuous adjoint method to obtain the solution of the flow on a unstructured domain. Anderson developed a continuous adjoint method to calculate the sensitivity derivatives using a second order discretisation approach for inviscid and viscous flow. A method to modify the grid for both viscous and inviscid cases during the design cycle was also developed. However, Anderson encountered with several drawbacks regarding the demand for exact second order derivatives of the velocities necessary to calculate the shape sensitivity derivatives of the continuous adjoint for the viscous case. To overcome this, Anderson proposed considering a higher order of discretisation of the flow field, which represents a higher computational cost due to a higher order of accuracy required to compute the flow field.

Baeza et al. [44] introduced the Rankine–Hugoniot conditions of shock waves [45] to deal with the discontinuities generated by a shock wave in the flow field, and the continuous adjoint equations for optimal geometry design of aerofoils governed by the Euler equations. Baeza intended to minimise the wave drag using the NACA 0012 aerofoils as the baseline at Mach 0.8 and AoA of 1.2 °, with a imposed constrain of a lift coef-

ficient grater than 0.36. Although Baeza accomplished to minimise the drag coefficient up to 10%, the enforcement of the adjoint/Rankine–Hugoniot condition is complex and requires the numerical location of the shock wave, also it is limited to a single Mach regime.

Jameson et al. [46] implemented the continuous adjoint approach for unstructured grids governed by the Euler equations, where the complications to obtain the gradients on unstructured grids presented in previous studies, was resolved by introducing a reduced gradient approach. This reduced gradient method was applied with an imposed thickness constraint and was implemented for aerofoils and three-dimensional wings. In general the process generates a sequence of grids that defines the aerofoil/wing geometry, then the flow field is computed by the Euler equations, furthermore the solution of the flow is evaluated by the adjoint solver, which performs the shape modification and mesh deformation in an iterative process until the design criterion is satisfied. For the case of aerofoils sections it was proven that for a specific Mach number and AoA at least 50 design cycles were needed.

Hicken et al. [47] developed an adjoint-based gradient algorithm for the Euler equations to perform wing optimisations. Hicken integrated a geometry parameterisation based on B-splines and a mesh movement method. The mesh movement method consisted in the parameterisation using B-splines of both the surface of the wing and the volume mesh. The mesh movement integration method resulted in a reduction in the computation time compared with node-based mesh movements.

Hu et al. [48] stepped forward to the study of aerodynamic optimisations at different flight conditions, implementing the adjoint-based optimisation method to optimise aerofoils in a Busemann configuration [49]. The adjoint optimisation was used to design optimum shapes for a wave drag reduction at Mach numbers ranging from 1.1 to 1.7 for a single angle of attack. Hu accomplished to find optimal aerofoil shapes at different Mach numbers by implementing a multiple design points, where the objective function is a weighted average of the drag coefficients at the different flight regimes. The com-

putation cost of the adjoint-based implementation was similar to the computation cost of solving the flow Euler equations.

In a more recent study Wang et al. [50] coupled the Stackelberg game theory [51] and the adjoint formulation for aerodynamic design optimisation for single-objective and two-objectives. In the framework developed by Wang two players are involved in the optimisation method, each o which is responsible of modifying a subset of the design variables. This framework is implemented to perform single-objective and two-objective optimisations of the ONERA M6 and the RAE 2822 aerofoils. Wang accomplished to minimise the drag coefficient for the single-objective optimisation at a flow regime of Mach 0.729 and AoA of 2.31 $^{\circ}$, and the two-objective optimisation at Mach numbers of 0.8395 and 0.83.

Presently adjoint methods have been used along heuristic methods such as generic algorithms. Zhu et al. [52] developed the Experience-Independent Inverse Design Optimisation (EIIDO) framework. This to overcome the need of experience from the designer, and implemented to the design process of compressor cascade aerofoils. EIIDO framework calculates the target pressure distribution automatically for a specific case by using Genetic Algorithms, as an alternative to the manual target pressure distribution specification. Then the shape of the aerofoil is estimated by the adjoint method. Zhu implemented the EIIDO framework on a NACA 65-(12)10 aerofoil to minimise the total pressure loss coefficient where two numerical cases were examined. In case 1 only the control points of the aerofoil shape are modified, whereas in case 2 the control points of the aerofoil shape and the pressure control points are varied. The overall computational time for case 1 was about 15 hours and 20 hours for case 2. Darwish et al. [53] adopted a similar approach coupling Genetic Algorithms along with adjoint-based optimisation codes for aerodynamic shape design of helicopter rotors. Using a NACA 0012 as the aerofoil baseline, Darwish accomplished to increase the L/D ratio from 0.0238 to 0.026.

In a different approach by Gagliardi et al. [54] the adjoint method is coupled with a CAD-compatible Boundary-Representation (B-rep) for shape parameterisation. Gagliardi

implemented the surrogate model Radial Basis Functions (RBF) [28] to transfer the movements of the control points from a Non-Uniform Rational Basis-Splines (NURBS) [55] to the B-rep. The adjoint-based B-rep model was implemented on a drag coefficient reduction in a two dimensional study optimisation of the RAE 2822 aerofoil, at flow conditions of Mach 0.725 and AoA 2.92°, using a 16 degrees of freedom NURBS-based parameterisation, and then applied to an aircraft model for a three dimensional case. Gagliardi demonstrated that the use of the B-rep model can reduce the degrees of freedom on the optimisation process, without damaging the optimisation accuracy. However, the optimisation method developed by Gagliardi does not include optimisation process for multiple flight conditions.

2.2 Aerodynamic shape design based on heuristic optimisation methods

One of the main challenges while performing computational-based design optimisation is to avoid local optima points within the design space. The gradient-based optimisation methods defined in the previous section struggles to find a global optimal design [56]. This due to the non-continuous search design space topology that can be encountered during aerodynamic shape optimisation, and the influence that the initial guess of the optimisation algorithm has on the likelihood in getting stuck in a local optima. In contrast with gradient-based design methods, non-gradient based methods do not need a continuous and mathematically predictable search space, and regularly are able to find the global optima of an objective function [57].

Non-gradient optimisation methods such as heuristic optimisation methods involves in general a probabilistic procedure, and usually implies a higher complexity in their implementation than gradient-based methods. These methods are metaheuristic, which means that they are approximate and they define some high-level principles to guide the search process in the design space [58]. These methods can offer a robust approach to increase

the probability to find a global optimum solution [24]. Among the gradient-free heuristic algorithms, population based optimisation have been widely implemented for optimal shape design. These methods are established on theory of evolution and the outlast of the fittest and are known as Evolutionary Algorithms (EA) [56]. Genetic Algorithms are one of the most popular EA methods and it have been proved that in most cases, although the computational cost of Genetic Algorithms is higher than gradient-based algorithms, these methods are usually highly efficient for the to solve the problem of finding an optimal aerodynamic geometry for a specific objective [59, 60, 61, 62].

2.2.1 A general description of Genetic Algorithms

Genetic Algorithms (GA) are population-based optimisation methods inspired by natural evolution, which is the process that animals and plants applies through generations to modify their biological characteristics to approximate an optimal form. GA are able to solve complex optimisation requirements due to their capability to search different solutions simultaneously in the search space [56].

In general, the initial population of individuals in a GA is randomly generated, followed by an evaluation of the traits and likelihood of the individuals to reproduce; this is also called the evaluation of the fitness of the individuals. Then based on the fitness evaluation, the individuals to take part in a genetic process of reproduction are selected. The selection process of the fittest individual can be applied by different methods such as tournament selection [56, 63], elitism [56, 64], and roulette wheel [26]. The individuals with the highest fitness have more chances to be selected in this process. The reproduction of the individuals selected to produce the next generation is achieved by using the crossover operator and the mutation operator.

The crossover process involves an exchange of characteristics between two individuals to generate two different child with a combination of both traits from their parents. There are different methods for crossover operations such as single-point crossover [56, 65], two-point crossover [56, 66], uniform crossover [66], and multi-point crossover

[67] among others.

The mutation process [56, 66, 68], in contrast to crossover, involves an alteration of the characteristics of the individuals in a probabilistic process. The advantages of applying a mutation operator is that it enables the GA to investigate locations in the search space that can be potentially favourable. The mutation process can be regulated by the mutation rate, where high mutation rates indulges a more diverse exploration in the search space to the detriment of loosing good solutions. On the other hand, low mutation rates generates a more smooth exploration in the search space at the cost of possibly getting stuck in a local optimal solution.

According to Chiba et al. [69] GA in general have noticeable features which benefits their implementation in aerodynamic optimisation problems. GA have the ability to perform a multi-point search which provides the ability to find global optimal solutions. In addition, CFD codes can be directly coupled with GA which makes it suitable for aerodynamic shape optimisation.

2.2.2 A general description of Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) algorithms are stochastic by nature and are based on population theories. PSO is inspired in the collective animal behaviour such as flocks of birds, emulating the their ability to share information in a socio-cognitive interaction among the swarm to find food sources and avoid predators. From an optimisation perspective, PSO randomly generates an initial swarm which is a set of solutions that propagates in the n-dimensional design space. These initial solutions then navigates through the design space towards a global optima by a defined number of iterations, exchanging critical information regarding the design space topology among the members of the swarm [25]. In general the PSO algorithm comprises three steps: the swarm particle's position and velocities generation, the velocity update, and the position update. Where a particle indicates a coordinate in the design space, and adjust its location iteratively updating the velocity from one coordinate to another [70].

2.2.3 Implementation of heuristic methods in shape design optimisation

Kim et al. [71] conducted a study of optimal shape of axial-flow fans by implementing the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [72], which is one of the GA variations. A multi-disciplinary aerodynamic and aeroacoustic optimisation was performed. The target of the multi-objective optimisation was to improve the total efficiency and reduce the sound pressure level of the axial-flow fan, by using two design variables being the sweep and lean angles of the blade tip. Although Kim's framework for multi-objective optimisation was restricted to only 2 design variables, the framework accomplished to improve by 2.20% the total efficiency and decrease the sound pressure level by 0.62 decibels.

From a different perspective Lyu et al. [73] implemented the NSGA-II for aerodynamic shape design optimisation of a wing shape under an airflow governed by the Reynolds-Averaged Navier–Stokes (RANS) equations. The objective of the study was to perform a drag minimisation with a lift-constrained condition. The initial conditions of the process was a population of 24 individuals for 200 generations. Lyu showed that the NSGA-II underperformed when handling more than eight design variables, also, its implementation resulted in a high computational cost, and failed to meet optimisation constrains.

Skinner et al. [74] developed a structured Genetic Algorithm (sGA) for aerodynamic optimisations. The framework of the sGA was based on the NSGA-II algorithm. However, it was found that the NSGA-II algorithm does not include infeasible solutions which violate constrains in the search for for Pareto-optimality. The NSGA-II considers these solutions as inferior, even those with superior objective function evaluations and small constraint violations. Motivated on the assumption that, Pareto-optimality solutions can exist between a region of feasible and infeasible solutions [75]. Skinner enforced a constraint based reasoning through proportional penalty functions, which enables the presence of feasible and infeasible solutions. The sGA was coupled with potential flow vortex

ring panel method to minimise the drag force of an alternative non-planar wing shape configuration with a lift constraint condition. The implementation of the sGA algorithm allows shape optimisation of a population of 100 individuals for 500 generations with 28 design variables.

In a different approach Zhao et al. [76] implemented a Multi-Island Genetic Algorithm (MIGA) [77] coupled with a CFD code to minimise the E387 aerofoil's drag by locating the optimal suction slot point on the aerofoil's upper surface. The MIGA algorithm implements a migration method between the algorithm's "islands" to control the diversity of the population, and to enhance the ability of multi-point search in the design space. The suction location and the suction requirement design variables in the MIGA algorithm are binary encoded. An initial population of 50 individuals dispersed over 50 islands were analysed over 50 generations, with a migration rate of 4 generations. Zhao demonstrated that using the MIGA algorithm, for a single suction slot, that the solution converged to an optimal after the 25th generation. Furthermore, the aerofoil drag was reduced by 8.3 %.

Sasaki et al. [78] carried out the study of the application of an Adaptive Range MOGA(ARMOGA) for transonic and supersonic aerodynamic wing shape design to minimise the drag forces. The wing's geometrical characteristics such as the wing's thickness distribution and warp shape were described using 72 design variables. Sasaki accomplished to perform four-objectives optimisations with geometric constraints, where the optimisation used an initial population of 64 individuals. This study highlighted the ability of GA to perform multi-objective optimisations. In addition, Sasaki et al. [79] implemented a GA to optimise a supersonic wing using 105 design variables and 15 geometrical constraints. The advantage of the real-coded MOGA over conventional multi-objective GA, such as the NSGA-II, is their ability to adapt their search regions based on the population distribution statistics. Sasaki established geometrical constraints at three locations over the wing's aerofoil chord line (5%, 50% and 80% of the chord line). However, Sasaki concluded that apart from the enormous computational cost of evolutionary algorithms,

GA not only satisfies the constraints but also takes advantage of loose defined constraints. This lead to over optimised wing geometries that were much thinner at points not defined in the constraints, which may cause structural problems when installing a wing box. This highlights the requirement of proper definition of the constraints to reach efficient optimum results.

Recently Swarm-based optimisation techniques have seen substantial implementation on aerodynamic optimisation problems. Jansen et al. [80] implemented a parallel augmented Lagrange multiplier Particle Swarm Optimiser (ALPSO), for induced drag and viscous drag minimisation of nonplanar lifting surfaces. The ALPSO algorithm directly enforces constraints to the optimisation problem. The aerodynamic optimisation was set with six design variables subjected to lift constraints and geometrical constraints. Jansen highlighted that optimal configurations were found for the case when only induced drag was considered. Geometrical constraints where violated when considering induced and viscous drag. For a two dimensional study, Masdari et al. [81] studied the use of a Salp Swarm Algorithms coupled with a Discrete Vortex Method (DVM) for the optimisation of a Savonius turbine aerofoil, aiming to maximise the power coefficient. Masdari's results indicates an increase on the power coefficient up to 27%.

2.3 Surrogate modelling: Neural Networks

Neural Networks are parallel computational models that have the ability to recognize patterns and highly non-linear relationships between variables and in addition, they are able to learn from them. These models consist in non-linear computational elements linked with weighted connections which forms the network nodes or neurons. They are inspired in the structure of the neurological system of animals [82].

Every neuron in the network may have more than one input, with their corresponding associated weight. The structure of the network is divided in different layers, each of which has several number of neurons. The first layer of the network is the input layer

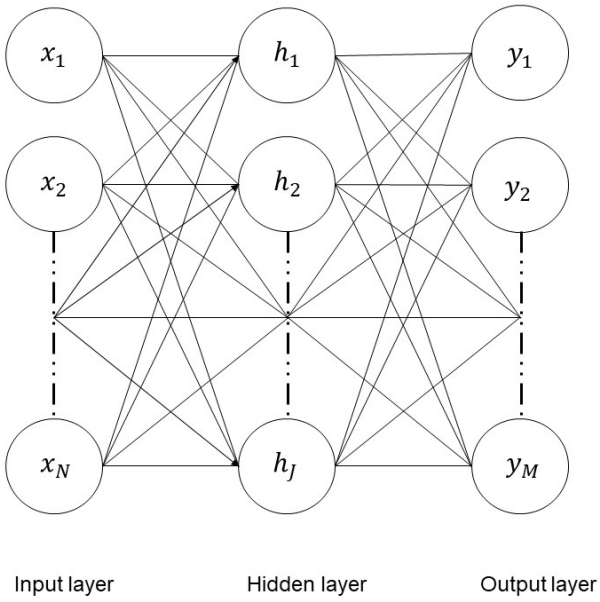

Figure 2.1: Basic topology of an ANN. The first layer of the network is the input layer, the last layer is the output layer, and the layers in-between are the hidden layers.

and this layer addresses the input values, the last layer of the network is the output layer which is composed by the output values. The hidden layers are the layers in-between the input and output layers.

A basic topology of a Neural Network is displayed in Figure 2.1, with specified set of input units $x_1, x_2 ... x_N$, output units $y_1, y_2 ... y_M$ and hidden neurons $h_1, h_2 ... h_J$. Where N is the number of input neurons, M is the number of output neurons and J is the number of the neurons in the hiddem layer. Every neuron h_j in the hidden layer performs a linear combination a_j, where $j = 1, ..., J$, and which involves the values from the input units x_i, where $i = 1, ..., N$ and weight parameters $w_{j,i}$, this linear combination is referred as the activation of the hidden neurons and it is expressed in Equation 2.1 [82]

$$a_j = \left(\sum_{i=1}^{N} w_{j,i} x_i \right) + w_{j,0} \tag{2.1}$$

The parameters $w_{j,0}$ are referred as the biases of the activations, both the weight parameters and the biases are shown in Figure 2.2. Each activation then is transformed using

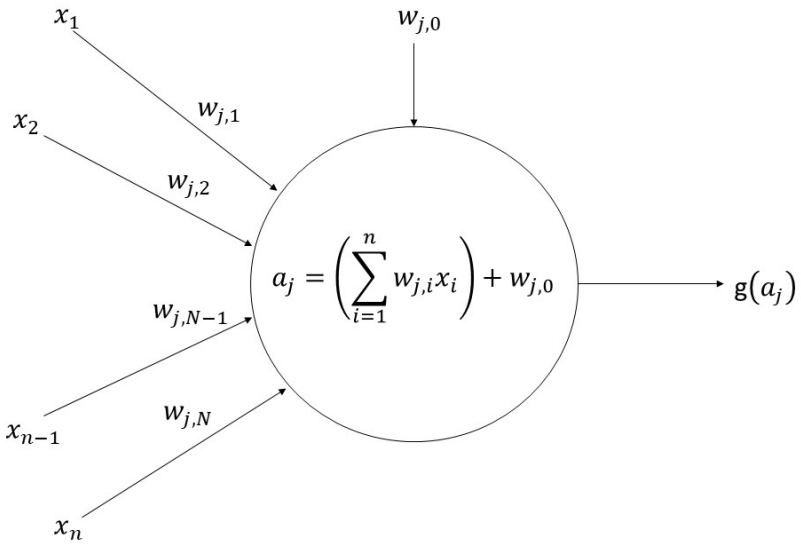

Figure 2.2: A single neuron activation function. The weights from the neurons of the previous layer and the bias factor are the inputs of the neuron.

a non-linear activation function $g()$. which gives:

$$z_j = g(a_j) \tag{2.2}$$

According to recent studies, in modern neural networks, the recommendation is to use the rectified linear unit or ReLU as the activation function of the neurons [83] [84] [85]. The ReLU activation function is defined in Equation 2.3. The function $g(a_j)$ is zero when a_j is less than zero and $g(a_j)$ is equal to a_j when a_j is above or equal to zero. In a feedforward Neural Network all the activation functions are modified by changing the weight factors to obtain an optimum hypothesis which minimises the cost function. This is carried out by an optimisation algorithm which is usually gradient-based.

$$g(a_j) = max\{0, a_j\} \tag{2.3}$$

The quantities z_j are the outputs of the hidden layer units. These values are again linearly combined with a set of weights $w_{k,j}$ to calculate activation units a_k of the network's output units, as shown in Equation 2.4, where $k = 1,...,M$. For a multi layer network the quantities z_j are combined with respective weights to estimate the activations of the neurons in the next hidden layer, and so on until reach the output layer.

$$a_k = \left(\sum_{j=1}^{J} w_{k,j} z_j \right) + w_{k,0} \tag{2.4}$$

The parameter $w_{k,0}$ represents the biases, and the activations are subjected to an activation function to obtain the network outputs values $\hat{y}_k$ which are the predictions of the network for a given set of input features. For the case of a regression problem, the activation function is the identity [82], then:

$$\hat{y}_k = a_k \tag{2.5}$$

Where the values $\hat{y}_k$ are output values of the k^{th} output neuron, which are used to train the network by evaluating them with the actual output values of a given dataset. The values $\hat{y}_k$ are a function of the weights of the network, as it is portrayed in Equations 2.1 and 2.4.

To train a Neural Network for a given set of m samples of inputs X and outputs Y it is necessary to minimise a function which relates the predicted outputs of the network $\hat{y}_k^{(m)}$ and real output values of the dataset $Y^{(m)}$, the superscript m indicates the output values of the m^{th} sample. This function is usually called the cost function and in general it represents the mean squared error between the predicted output values of the network and the true values of the outputs set. To minimise the cost function $C(w)$ the weights w of the network are updated using an optimisation algorithm until a minimum is reached. For a general regression problem the cost function is presented in Equation 2.6 [82]:

$$C(w) = \frac{1}{2m}\left[\sum_{i=1}^{m}\sum_{k=1}^{M}\left(\hat{y}_k^{(i)} - Y_k^{(i)}\right)^2\right] + \frac{\xi}{2}\left[\sum_{l=1}^{L-1}\sum_{w\in W_l} w^2\right] \tag{2.6}$$

The first term of Equation 2.6 represents the mean squared error where M represents the number of output units and m represents the sample size. The second term of Equation 2.6 represents a regularisation therm to avoid over-fitting, where L represents the total number of layers in the network, and W_l represents the set of weights w in layer l. Over-fitting occurs when the network outputs fits the target values of the dataset in the training process with good accuracy, but fails when predicting samples of a dataset not used in the training process. This occurs due the complexity of including multiple hidden layers and multiple neurons on each hidden layer, and can be regularised by adding a regularisation term[82]. The term ξ is the learning rate of the network, this parameter of the network controls how much to update the weights of the model to minimise the cost function.

2.3.1 Implementation of Neural Networks in aerodynamic shape design optimisation

GA and PSO algorithms enhanced by Neural Networks

ANN along with multi-objective Genetic Algorithms are used for aerodynamic optimisation of wind turbines aerofoils in [60]. The aerofoil was parametrised using Bezier curves and the algorithm was trained with their respective control points. The aerodynamic data of the aerofoils were estimated by the implementation of a CFD solver to obtain the solution of the Reynolds Averaged Navier–Stokes Equation (RANS) in steady state. The work demonstrated that the ANN were able to optimise the aerofoils with a reduction in the computational time by almost 50%.

Bellman et al. [86] employed a GA along with ANN for aerofoil shape optimisation in low Reynolds numbers regimes, with the main objective of maximising the lift for Micro-Air-Vehicles applications. The parametrisation scheme used to train the ANN was the

Joukowski transformation. It was shown that the technique of combining GA and ANN can be implemented accurately to obtain an optimal aerofoil design for a target objective parameter.

Kotinis et al. [87] implemented a multi-objective optimiser based on swarm intelligence to obtain solutions that maximises the L/D ratio of a transonic aerofoil in cruse conditions, and minimises the trailing edge noise when approaching to land. ANN where employed to approximate the objective functions via surrogate models. Kotinis demonstrated the effectiveness of the iterative optimisation process enhanced with ANN, by improving the objectives with only five iterations. In a similar approach, Wang et al. [88] constructed a robust optimisation framework based on PSO. The framework is coupled with a RANS solver CFD code and a surrogate model based on a Neural Network. The surrogate model is constructed using the relationship between the design variables, the aerodynamic characteristics of the aerofoil and the Mach number. It was proven that with the combination of the searching abilities of the PSO and the Neural Network, a fast convergence speed is achieved. Khurana et. al. [89] employed a PSO along with a Self Organising Map (SOM) [90] to perform aerodynamic shape design optimisations of aerofoils. Khurana implemented an ANN model with a dataset of 3000 aerofoils for training, to develop the surrogate model that links the geometry variables with the aerodynamic coefficients. The aerodynamic characteristics of the aerofoils in the dataset were computed for a Reynolds number of 3.0 million and a Mach number of 0.35, at a fixed angle of attack of zero degrees.

The population based surrogate assisted methods stills represents a complex process where the initial ample in the design space must be done, followed by the numerical simulations to construct the surrogate model, and then it becomes an iterative process where the new individuals in the population are evaluated and assessed by the surrogate model.

Neural Networks for aerodynamic inverse design

The implementation of inverse design methodologies is increasing in the field of aerodynamic design for many applications. Pierret et al. [91] developed a knowledge-based method based on ANN for an automated way to design turbine blades. The database was constructed by the use of the solution of the Navier-Stokes equations from previous designs and Bezier curves [92] to reconstruct the geometries.

Sun et al. [93] implemented an ANN along with a Self Organising Map (SOM) in order to develop an inverse aerodynamic design framework for aerofoils and wings. The ANN was trained in order to obtain a correlation between aerofoil/wing shape and their corresponding aerodynamic characteristics under certain work conditions. The database contained 208 aerofoils with their respective shape parameters from the PARSEC parametrisation scheme [94] and their aerodynamic coefficients for a fixed flight condition case. It was shown that the implementation of ANN for inverse aerofoil/wing design performs better in accuracy and efficiency than other conventional design methods.

Kharal et al. [95] employed an ANN for aerofoil generation for a given pressure distribution around the aerofoil. The database was comprised by the shape parameters from the Bezier-PARSEC method [96] and the pressure distribution the aerofoils from Panel Methods. Feed-Forward Back Propagation (FFBP), Generalized Regression (GR) and Radial Basis (RB) neural networks were trained and the performance was evaluated. It was proven that for the specific case in the paper mentioned before the FFBP algorithm presented superior performance.

Sekar et al. [97] performed a Convolutional Neural Network (CNN) to obtain the aerofoil shape from their respective pressure distributions. The pressure distribution around the aerofoils were obtained by the use of XFOIL code for a fixed Reynolds number of 100,000 and angle of attack of 3 $^{\circ}$. The dataset containing 1343 aerofoils encloses the aerofoil coordinates and the pressure distribution.

Emre Yilmaz et al. [98] implemented a deep learning approach to acquire the hypothesis that describes the links between the aerofoil geometries and the surface pressure

distribution. The dataset was generated using the aerofoil coordinates from the UIUC Applied Aerodynamic Groups [34] and the the pressure distributions of the aerofoils were for a fixed Reynolds number of 10^5 and zero angle of attack. Achieving an accuracy on their predictions of about 80%.

2.4 Gap in the knowledge

Based on the literature review of the adjoint methods applied to aerodynamic shape design, it can be summarised that very accurate algorithms have been developed over the last decade, capable of minimising the drag coefficient of an aerofoil while maintaining or improving the lift coefficient, or capable of describe an aerofoil geometry for a target pressure distribution. In addition, several research have been done in order to undermine the computational burden of the iterative process of implementing deterministic strategies, through the use of efficient re-meshing techniques and enhanced adjoint methods coupled with population based algorithms, the computational cost of shape design have seen substantial reduction. However, a drawback of the adjoint is their underperform search for an optimal solution in a highly complex design space, as such as presented for a wide range of flight conditions targets. Adjoint solvers may become local-optimally trapped in a non-continuous space, which limits their implementation for aerodynamic design optimisation problems regarding different flight conditions requirements.

In addition, the principal drawbacks related with heuristic algorithms such as GA and PSO are the high computational burden when coupling gradient-free algorithms with CFD codes. This computational expense can be alleviated by implementing surrogate models such as ANN to approximate the objective function. However, most of the literature regarding aerodynamic shape optimisations are oriented for a single flight condition. In addition, the underperformed constrain handling capabilities, and the requirement of specific tuning of the initial conditions such as the chromosomes in GA, the initial population and generations, and the swarm picking of PSO; suggest that the experience from

the designer when implementing a heuristic approach plays an important role.

Furthermore, the literature reviewed it seems that surrogate methods have been implemented mostly to enhance population based optimisation algorithms for aerodynamic design. Heuristic assisted methods with surrogate models such as ANN have become highly popular in aerodynamic design due to their capability to reduce the computational cost of the process, compared with a purely heuristic approach such as GA and PSO. Now, aerodynamic inverse design by surrogate models such as ANN and DNN is fairly recent, and little research have been done regarding the influence of different network architectures and different optimisers within the neurons on the accuracy of the aerodynamic shape outputted by the network. In addition, most of the inverse design cases by ANN are for a single flight conditions such as reviewed in Section 2.3.1. This brings the opportunity to study the capabilities of surrogate models such as ANN and DNN to perform aerodynamic inverse design for multiple flight conditions, e.g. transonic and supersonic flight conditions at different angles of attack.

Chapter 3

Aerofoil shape representation: Parameterisation

Geometric parameterisation plays an important role in inverse aerodynamic design optimisation due to the necessary dimensional reduction of the database. One of its main objectives is to describe a geometry with a minimum number of variables. Thus, the optimisation design output is then a function of the parametric representation of the geometry. The selection of the mathematical representation of the aerofoil during a design optimisation process is of great importance, this due to its influence on computational time, and its effect on the design space. The parameterisation of the aerofoils is in essence the unique numerical identification of every aerofoil which is related with a mathematical scheme that describes its geometry.

During the last decades, important research have been done related to the subject of shape parameterisation and shape optimal design in order to fulfil design difficulties, such as how to describe general shapes which indulges different requirements, and how to meet realistic constrains and realistic geometries, among others. Shaping lift generating devices such as aerofoils or aircraft wings is accomplished by the use of mathematical tools on rapid computers. The main goal of a designer is for example to obtain certain pressure distributions on the aerofoil, or a certain lift coefficient with minimum drag. Therefore

this may be reached by applying optimisation strategies to lead the results towards ideal values.

Back in 1984 Braibant [99], proposed a new choice of design variables instead of using the nodal coordinates of a finite element model as shape variables, which exhibits difficulty of preserving an acceptable finite element during optimisation and the difficulty of dealing with geometrical requirements. Braibant proposed the use of Bezier and B-spline techniques to determine the coordinates of any point inside the design boundaries. Consequently the shape variables were no longer the position of the nodes of a two-dimensional element, but the points which commands the curves. Bezier curves present interesting properties such as they are considered as ”variation diminishing”, which means that each curve is situated within the control points that defines it. Bezier curves offers the capability of the use of multiple variables, this permits the representation of general functions such as spirals and closed curves. The B-spline and the Bezier polynomial methods are extensively applied in aerodynamic design parameteriation and shape optimisation. These methods are capable of construct smooth curves with a small set of variables, which makes them convenient for aerofoil design and optimisation. Research on the subject can provide proof that a polynomial can describe a curve with a small group of design variables. However, there are some characteristics of the Bezier curves that limit its flexibility. First the number of control points is proportional to the degree of the polynomial which defines the curve. For instance a very complex shape would have the need of very high order of the degree of the Bezier polynomials.

Sripawadkul et al. [100], presented a comparison between five aerofoil shape parameterisation methods, to be more specific, Ferguson’s curves, Hicks-Henne bump functions, B-Splines, PARSEC, and Class/Shape function Transformation. In Sripawadkul’s work the comparison is based on 5 desirable characteristics of parametric aerofoils, namely; parsimony, completeness, orthogonality, flawlessness and intuitiveness. Parsimony refers to the ability of a geometric parameterisation to successfully modify the main aerofoil shape by using the smallest number of design variables. Completeness denotes whether

a parameterisation method is capable to describe any aerofoil with an specific accuracy, in essence a paramaterisation scheme should be able to reconstruct a large number of aerofoil geometries for its optimal use. Orthogonality makes reference to assurance that each parameterised aerofoil shape correlate with a unique set of parameters. Flawlessness measures the degree of guarantee that the parameterisation method will generate a well behaved, smooth, and realistic aerofoil shapes. The intuitiveness of the method refers whether the method's parameters are related to the physical characteristics of the aerofoil. A highly intuitive parameterisation method gives insight to the designer about the parameter to be modified.

The PARSEC method proposed by Sobieczky [94], is an intuitive method for 2D airfoil parameterization which is known as the PARSEC method [94]. Sobieczky developed this method in order to decouple a wing section using published databases to improve the aerodynamic performance of an aerofoil. This method implements 11 basic parameters to gain control over the aerofoil curvature. The PARSEC method is one of the most used in terms of aerodynamic parameterisation due to its intuitive characteristics for aerofoil representation, the small number of design variables employed and the smoothness of the shape curves produced by this method. However, research about the comparison between different parameterisation methods have demonstrated that regardless of the intuitiveness of this method, it develops poorly on other desirable parametric characteristics.

A paper by Kulfan [33] presented a universal parameterisation for describing aerofoils, the CST method. The CST method is well know and simple to use, many research on aerodynamic optimisation have been conducted using CST methodology. In Kulfan's work the term "shape function" is introduced. The shape function allows the control of the geometry parameters such as the leading edge radius, trailing edge boattail angle, a specific thickness among others. A "class-function" is introduced to generalise the method for different aerofoil geometry configurations, such as round nose, elliptic, and biconvex aerofoils, among others.

The work developed by Sripawadkul in [100] presents a useful insight of the different

parameterisation methods to be selected in an optimisation process, being the PARSEC and the CST methods the better ranked among the schemes studied. The process to determine the performance of the methods was a ranging score method from 1 being the worst fit of a desired characteristic (parsimony, completeness, orthogonality and flawlessness) to 4 being the full score. However, although the results presented in Sripawadkul's work are a good aid to have a basic understanding of the limitations of each scheme, it does not display a detailed comparison in terms of the accuracy regarding the difference between the reconstructed shape with the original aerofoil shape along the chord line.

The decision making process for selecting a suitable parameterisation scheme for this research was based upon an evaluation of two of the most used schemes, the PARSEC [94] and the Class/Transformation Function (CST) [33] parameterisation methods. The PARSEC method proposed by Sobieczky is well-known and it is often used for aerodynamic optimisations due to its intuitiveness and simplicity. This scheme uses 11 basic parameters which are directly related with the physical characteristics of the aerofoil. On the other hand, Kulfan [33] presented the CST, a universal parameterisation for describing aerofoils. This technique represents a big step towards increasing the flexibility when describing aerofoil shapes. The CST method introduces the term "shape function" which allows the control of the shape parameters which are the physical specifications of the aerofoil. A "class-function" is introduced to generalise the method for different aerofoil geometry configurations.

3.1 PARSEC Method

The PARSEC method is composed by 11 basic parameters to gain control over the curvature as shown in Figure 3.1. These parameters are the leading edge radius (R_{le}), the trailing edge location (Z_{te} at $X = 1$) and thickness (dZ_{te}), the upper and lower crest location coordinates ((X_{up}, Z_{up}), (X_{lo}, Z_{lo}) and their respective curvatures (Z_{xxup}, Z_{xxlo}), the trailing edge angle (α_{te}) and boat-tail angle (β_{te}). Two polynomials are used to describe

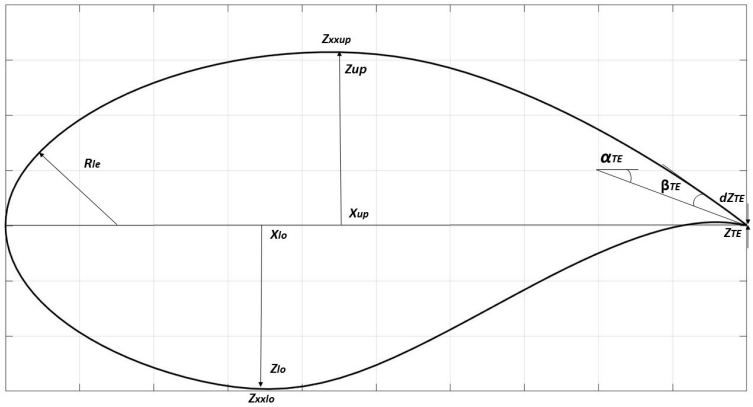

Figure 3.1: PARSEC method parameters [94].

the upper and the lower airfoil surface shape [94]:

$$Z_{up} = \sum_{n=1}^{6} A_{up} X^{n-1/2} \tag{3.1}$$

$$Z_{lo} = \sum_{n=1}^{6} A_{lo} X^{n-1/2} \tag{3.2}$$

where Z_{up} and Z_{lo} are the coordinates of the upper and the lower airfoil surface which can be derived by the linear systems described in Equations 3.3 and 3.4:

$$A_{up}.V_{up} = B_{up} \tag{3.3}$$

$$A_{lo}.V_{lo} = B_{lo} \tag{3.4}$$

where the matrices A_{up} and A_{lo} are defined in Equations 3.5 and 3.6, the term X_{te} refers to the trailing edge location. The vectors V_{up}, V_{lo}, B_{up}, B_{lo} are defined in Equations

3.7 and 3.8:

$$A_{up} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ X_{te}^{1/2} & X_{te}^{3/2} & X_{te}^{5/2} & X_{te}^{7/2} & X_{te}^{9/2} & X_{te}^{11/2} \\ X_{up}^{1/2} & X_{up}^{3/2} & X_{up}^{5/2} & X_{up}^{7/2} & X_{up}^{9/2} & X_{up}^{11/2} \\ \frac{1}{2}X_{te}^{-1/2} & \frac{3}{2}X_{te}^{1/2} & \frac{5}{2}X_{te}^{3/2} & \frac{7}{2}X_{te}^{5/2} & \frac{9}{2}X_{te}^{7/2} & \frac{11}{2}X_{te}^{9/2} \\ \frac{1}{2}X_{up}^{-1/2} & \frac{3}{2}X_{up}^{1/2} & \frac{5}{2}X_{up}^{3/2} & \frac{7}{2}X_{up}^{5/2} & \frac{9}{2}X_{up}^{7/2} & \frac{11}{2}X_{up}^{9/2} \\ -\frac{1}{4}X_{up}^{-3/2} & \frac{3}{4}X_{up}^{-1/2} & \frac{15}{4}X_{up}^{1/2} & \frac{35}{4}X_{up}^{3/2} & \frac{53}{4}X_{up}^{5/2} & \frac{99}{4}X_{up}^{7/2} \end{bmatrix} \tag{3.5}$$

$$A_{lo} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ X_{te}^{1/2} & X_{te}^{3/2} & X_{te}^{5/2} & X_{te}^{7/2} & X_{te}^{9/2} & X_{te}^{11/2} \\ X_{lo}^{1/2} & X_{lo}^{3/2} & X_{lo}^{5/2} & X_{lo}^{7/2} & X_{lo}^{9/2} & X_{lo}^{11/2} \\ \frac{1}{2}X_{te}^{-1/2} & \frac{3}{2}X_{te}^{1/2} & \frac{5}{2}X_{te}^{3/2} & \frac{7}{2}X_{te}^{5/2} & \frac{9}{2}X_{te}^{7/2} & \frac{11}{2}X_{te}^{9/2} \\ \frac{1}{2}X_{lo}^{-1/2} & \frac{3}{2}X_{lo}^{1/2} & \frac{5}{2}X_{lo}^{3/2} & \frac{7}{2}X_{lo}^{5/2} & \frac{9}{2}X_{lo}^{7/2} & \frac{11}{2}X_{lo}^{9/2} \\ -\frac{1}{4}X_{lo}^{-3/2} & \frac{3}{4}X_{lo}^{-1/2} & \frac{15}{4}X_{lo}^{1/2} & \frac{35}{4}X_{lo}^{3/2} & \frac{53}{4}X_{lo}^{5/2} & \frac{99}{4}X_{lo}^{7/2} \end{bmatrix} \tag{3.6}$$

$$V_{up} = \begin{bmatrix} a_{1_{up}} \\ a_{2_{up}} \\ a_{3_{up}} \\ a_{4_{up}} \\ a_{5_{up}} \\ a_{6_{up}} \end{bmatrix} \qquad B_{up} = \begin{bmatrix} \sqrt{2R_{le}} \\ Z_{te} \pm dZ_{te}/2 \\ Z_{up} \\ tan(\alpha_{te} \pm \frac{1}{2}\beta_{te} \\ 0 \\ Z_{xxup} \end{bmatrix} \tag{3.7}$$

$$V_{lo} = \begin{bmatrix} a_{1_{lo}} \\ a_{2_{lo}} \\ a_{3_{lo}} \\ a_{4_{lo}} \\ a_{5_{lo}} \\ a_{6_{lo}} \end{bmatrix} \qquad B_{lo} = \begin{bmatrix} \sqrt{2R_{le}} \\ Z_{te} \pm dZ_{te}/2 \\ Z_{lo} \\ tan(\alpha_{te} \pm \frac{1}{2}\beta_{te} \\ 0 \\ Z_{xxlo} \end{bmatrix} \tag{3.8}$$

3.2 CST Method

The expressions that describes an aerofoil geometry by the use of this approach are Equation 3.9 for the upper surface, and Equation 3.10 for the lower surface of the aerofoil:

$$\zeta(\Psi)_{up} = C_{N2}^{N1}(\Psi)S_{up}(\Psi) + \Psi\Delta\xi_{up} \tag{3.9}$$

$$\zeta(\Psi)_{lo} = C_{N2}^{N1}(\Psi)S_{lo}(\Psi) + \Psi\Delta\xi_{lo} \tag{3.10}$$

The variables Ψ and ζ are functions of the chord line with the following relations

$\Psi = x/c$ and $\zeta = z/c$, where x is the component of the aerofoil coordinates in the x direction (along the chord line c) and z is the component of the aerofoil coordinates in the z direction (normal to the chord line c). The parameter $\Delta\xi$ is the trailing edge thickness and it can be decomposed into the upper and lower trailing edge thickness by the relations in Equation 3.11.

$$\Delta\xi_{up} = \frac{zu_{TE}}{c} \qquad \Delta\xi_{lo} = \frac{zl_{TE}}{c} \tag{3.11}$$

Where zu_{TE} and zl_{TE} are the z coordinates of the trailing edge for the upper and lower surfaces respectively. From Equations 3.9 and 3.10 the parameter $C_{N2}^{N1}(\Psi)$ is the class function term where the coefficients $N1$ and $N2$ provides control of the shape function curve to represent a large number of geometries. To define a biconvex airfoil for supersonic flight the parameters are both usually set to 0.75 [33]. The class function is represented by equation 3.12:

$$C_{N2}^{N1}(\Psi) \quad = \quad (\Psi)^{N1}[1-\Psi]^{N2} \tag{3.12}$$

From Equations 3.9 and 3.10 the parameters S_{up} and S_{lo} are the general set of functions known as "shape functions" that together represents the unique shape of the aerofoil. The uninvolved method to represent the shape function is the unit shape function ($S(\Psi) = 1$). The unit shape function can be break down into aerofoil components of the order $n+1$ where n is the order of the Bernstein polynomials 3.13.

$$S_{r,n}(x) \quad = \quad K_{r,n}x^r(1-x)^{n-r} \tag{3.13}$$

Where $r = 0-n$ and the variables $K_{r,n}$ are defined in Equation 3.14. Then the shape

functions can be expressed as shown in Equation 3.15.

$$K_{r,n} = \frac{n!}{r!(n-r)!} \tag{3.14}$$

$$S_{up}(\Psi) = \sum_{i=0}^{n} Au_i S_i(\Psi) \qquad S_{lo}(\Psi) = \sum_{i=0}^{n} Al_i S_i(\Psi) \tag{3.15}$$

Where n is the number of design parameters. Au_i and Al_i are the shape coefficients and $S_i(\Psi)$ are Bernstein polynomials of order n. The physical parameters of the aerofoil such as the leading edge radius, the width of the trailing edge and the boat-tail angle (β) are closely related to the first and last shape functions as presented in Equation 3.16.

$$A_0 = \sqrt{\frac{2R_{le}}{c}} \qquad A_n = \tan\beta + \frac{\Delta\xi}{c} \tag{3.16}$$

As stated before for any order of the Bernstein polynomials to modify the unit shape function, the shape physical characteristics are defined only by the first and the last terms of the shape function. The other parameters in between have no effect in the leading edge radius nor trailing edge thickness or the trailing edge boat-tail angle. The position of the middle terms of the shape function are evenly spaced along the chord line at points which are established as:

$$(\Psi)_{S,max,i} = \frac{i}{n} \tag{3.17}$$

For $i = 0 - n$. The position of the peaks of the aerofoil are evenly separated along the chord line and defined by the same terms that defines the class function:

$$(\Psi)_{Zmax} = \frac{N1+1}{N1+N2+n} \tag{3.18}$$

The evenly spaced shape functions over the chord line in the CST parameterisation may lead to resolution limitations when reconstructing aerofoils. As for example, when reconstructing complex aerofoils which needs higher detail over the leading edger or the trailing edge surfaces, may require more control points over these areas, as it is found later in some case studies within this thesis.

3.3 Aerofoil inverse shape fitting

In the interest of measuring the effectiveness and accuracy of a parametrisation scheme when describing a geometry, it is necessary to analyse the ability to reconstruct different aerofoils with minimum error. The process of inversely reconstruct an aerofoil based on the coefficients from the parameterisation scheme is portrayed in Figure 3.2. Aerofoil inverse shape fitting represents an optimisation problem by definition. Given the aerofoil coordinates, the difference (error) between the parametrised and the original curves have to be minimised, this difference is described by Equation 3.19, where $Z_i original^{(X)}$ is the Z aerofoil coordinate at point X_i on the original aerofoil chord line, and $Z_i parameterised^{(X)}$ is the Z aerofoil coordinate at point X_i on the parameterised aerofoil chord line. From Appendix B the MATLAB script B.1 uses the optimisation algorithm fmincon from the MATLAB library, to evaluate the difference between the aerofoil geometry generated by the parameterisation scheme (PARSEC or CST) and the aerofoil shape described by the original coordinates of the aerofoil. The fmincon function iteratively modifies the parameterisation coefficients until a minimum of the function $f(y)$ from Equation 3.19 is achieved. The fmincon calls the functions CST_fitting and PARSECfitting described in the MATLAB scripts B.2 and B.3, where the CST and the PARSEC functions are called to generate the parameterised geometry (see MATLAB scripts

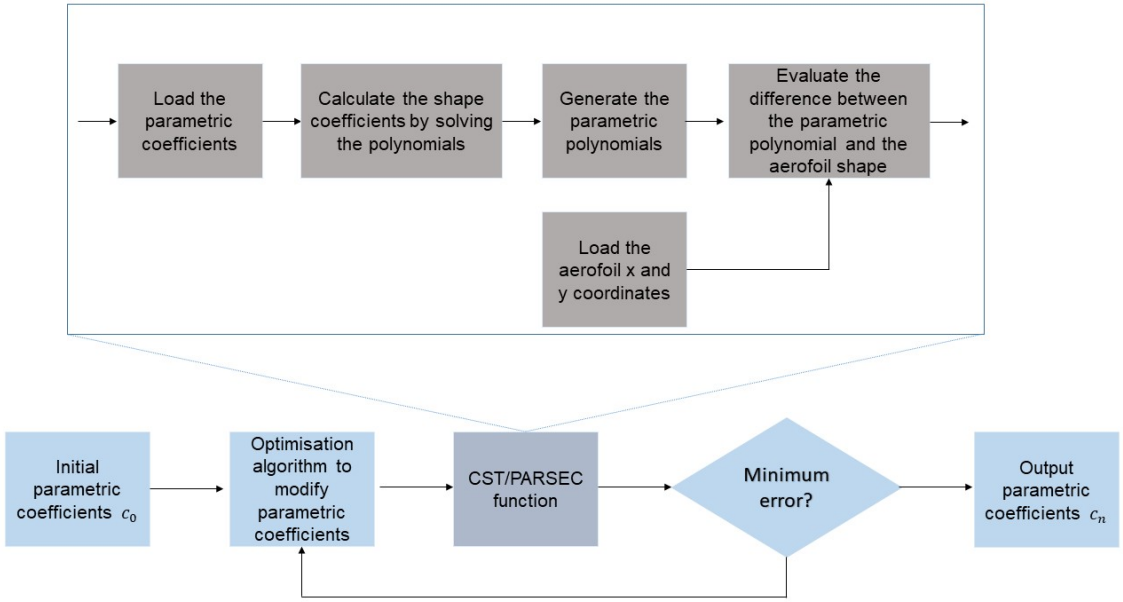

Figure 3.2: Aerofoil inverse shape recovery process: The optimisation algorithm begins with an initial guess of the parameterisation coefficients which then modifies to minimise $f(y)$ from Equation 3.19.

B.4 and B.5).

$$f(y) = \sqrt{\sum_{i}^{n} \left(Z_i original^{(X)} - Z_i parameterised^{(X)}\right)^2} \tag{3.19}$$

3.4 Parameterisation evaluation

According to the comparison study presented by Sripawadkul et al. [100], the completeness refers to whether a parameterisation scheme can reconstruct any aerofoil up tu a specific degree of accuracy. To evaluate the completeness they took some considerations, as for example only evaluating conventional aerofoil types and only aerofoils which are described with more then 60 coordinates. In addition, they used the standard wind tunnel tolerances as constraints to evaluate the accuracy of the parameterisation. The standard wind tunnel tolerances are the constraints for the residual differences between the parameterised geometry and the official aerofoil geometry. The standar wind tunnel tolerances

for the residuals are $\pm 3.5 \times 10^{-4}$ from the leading edge to 20% of the chord length, and $\pm 7 \times 10^{-4}$ elsewhere [100, 101, 102]. The considerations of including only aerofoils which are represented with more than 60 coordinates may lead to some limitations on the comparison of parameterisation schemes. Most of the aerofoils selected for the investigation within this thesis are described with less than 60 coordinate points among the chord line, as for example the family of transonic aerofoils Grumman Gulfstream GIII which are described with 26 coordinates along the chord line. However, the study by by Sripawadkul et al. gives important insight regarding the different methods for parameterisation. The CST and PARSEC methods were selected for further investigation based upon the study presented by Sripawadkul et al [100].

To further evaluate the CST and the PARSEC methods, 50 aerofoils for high speed with complex shapes, and which are described with less than 60 coordinates are reconstructed. The standard wind tunnel tolerances are used as a constraint for the accuracy of the parameterisation schemes. To have a fair comparison, taking into consideration that the PARSEC method uses 12 design variables, the CST method was set to have 12 design variables which correspond to a Bernstein polynomial "n" of order of 3 (n = 3). It was found that for complex aerofoil shapes with a small number of coordinates, the PARSEC and the CST with 12 design variables did not always fitted the standard wind tunnel tolerances. Figure 3.3 shows the error distribution along the chord line of the reconstructed transonic aerofoil NASA/AMES/Hicks A-01, which is represented with 41 coordinates. As it is shown in Figure 3.3 both the CST and PARSEC methods fails to fit the error within both of the standard wind tunnel tolerances.

However, for some aerofoils the CST managed to reconstruct the shapes with the error within the wind tunnel tolerances, as it is shown in Figure 3.4. A NACA 0010-35 which is defined by only 17 coordinates is reconstructed, and as it is portrayed in Figure 3.4 only the CST method manages to accurately reconstruct the aerofoil shape. The peaks of the errors outside the boundary of the tolerances for the trailing edge in the CST scheme in Figure 3.4, are due to the fact that the original aerofoil shape is represented by a set

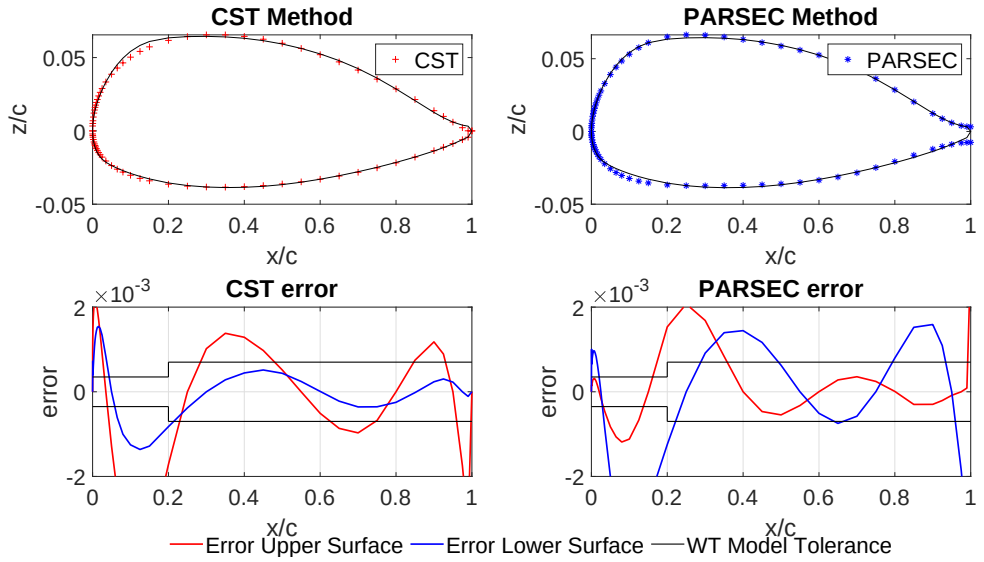

Figure 3.3: NASA/AMES/Hicks A-01 error distribution comparison between CST method with 12 design variables (n = 3) and PARSEC method. The red line represents the error of the upper surface, the blue line represents the error of the lower surface and the black lines comprises the standard wind tunnel (WT) tolerances, for the leading edge up to 20% of the chord and elsewhere.

of coordinates for the upper and lower surfaces which does not finish in a single point. Whereas the CST scheme always finish in a single point at the trailing edge, for both upper and lower surfaces. On the other hand the error of the PARSEC does not accomplish to remain inside the wind tunnel tolerances in some parts of the geometry along the chord line.

There were some cases on which the PARSEC method performed better than the CST method, as for example in the case of the Grumman/Gulfstream GIIIe transonic airfoil which is described with 26 coordinates, and it is shown in Figure 3.5. It can be seen from Figure 3.5 that the PARSEC method accomplished to reconstruct the aerofoil within the standard wind tunnel tolerances for almost all of the geometry along the chord line, except for the segments near the trailing edge. On the other hand, the CST method struggles to stay within the tolerances in several points of the geometry. However, this was not the case for most of the aerofoil geometries as it can be seen in Figures 3.6 and 3.7.

In Figure 3.6 the pie chart of the statistical information of the aerofoils reconstructed with the CST scheme, and whether or not accomplished to remain within the standard

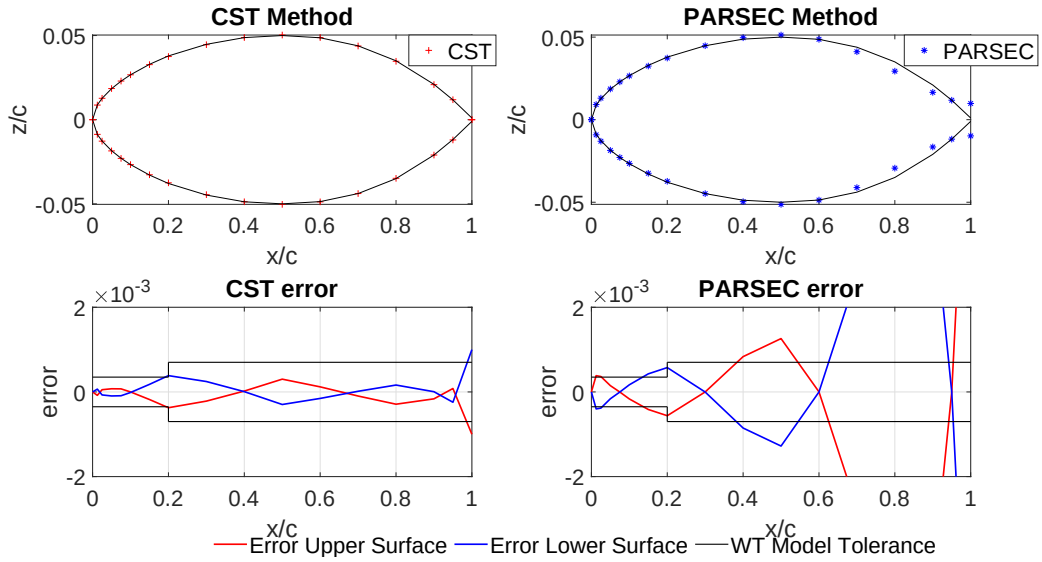

Figure 3.4: NACA 0010-35 error distribution comparison between CST method with 12 design variables (n = 3) and PARSEC method. The red line represents the error of the upper surface, the blue line represents the error of the lower surface and the black lines comprises the standard wind tunnel (WT) tolerances, for the leading edge up to 20% of the chord and elsewhere.

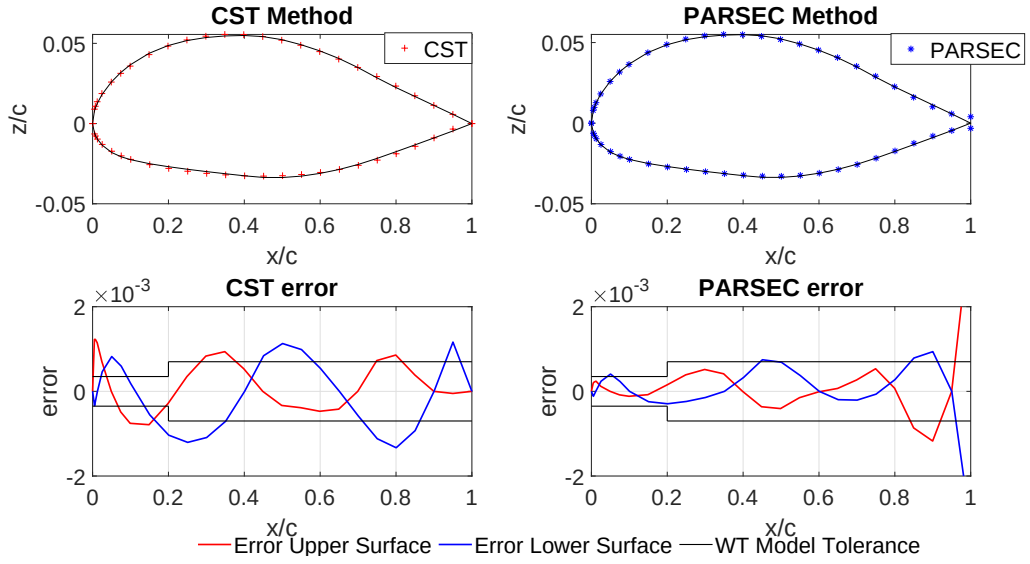

Figure 3.5: Grumman/Gulfstream GIIIe error distribution comparison between CST method with 12 design variables (n = 3) and PARSEC method. The red line represents the error of the upper surface, the blue line represents the error of the lower surface and the black lines comprises the standard wind tunnel (WT) tolerances, for the leading edge up to 20% of the chord and elsewhere.

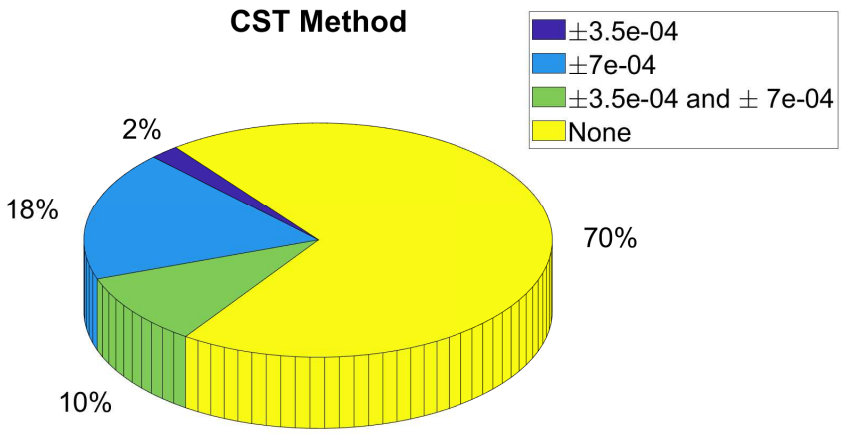

Figure 3.6: Pie chart of the statistical information of aerofoils reconstructed with the CST scheme using 12 design variables. The purple color represents the percentage of aerofoils which only accomplished the $\pm 3.5 \times 10^{-4}$ tolerance. The blue color represents the percentage of aerofoils which only accomplished the $\pm 7 \times 10^{-4}$ tolerance. The green color represents the percentage of aerofoils which accomplished both toleranes. The yellow color represents the percentage of aerofoils which did not accomplished any of the tolerances.

wind tunnel tolerances is portrayed. As it is shown the CST method accomplished to fully reconstruct 10% of the aerofoils within the tolerances. In addition, only 2% of the aerofoils were reconstructed meeting the $\pm 3.5 \times 10^{-4}$ tolerance, but not the $\pm 7 \times 10^{-4}$ tolerance. On the other hand, 18% of the aerofoils were reconstructed meeting the $\pm 7 \times 10^{-4}$ tolerance, but not the $\pm 3.5 \times 10^{-4}$ tolerance. However, most of the aerofoils (70%), were reconstructed by the CST scheme with 12 design variables did not accomplished to meet the wind tunnel tolerances.

In Figure 3.7 the pie chart of the statistical information of the aerofoils reconstructed with the PARSEC method, and whether or not accomplished to remain within the standard wind tunnel tolerances is portrayed. A different picture from the CST scheme is shown in Figure 3.7, out of the 50 aerofoils used to evaluate the parameteriation methods only 6% managed to meet the $\pm 3.5 \times 10^{-4}$ tolerance. The remaining 94% did not accomplished to meet any of the tolerances.

Following the information portrayed in Figures 3.6 and 3.7, the CST presents better

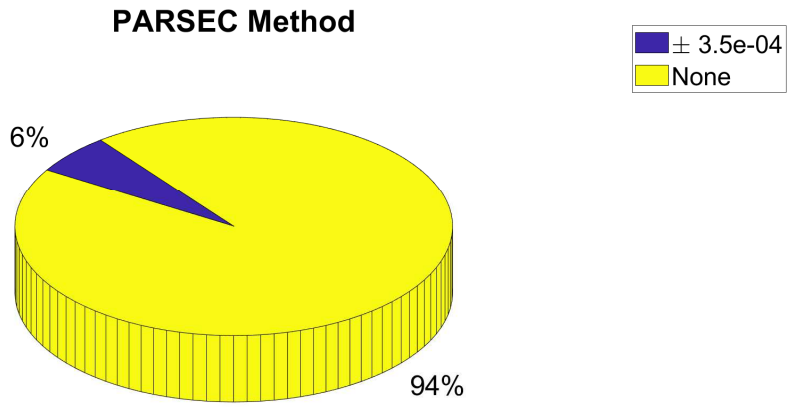

Figure 3.7: Pie chart of the statistical information of aerofoils reconstructed with the PARSEC method. The purple color represents the percentage of aerofoils which only accomplished the $\pm 3.5 \times 10^{-4}$ tolerance. The yellow color represents the percentage of aerofoils which did not accomplished any of the tolerances.

performance when reconstructing aerofoil shapes for high speed and complex shapes, and which are also described with less than 60 coordinates. However, even when the CST performed better than the PARSEC method there is still room for improvement, taking into consideration than 70% of the aerofoils reconstructed by the CST method did not managed to restrain the error within the wind tunnel tolerances. To further improve the CST method performance on reconstructing complex aerofoil geometries, the order of the Bernstein polynomials were increased from n = 3 to n = 5, which correspond to 16 design variables.

Figure 3.8 again, shows the error distribution along the chord line of the reconstructed transonic aerofoil NASA/AMES/Hicks A-01. For this case the CST with 12 and 16 design variables is analysed. As it is shown in Figure 3.8 both the CST with 12 and 16 design variables fails to fit the error within both of the standard wind tunnel tolerances. However, significant improvement for the lower surface error is presented when increasing the design variables and order ”n” of the Bernstein polynomials. In the case of the upper surface there are some improvements relative to the CST with 12 design variables. Still, the CST

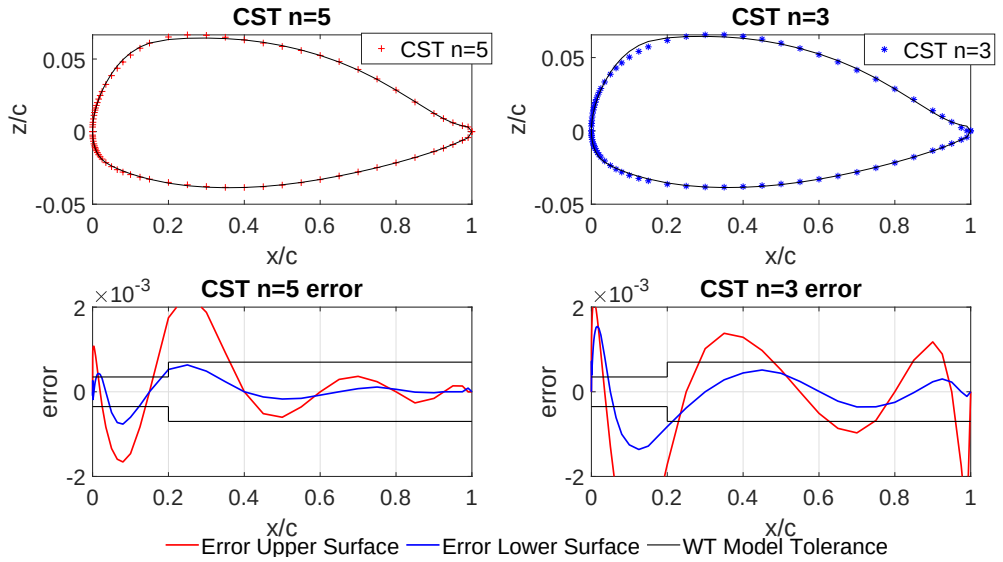

Figure 3.8: NASA/AMES/Hicks A-01 error distribution comparison between CST method with 12 design variables (n = 3) and CST method with 16 design variables (n = 5). The red line represents the error of the upper surface, the blue line represents the error of the lower surface and the black lines comprises the standard wind tunnel (WT) tolerances, for the leading edge up to 20% of the chord and elsewhere.

method with 16 design variables, for the NASA/AMES/Hicks A-01 aerofoil, still fails to remain the residuals within the standard wind tunnel tolerances.

In contrast of the comparison showed in Figure 3.8, there were cases where the CST scheme with 16 design variables managed to reconstruct the aerofoil within the standard wind tunnel tolerances. Figure 3.9 shows the error distribution along the chord line of the reconstructed transonic aerofoil Grumman/Gulfstream GIIIf, for the CST with 12 and 16 design variables. It can be seen from Figure 3.9 a significant improvement on the accuracy when using 16 design variables, all the residuals remains within the wind tunnel tolerances. However, this is not the case for most of the aerofoils reconstructed for the comparison analysis. In Figure 3.10 the pie chart of the statistical information of the aerofoils reconstructed with the CST scheme with 16 design variables and Bernstein polynomials of order n = 5, and whether or not accomplished to remain within the standard wind tunnel tolerances is portrayed. As it is shown the CST method with n = 5 accomplished to fully reconstruct 16% of the aerofoils within the tolerances, which is an improvement over the 10% of the CST with n = 3. In addition, 68% of the aerofoils were

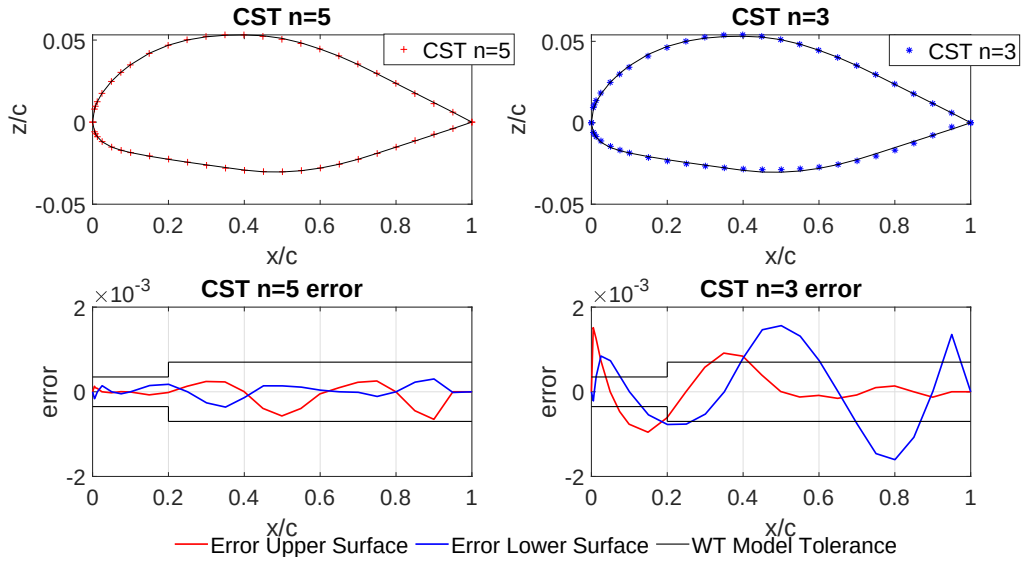

Figure 3.9: Grumman/Gulfstream GIIIf error distribution comparison between CST method with 12 design variables (n = 3) and CST method with 16 design variables (n = 5). The red line represents the error of the upper surface, the blue line represents the error of the lower surface and the black lines comprises the standard wind tunnel (WT) tolerances, for the leading edge up to 20% of the chord and elsewhere.

reconstructed meeting the $\pm 7 \times 10^{-4}$ tolerance, but not the $\pm 3.5 \times 10^{-4}$ tolerance, which brings room for improvement in the accuracy regarding the reconstruction of the leading edge. However, there were some cases, 16%, were reconstructed by the CST scheme with 16 design variables did not accomplished to meet the wind tunnel tolerances.

In Figure 3.11 the pie chart of the statistical information of the aerofoils reconstructed with the CST scheme with 20 design variables and Bernstein polynomials of order n = 7, and whether or not accomplished to remain within the standard wind tunnel tolerances is portrayed. As it is shown the CST method with n = 7 accomplished to fully reconstruct 38% of the aerofoils within the tolerances, which is an improvement over the 16% of the CST with n = 5. In addition, all of the aerofoils met at least one of the wind tunnel tolerances. Thus, for this reason, and to avoid increasing further the number of the design variables that comes with increasing the order of the Bernstein polynomials. The CST parameterisation scheme with Bernstein polynomials of order n=7 is selected to parameterise the aerofoils in the dataset used for the investigation in this Thesis.

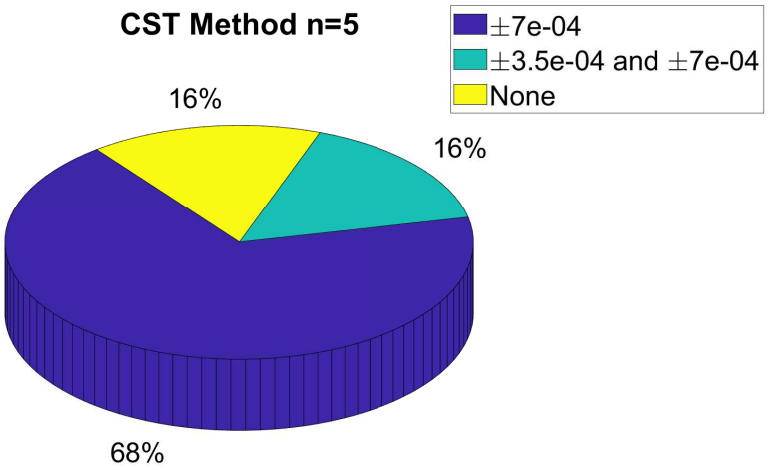

Figure 3.10: Pie chart of the statistical information of aerofoils reconstructed with the CST scheme using 16 design variables, and Bernstein polynomials of order n = 5. The purple color represents the percentage of aerofoils which only accomplished the $\pm 7 \times 10^{-4}$ tolerance. The aqua color represents the percentage of aerofoils which accomplished both toleranes. The yellow color represents the percentage of aerofoils which did not accomplished any of the tolerances.

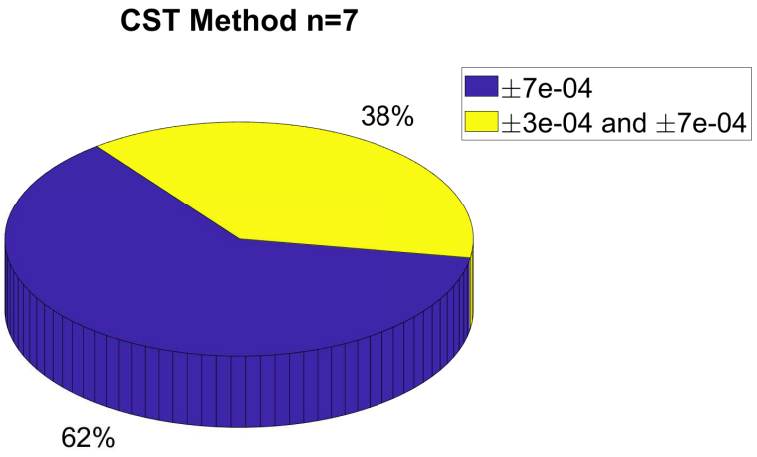

Figure 3.11: Pie chart of the statistical information of aerofoils reconstructed with the CST scheme using 16 design variables, and Bernstein polynomials of order n = 7. The purple color represents the percentage of aerofoils which only accomplished the $\pm 7 \times 10^{-4}$ tolerance. The yellow color represents the percentage of aerofoils which were reconstructed within both tolerances.

Chapter 4

Flow field calculation

The aerodynamic features of the database in this research are obtained by solving the Euler equations for compressible flow. As this work is intended to predict aerodynamic characteristics of aerofoils at high velocities as well as the effect of shock waves forces on the pressure distribution around the aerofoil, and taking into account that the Reynolds number is sufficiently large to minimise viscous effect [103]. The flow is considered to be inviscid in a steady state condition and adiabatic without body forces. The calculation of the flow field is performed using the Stanford University SU2 code to solve the Euler equations for compressible flow with an implicit density-based formulation [104]. The solution convergence criteria for the CFD simulations in this research is one of the following: the change in the drag coefficient is in the order of 10^{-6} over the last 100 iterations, or a number of 1000 iterations is met. A general SU2 configuration file for the flow simulations on this research is displayed in Appendix D.

The SU2 is an open source code that solves the partial differential equations (PDES) on an unstructured grid domain using different numerical methods and based on C^{++} language. The SU2 software was developed in Stanford University and it is primarily applied for CFD analysis and aerodynamic shape optimisations. However, it have been extended to treat potential flow, and electrodynamics among others. Among the capabilities of the SU2 software the most significant are [105]:

- Performing compressible and incompressible analysis by solving the Euler, Navier–Stokes and RANS equations.
- Adjoint-based optimisation design on unstructured meshes.
- Convergence acceleration techniques such as: multi-grid method, linelet preconditioning, Roe-Turkel low Mach number preconditioning
- Mesh refinement and deformation techniques for goal-oriented optimisations
- Solver automation via Python language scripts
- Parallelisation using Message Pass Interface

4.1 Computational Domain

An essential part of this research is the flow field solution via CFD simulations to obtain the aerodynamic data of the database. Thus, it is necessary to create a mesh that adapts to every aerofoil in the database. The grid generation is important to capture essential fluid dynamics phenomena of the flow (e.g. shock waves) around the aerofoil. As this research is intended to develop a multi-objective optimisation framework for transonic and supersonic aerofoils.

A paper by Palacios et al. [105], presents different test cases to validate the Reynolds Average Navier-Stokes (RANS) equations models within the SU2 code. These tests included a transonic case of the flow solution for a RAE 2822 aerofoil at a Mach number of 0.734 and an angle of attack of 2.79°. The domain was an unstructured O-grid (192 x 40) with the outed boundary (far-field) located 100 chord lengths away from the aerofoil. This to reduce the influence of the far-field on the solution of the aerofoil surface.

In a study by Anderson et al. [106] it is reported that based on a sensitivity analysis regarding the farfield, inadequate farfield distance from the aerofoil can lead to solutions of the flow field with stronger shock waves and roughly doubling the drag coefficient.

Then, according to the data published in the mentioned paper, it was stated that the drag could not be accurately resolved with far-fields with a diameter smaller than 96 chord lengths. Therefore, it is important to perform the CFD study with a far-field at least 100 chords lengths away from the aerofoil.

Within this thesis, an O-grid computational domain composed by a far-field and the aerofoil geometry with a wall function for no-penetration boundary condition is generated using the Pointwise software. The radius of the farfield boundary is set to 100 chords lengths away from the aerofoil, to capture essential information of the shock waves at high velocities as portrayed in Figure 4.1, the grid size is controlled by the number of cells in the aerofoil surface. The boundary conditions for the simulation are established on the SU2 configuration file as it is shown in Appendix D. The free-stream velocity is calculated from the Mach number and the thermodynamic state [105]. Also, the direction of the flow is calculated based on the angle of attack established on the configuration file.

The computational grids are structured, body-fitted and clustered around the aerofoil; with a size increment as they are placed away from the center of the domain. A sensitivity analysis regarding the density of the of the computational grid was performed, it was proven from Figure 4.2 that a grid of 599 x 100 cells is required for a change in the lift coefficient of $\Delta C_L \leq 10^{-4}$. To reconstruct all the aerofoils within the Pointwise environment the CST model is solved uploading the CST coefficients of every aerofoil in a Glyph script as shown in Appendix D. The Glyph script C.1 solves the CST parameterisation to reconstruct the aerodoil, generate the mesh, and create the corresponding SU2 mesh file; automating the mesh process as displayed in Figure 4.3. A validation study was performed using the NACA 0012 aerofoil at M = 0.805 compared with experimental data from NASA's Ames High Reynolds Number Facility[107]. The study revealed that the pressure distribution of the upper and lower surface of the aerofoil fits the experimental data with slight deviation as shown in Figure 4.4.

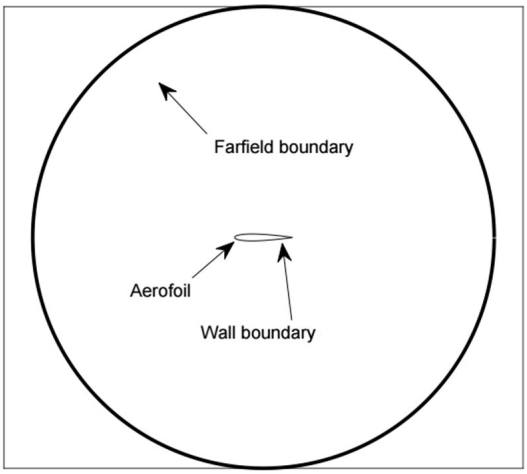

Figure 4.1: Boundary conditions

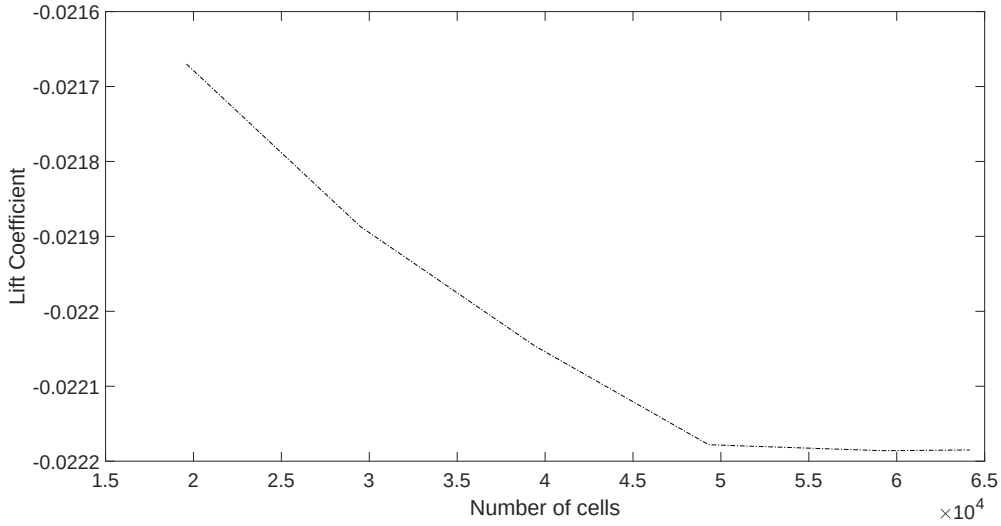

Figure 4.2: Grid convergence study for a NACA 0012 at Mach = 0.801 and angle of attack of a = -0.08; Lift coefficient versus the number of cells.

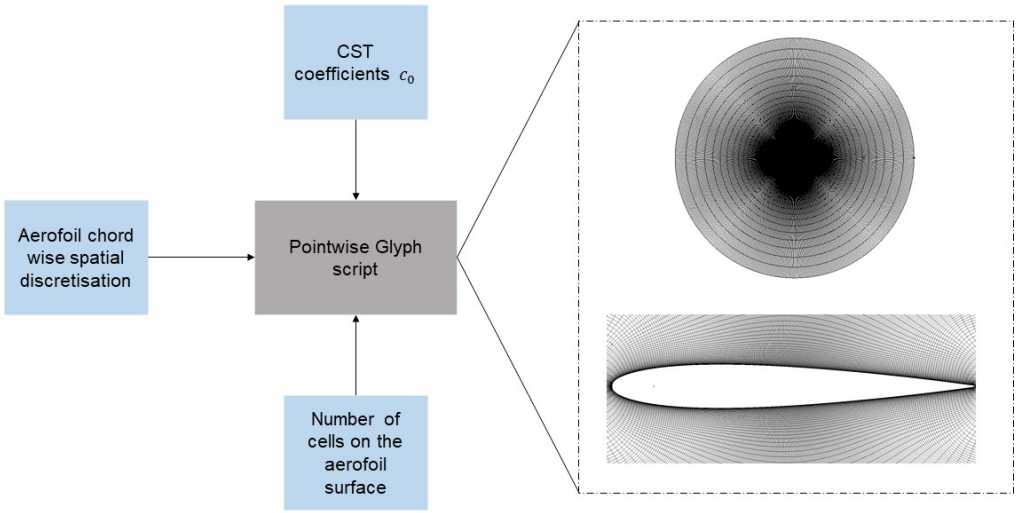

Figure 4.3: Grid generation process: The Glyph code updates the parametric coefficients of the geometries to create the aerofoil shape and automatically generates the mesh.

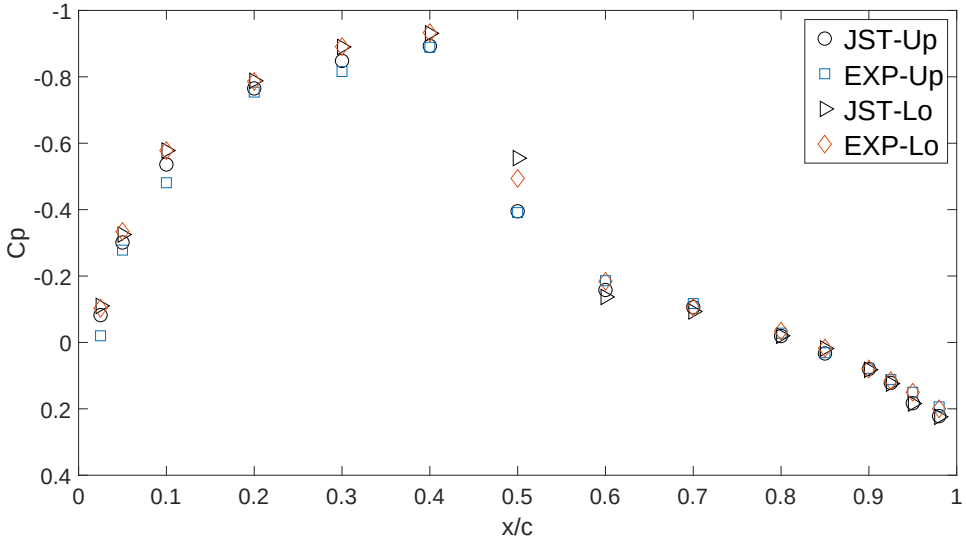

Figure 4.4: Surface pressure coefficient distribution of a NACA 0012 aerofoil at Mach 0.8 and angle of attack of -0.08 degrees compared with experimental data from NASA's Ames High Reynolds Number Facility[107]. Mesh size of 27,681 cells.

4.2 Governing equations

The governing equations that describes the behaviour of the fluid flow around the aerofoil are the conservation laws of mass (continuity equation), momentum and energy , given by the compressible Euler equations in the conservation form [108].

$$\frac{\partial U}{\partial t}+\frac{\partial F}{\partial x}+\frac{\partial G}{\partial y}+\frac{\partial H}{\partial z}=0 \tag{4.1}$$

Where F, G and H are the flux terms, and U is a vector with the dependent variables. These are defined as follows:

$$U=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho w\\ \rho(e+\frac{V^2}{2})\end{bmatrix} \tag{4.2}$$

$$F=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uw\\ \rho u(e+\frac{V^2}{2})+pu\end{bmatrix} \tag{4.3}$$

$$G = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho wv \\ \rho v(e + \frac{V^2}{2}) + pv \end{bmatrix} \tag{4.4}$$

$$H = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ \rho w(e + \frac{V^2}{2}) + pw \end{bmatrix} \tag{4.5}$$

The convective fluxes at each cell face from the finite volume discretisation of the Euler equations, are evaluated using the central scheme JST proposed by Jameson, Schmidt and Turkel [109]. This scheme computes the fluxes from the average of the conservative elements. The JST implements a a second and fourth order flux artificial dissipation to overcome overshoots at shock waves. To accelerate the convergence of the solution a three multi-grid level proposed by Jameson is implemented [110]. In spite of the fact that the upwind schemes are more accurate when resolving shocks, the JST is computationally low-cost. Taking into consideration that for this research 9,480 simulations were performed for database generation and evaluation, the JST scheme was employed to solve the Euler equations. In addition, the JST scheme have been successfully implemented to solve the compressible ($\rho \neq$ constant) Euler equation in several research.

In a paper by Hu et al. [48], an aerodynamic optimisation study of the Busemann type supersonic biplane aerofoils is addressed. Hu successfully implemented the Jameson-Schmidt-Turkel (JST) model to solve the compressible Euler equations and obtain the

aerodynamic characteristics of the aerofoils at supersonic speeds. In a similar study Economon et al. [104] performed a series of CFD simulations for different test cases, which included the inviscid compressible fluid flow simulation of a supersonic aircraft in a flow regime of Mach = 1.6. The solution of the flow was performed by implementing the JST spatial discretisation. Sengupta et al. [111] carried out CFD simulations using different fluid schemes for transonic flow around a NACA 0012 and SHM-1 aerofoils. It was proven that the JST scheme is suitable to capture phenomena that occurs in transonic flows, such as shock waves, drag divergence, and shock-boundary layer interactions for the viscous case. Siegler et al. [112] developed a supersonic aerofoil shape optimisation with variable fidelity, the convective fluxes were calculated using JST scheme.

Chapter 5

Database construction and DNN implementation

The dataset is composed by the aerodynamic coefficients of the aerofoils and the geometry parameters from the parameterisation scheme. The construction of the database is a crucial part of this research due to its influence in the network training. The aerodynamic coefficients are set to be the inputs of the network and the geometry parameters are the outputs.

To create the database and fascilitate the structure of the dataset used to train the network, a series of Python scripts were developed. From Appendix A the Python script A.1 creates a set of directories and sub-directories as shown in Figure 5.1 to store the mesh files, and the configuration files for the SU2 code. To run the SU2 code in all the database directories the Python script A.2 moves the SU2_CFD.exe file to directories and executes it with the specifications from the SU2 configuration file. Subsequently the Python script A.3 reads through all the surface flow solution output files from the CFD simulations to store the lift and drag coefficients in an array of numbers. The aerodynamic coefficients are then separated by Mach numbers using the Python script A.4 to obtain 4 separated arrays, one for each Mach number. Then the aerodynamic coefficients and the CST coefficients then are placed together using the Python script A.5. To conclude the

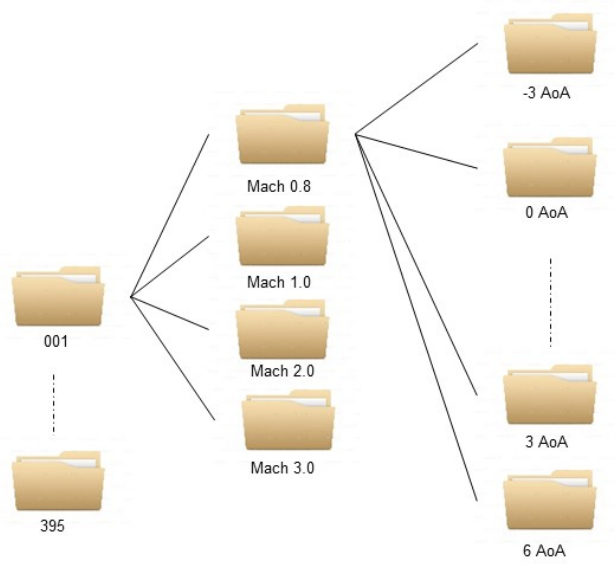

Figure 5.1: Database structure. The aerodynamic coefficients are stored in an structured database of directions and sub-directions.

generation of the dataset used to train the network, the MATLAB script C.1 which can be found in Appendix B, is implemented outputting an matrix where each row represents an aerofoil and each column represents the aerodynamic coefficients for the 4 Mach numbers and the CST coefficients.

5.1 Deep Neural Network implementation on aerofoil inverse design

An important part of the implementation of Neural Networks is the network training process. For this research, the definition of the training process and the network hyperparameters, such as the number of hidden layers, number of neurons on each layer and the learning rate was evaluated. To determine the optimum configuration of these parameters

an evaluation of different configuration was carried out using 4 optimisers: The Stochastic Gradient Descent (SGD) optimiser [113], the Adam optimiser [114], the Adadelta optimiser [115], and the Adagrad optimiser [116]. With the configuration established in Figure 5.2 different network architectures and optimisation algorithms are training and tested to evaluate the error between the network configurations. The network hyperparameters are the optimiser which is a function of $f(O)$ as stated in Equation 5.1, the number of layers which is a function of $f(L) = 1:10$, the number of neurons on each layer specified by $f(N) = 20:10:100$ and the learning rate defined in Equation 5.2

$$f(O) = \begin{cases} \text{SGD} \\ \text{Adam} \\ \text{Adadelta} \\ \text{Adagrad} \end{cases} \tag{5.1}$$

$$f(LR) = 0.1^n, \qquad n = 1:4 \tag{5.2}$$

The neural network attempts to calculate the non-linear links between the aerodynamic coefficients and the parameterisation coefficients. Thus, predicting the geometry parameters for a given set of aerodynamic coefficients.

5.2 Network evaluation

This research hopes to find the interconnections between the aerofoil shape coefficients and its corresponding aerodynamic characteristics by the use of Neural Networks. Thus, it is expected that the framework of this research can obtain immediately an aerofoil shape that fits a given aerodynamic coefficients at transonic and supersonic regimes. The

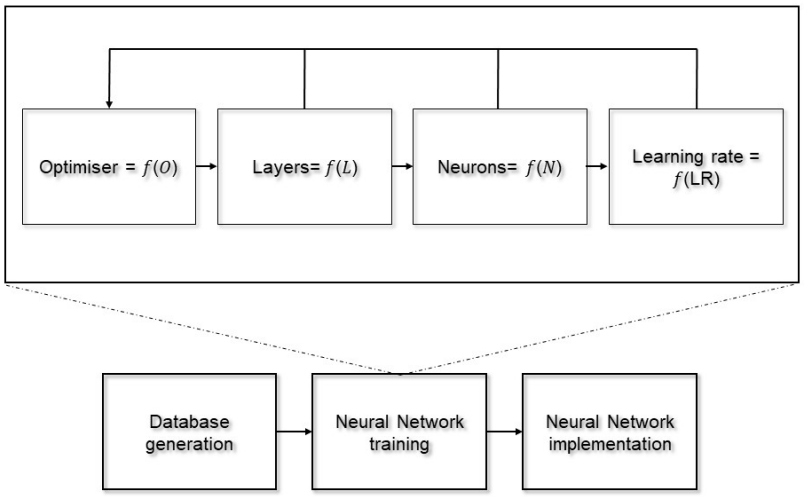

Figure 5.2: Different network architectures are trained with different hyperparameters, such as the optimiser to adjust the weights of the network, the number of layers, the number of neurons on each layer, and the learning rate.

scheme of the network implementation is shown in Figure 5.3. The network was trained using the TensorFlow 1.13.1 version using an Intel Core i7-8700 CPU with a frequency of 3.20GHz. The inputs of the DNN comprised the lift and drag coefficients for a wide range of angles of attack and 4 Mach numbers regimes for each aerofoil. The output of the network contains 20 CST coefficients for every aerofoil.

The evaluation of the optimisers was set for a number of hidden layers ranging from 1 to 10, a number of neurons on each layer ranging from 20 to 100 and 4 different learning rates : 0.1, 0.01, 0.001, and 0.0001. The database was splitted into a training set of 70% of the dataset, a validation set taking 20% of the data, and 10% of the database to test the network model after training. The training process was set for a maximum number of 3000 Epochs with an early stop callback to avoid overfitting. The early stop callback automatically stops the training when the validation score does not present improvements over the last 10 iterations. The results of the test set Mean Squared Error (MSE) compared with the number of layers for a given number of neurons and different learning rates are

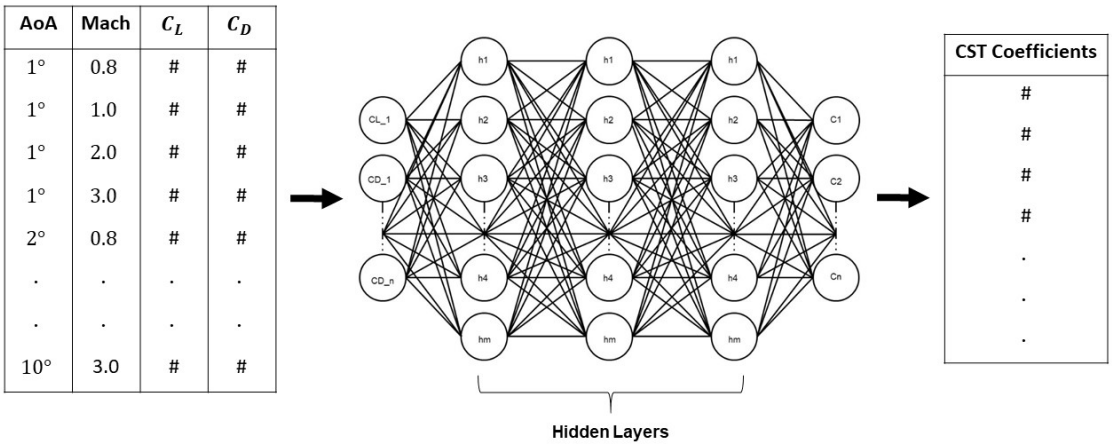

Figure 5.3: Deep Neural Network implementation scheme. The lift and drag coefficients for an specific angle of attack and mach number, are fed into the network in a matrix array. The output of the network is a vector array containing the CST coefficients.

shown in Figure 5.4. The green line represents the configuration with the lowest MSE.

Figure 5.5 presents a more detailed view of the information regarding the MSE for values lower than 0.001. It can be seen from Figure 5.5 that the Adagrad optimiser with learning rates of 0.01, 0.001 and 0.0001, and the Adam optimiser with learning rates of 0.001 and 0.0001. Here again, the green line represents the configuration with the smallest MSE. It can be noted that the most favourable configuration for the smallest MSE, is a network with the Adam optimiser, with a learning rate of 0.0001, 8 hidden layers, and 80 neurons on each hidden layer.

Now, in contrast to Figure 5.5, Figure 5.6 presents a detailed representation filtering out the MSE values lower than 0.5, to visualise the configurations with the worst performance. It can be seen from Figures 5.5 and 5.6 that the learning rate plays a very important role in the accuracy of the prediction of the network. Analysing the Adam and the Adagrad optimisers, both present similar characteristics when increasing the learning rate, it is evident that the MSE escalates. This according to Y. A. LeCun et al. [117] is due to the fact that high learning rates, depending on the curvature of the error surface, can

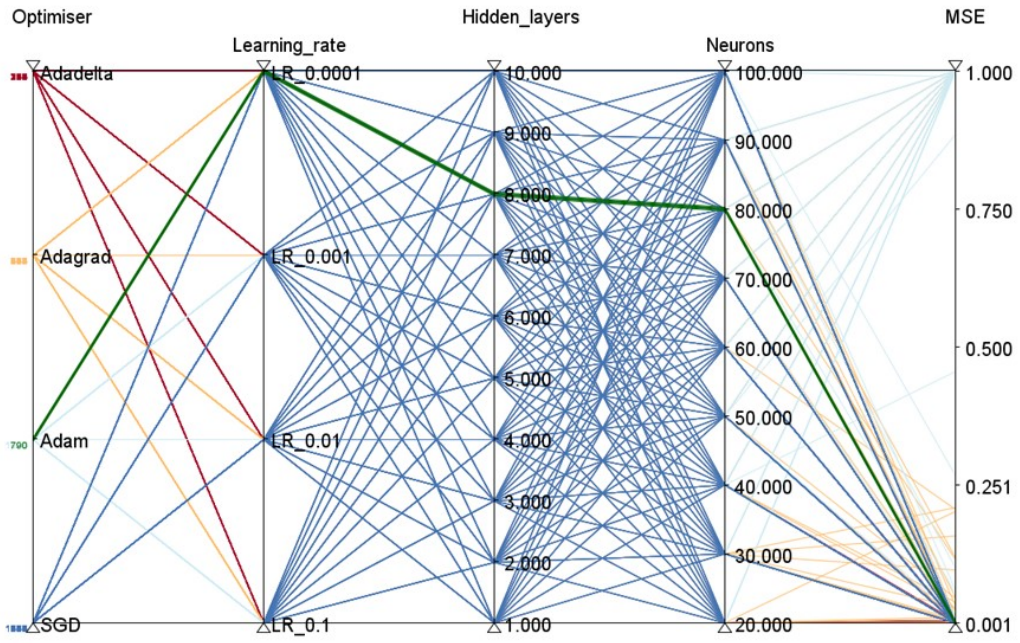

Figure 5.4: Parallel plot coordinates for the different optimisers, learning rates, number of hidden layers, number of neurons on each hidden layer, and MSE. The green line represents the configuration with the smallest MSE

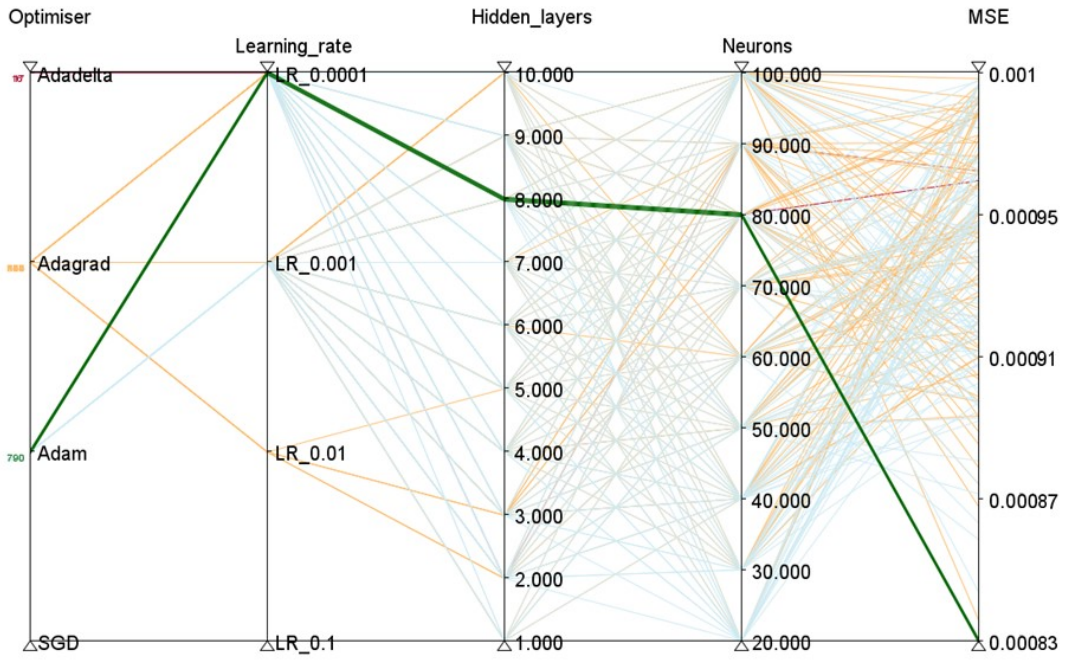

Figure 5.5: Parallel plot coordinates for the different optimisers, learning rates, number of hidden layers, number of neurons on each hidden layer, and MSE. The green line represents the configuration with the smallest MSE. The MSE values bigger than 0.001 are filtered out

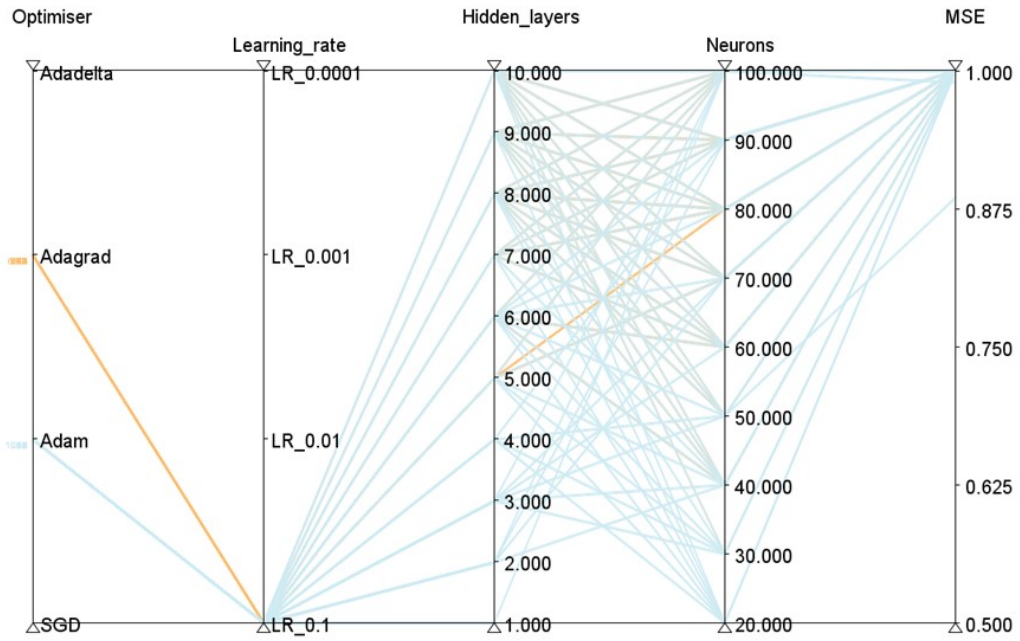

Figure 5.6: Parallel plot coordinates for the different optimisers, learning rates, number of hidden layers, number of neurons on each hidden layer, and MSE. The MSE values smaller than 0.5 are filtered out

lead to accuracy divergence. On the other hand, in the case of the SGD and Adadelta algorithms for the given dataset of aerodynamic and shape coefficients, the learning rate does not influence the MSE excessively. However, for this case, the SGD and Adadelta does not excel in terms of minimum MSE compared with the Adam and Adagrad optimisers.

5.3 DNN configuration for aerofoil inverse design

Based on the network architecture evaluation and taking into consideration the MSE, the network architecture specification for this thesis are showed in Table 6.1. Despite the fact that the Adam optimiser portrays a underperformed prediction when a high learning rate is used, when using a small learning rate it outperforms the remaining optimisers in terms of predicting aerofoil shapes with a small MSE. The best network hyperparameters

Table 5.1: Deep Neural Network hyperparameters specification for a minimum MSE based on the network evaluation.

Network parameter	**Specification**
Number of hidden layers	8
Number of neurons	80
Learning rate	0.0001
Optimiser	Adam

configuration, which gives the smallest MSE (see Figures 5.4 - 5.6) for the given dataset was found to be a DNN with 8 hidden layers containing 80 neurons on each layer, and a learning rate of 0.0001 using the Adam optimiser.

The Gaussian error distribution of the predictions from test set are specified in Figure 5.7. Which suggest that the error of the predictions of the network from the test set are in the majority close to zero, for an exception in few cases where the error goes up to -0.15.

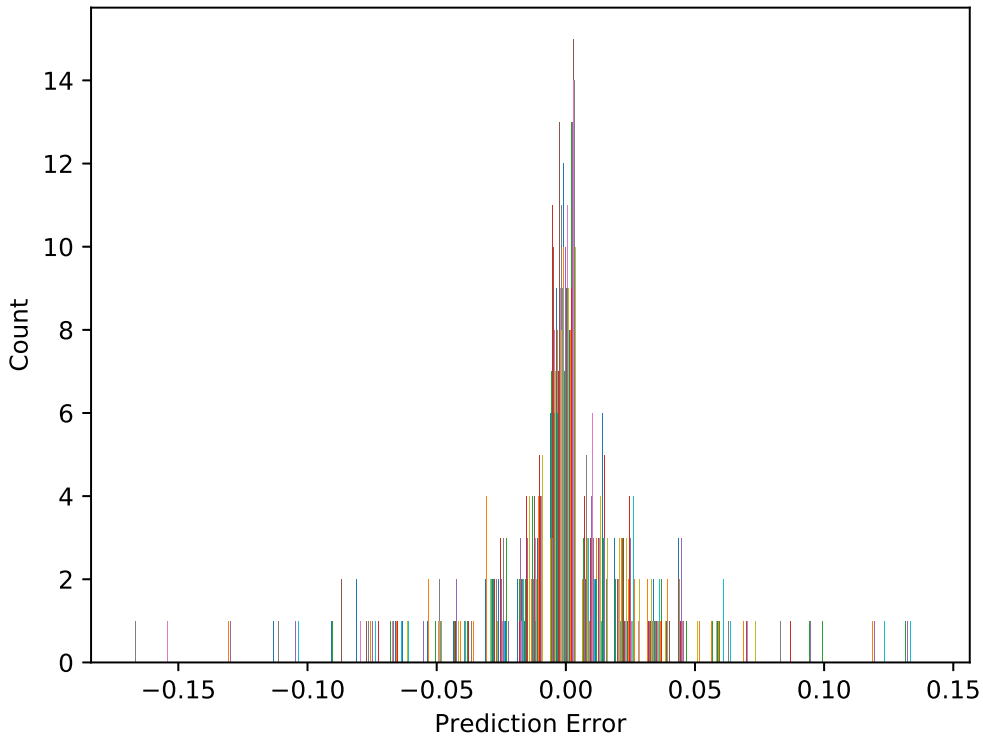

Figure 5.7: Error distribution of the network's prediction using the test set data.

Chapter 6

Aerofoil inverse design

This section is devoted to the implementation and evaluation of the DNN for transonic/supersonic aerodynamic inverse design. Within this section a series of 5 cases with different aerodynamic requirements are established to obtain their corresponding aerofoils shapes. Then a CFD evaluation of the predicted geometries is performed using the Stanford University code to evaluate and validate the DNN predictions. For this section, each case is based on an inverse aerodynamic design of the NACA 66-206 for a target aerodynamic characteristics at different flight conditions, however the implementation of this DNN framework can be applied to any other aerofoil. The NACA 6-series aerofoil family was developed for high-speed performance with the objective of achieving a target drag, critical Mach number, and a maximum lift [4]. The NACA 66-206 particularly exhibits characteristics similar to supercritical aerofoils, being a flat top surface portion of the aerofoil which delays the formation of shock waves and a slightly rounded bottom surface with aft camber to promote lift in the subsonic regime [5].

The number of aerodynamic targets is based upon the number of flight conditions considered. It is important to mention that the design requirements specified in each case of this section are not for a particular specific application, these case studies are rather a more general demonstrative study of the capabilities of the DNN developed by this research. In addition, it is worth mentioning that the Deep Neural Network was trained only

with the aerodynamic coefficients lift and drag. The pithing moment coefficient was not taking into consideration for this first approach, which limits the fidelity of the aerofoils outputted by the network. However, for this initial approach the main objective is to study the capabilities of the DNN to inverse design aerofoils for the required aerodynamic lift and drag coefficients only. The network does not have information regarding other aerodynamic characteristics such as pitching moment, shock wave generation, or pressure distribution. Further investigation taking into account the pitching moment and the remaining aerodynamic information will be needed in future work. Nevertheless, despite the limitations mentioned above, a comparison of the aerodynamics between the baseline aerofoil and the aerofoil predicted by the DNN takes place in this chapter.

6.1 6 objectives aerodynamic inverse design

The objective of this case study is to inverse design an aerofoil based on the NACA 66-206, with a reduction in the drag coefficient C_d by 25% while maintaining or improving the lift coefficient C_l for a single angle of attack and 3 flight velocities, making the inverse design problem for 6 parameters (lift and drag coefficients for 3 velocities and a single angle of attack). The aerofoil original lift and drag coefficients are specified in Table 6.1, where the NACA C_l and NACA C_d are the lift and drag coefficients of the NACA 66-206, and DNN C_l and DNN C_d stands for the lift and drag coefficients of the inverse designed aerofoil. The design requirements are defined in the case bellow:

- NACA C_d reduction by 25%. At angle of attack of 1°, and Mach numbers: 1.0, 2.0, and 3.0
- DNN $C_l \geq$ NACA C_l. At angle of attack of 1°, and Mach numbers: 1.0, 2.0, and 3.0

A comparison of the outcome aerofoil inverse design shape predicted by the DNN, and the original aerofoil geometry can be seen in Figure 6.1. As it is shown in Table

Table 6.1: NACA 66-206 aerodynamic coefficients, DNN aerofoil aerodynamic coefficients, and drag coefficient reduction for a 6 objectives aerofoil design.

AoA	**Mach**	**NACA** C_l	**NACA** C_d	**DNN** C_l	**DNN** C_d	C_d **reduction**%
1	1.0	0.137	0.043	0.146	0.035	18.604
1	2.0	0.030	0.028	0.036	0.026	7.142
1	3.0	0.014	0.025	0.020	0.024	4.000

6.1 the DNN predictions accomplished to reduce the drag coefficients increase of the lift coefficient for the 3 Mach regimes, being the most significant drag reduction the case of Mach 1. The increase in the lift and the reduction in the drag coefficients are due to the difference in the pressure distribution around the aerofoil predicted by the network and the original aerofoil. In the case of Mach 1 as it is shown in Figure 6.2, the NACA 66-2206 presents a negative lift force from the leading edge to a point about 15% of the chord line, this due to the fact that the pressure in the upper surface of the aerofoil is higher than the pressure in the lower surface.

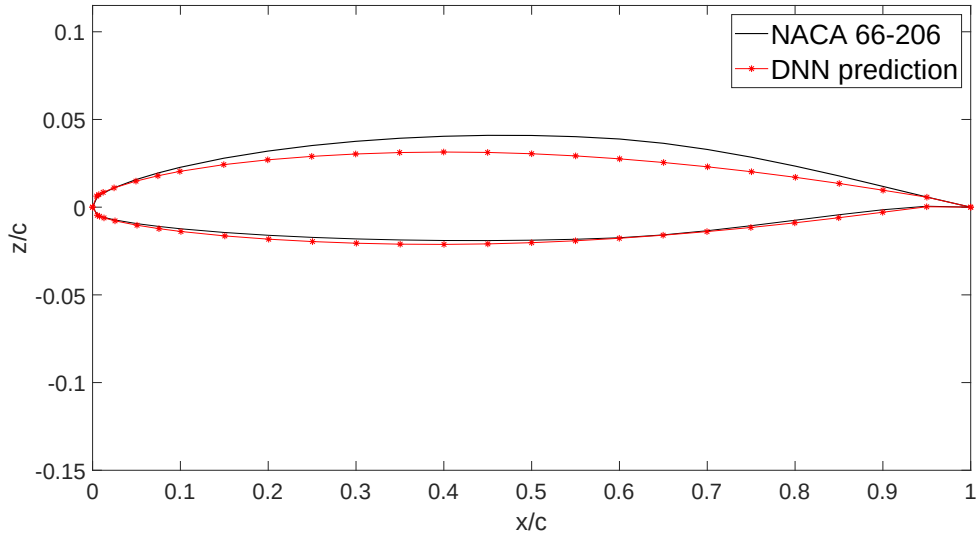

Figure 6.1: Shape comparison between NACA 66-206 and the aerofoil generated by the DNN. Case study: 6 objectives aerofoil design.

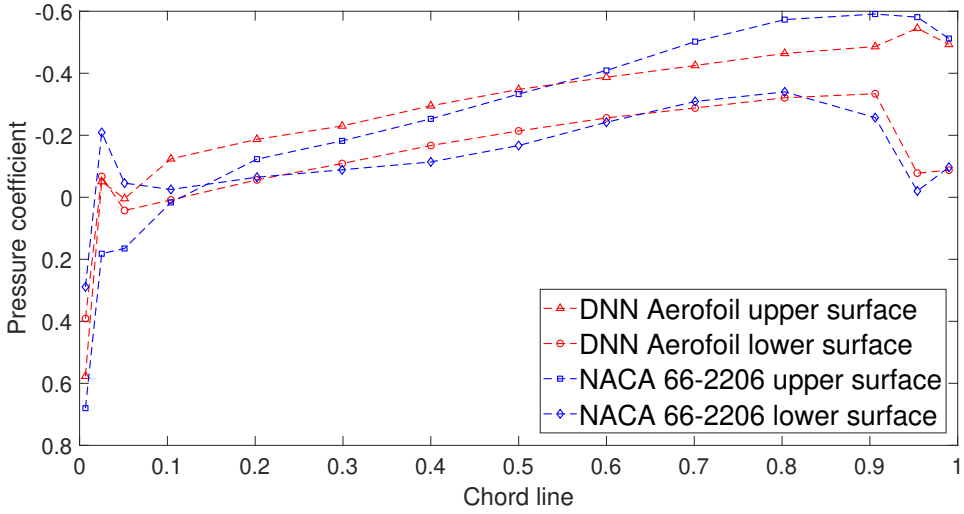

Figure 6.2: Pressure coefficient distribution comparison between NACA 66-206 and the aerofoil generated by the DNN at Mach = 1 and AoA = 1°. For a 6 objectives aerodynamic design requirements.

At the point of 15% of the chord it reaches a zero lift, and then the pressure in the lower surface begins to increase relative to the pressure on the upper surface. In contrast to the NACA 66-206, the aerofoil generated by the DNN presents a small (compared with the original aerofoil) negative lift force from the leading edge to a point at 5% of the chord line. Subsequently, there is an increase in pressure on the lower surface relative to the upper surface. A similar behaviour is observed in the case of Mach 2 and 3, as it is shown in Figure 6.3 for the case of Mach 2. The NACA 66-206 presents a negative lift force from the leading edge to a point about 25% of the chord line. In the other hand, the aerofoil predicted by the DNN present a negative lift force until a point located at 10% of the chord line, then the lower surface pressure becomes higher than the upper surface pressure. Taking into consideration the drag reduction and the increase in the lift coefficient, the lift/drag ratio (at the Mach numbers specified for this case) of the aerofoil predicted by the DNN presents an improvement in comparison with the NACA 66-206 as shown in Figure 6.4.

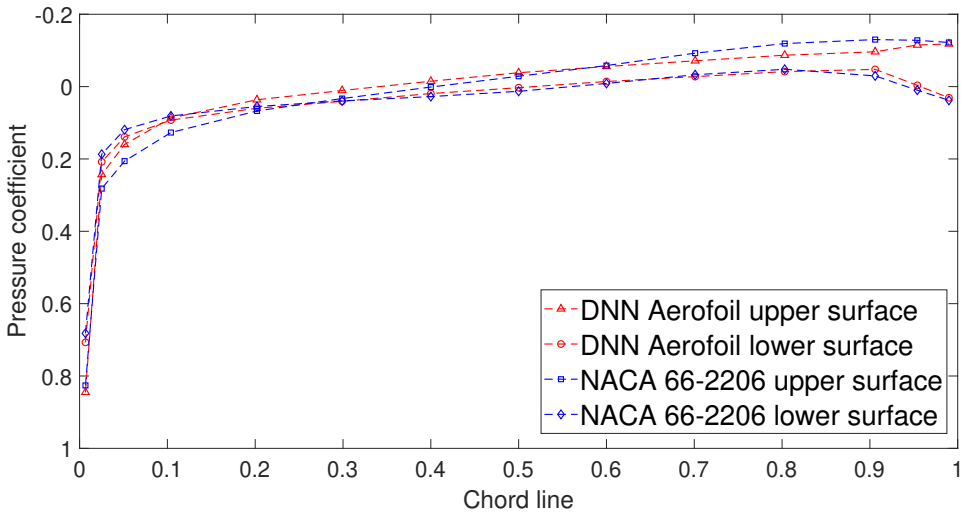

Figure 6.3: Pressure coefficient distribution comparison between NACA 66-206 and the aerofoil generated by the DNN at Mach = 2 and AoA = 1°. For a 6 objectives aerodynamic design requirements.

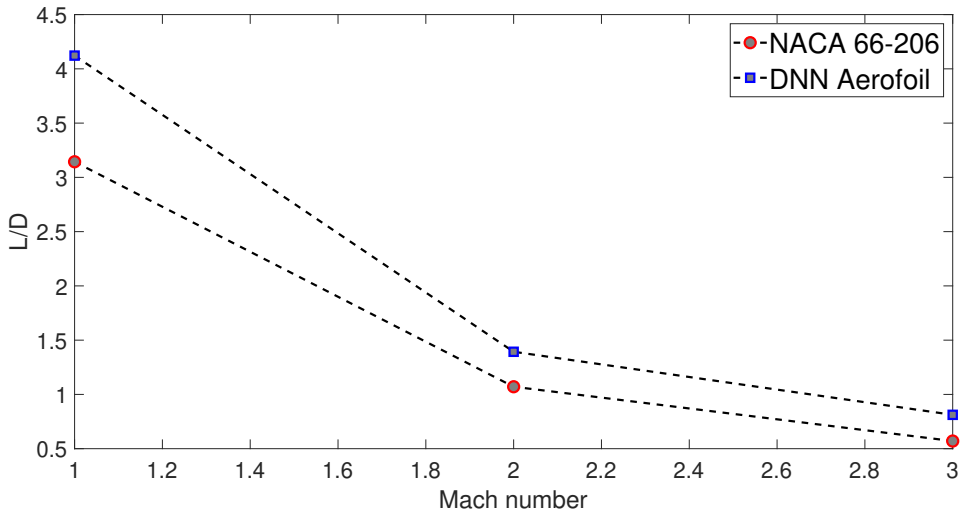

Figure 6.4: Lift/Drag ratio comparison of the NACA 66-206 and the DNN aerofoil prediction. For a 6 objectives aerodynamic design requirements.

6.2 12 objectives aerodynamic design

The objective of this case study is to inverse design an aerofoil based on the NACA 66-206, with a reduction in the drag coefficient C_d by 25% while maintaining or improving

the lift coefficient C_l for a single Mach number of 2.0, and 6 different angles of attack. This makes the inverse design problem of 12 parameters (lift and drag coefficients for 6 different angles of attack and a single flow velocity). The aerofoil original lift and drag coefficients are specified in Table 6.2, where the NACA C_l and NACA C_d are the lift and drag coefficients of the NACA 66-206, and DNN C_l and DNN C_d stands for the lift and drag coefficients of the inverse designed aerofoil. The design requirements are defined in the case bellow:

- NACA C_d reduction by 25%. At Mach number of 2.0, and angles of attack: -3°, 0°, 1°, 2°, 3°, 6°
- DNN $C_l \geq$ NACA C_l. At Mach number of 2.0, and angles of attack: -3°, 0°, 1°, 2°, 3°, 6°

The comparison of the NACA 66-206 shape and the aerofoil predicted by the DNN are shown in Figure 6.5. It can be notice that near the trailing approximately at 95% of the chord line, in the intrados of the aerofoil there is an abrupt change in the geometry, leading to a non-smooth path from a point of 90% to the trailing edge. Despite the improvements on the lift and drag coefficients, this abrupt change in the geometry could lead to aerodynamic deficiencies. This highlights the limitations of evenly spaced parameterisation schemes such as the CST, this issue will be tackle in future work.

Taking into consideration the limitations mentioned above, and acknowledging that the NACA C_d was not reduced by 25%. In this inverse design case the aerodynamic coefficients of the NACA 66-206 were improved for 12 flight configurations. As it is shown in Table 6.2 the drag coefficients were reduced at all the flight angles of attack, and in addition the lift coefficients were improved. Similarly to the 6 objectives case, the improvements in the lift and drag coefficients are due to the pressure distribution around the aerofoils.

Table 6.2: NACA 66-206 aerodynamic coefficients, DNN aerofoil aerodynamic coefficients, and drag coefficient reduction for a 12 objectives aerofoil design.

AoA	**Mach**	**NACA** C_l	**NACA** C_d	**DNN** C_l	**DNN** C_d	C_d **reduction**%
-3	2.0	-0.124	0.035	-0.122	0.029	17.142
0	2.0	-0.007	0.027	-0.005	0.022	18.518
1	2.0	0.030	0.028	0.032	0.022	21.428
2	2.0	0.06	0.030	0.071	0.024	20.000
3	2.0	0.108	0.033	0.110	0.027	18.181
6	2.0	0.224	0.050	0.227	0.045	10.000

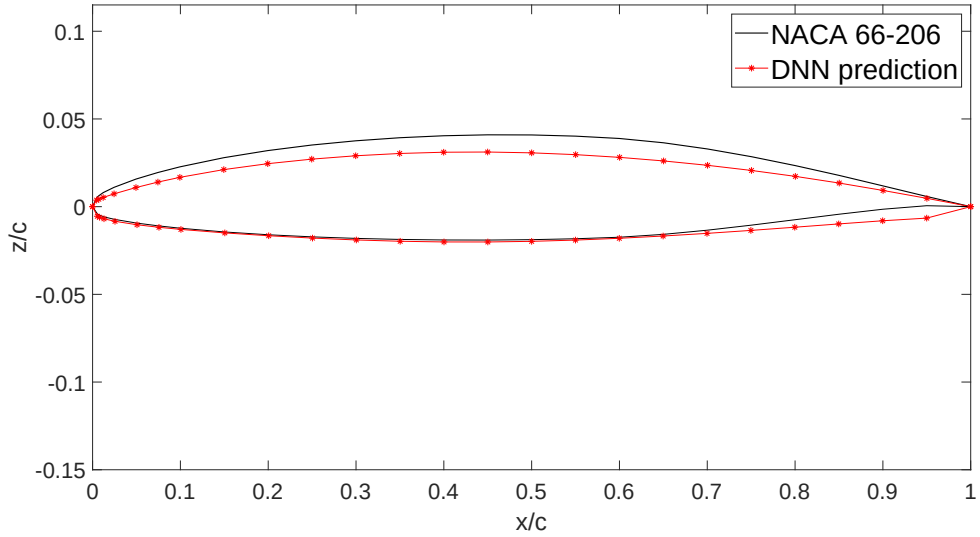

Figure 6.5: Shape comparison between NACA 66-206 and the DNN prediction. Case study: 12 objectives aerofoil design.

As it is shown in Figure 6.6, the pressure distribution around the NACA 66-206 at Mach 2.0 and angle of attack of 2° presents a negative pressure distribution from the leading edge to a point at 10% of the chord line. In contrast with the NACA 66-206, the aerofoil predicted by the DNN presents a positive pressure distribution e. g. the pressure in the lower surface is higher than the pressure in the upper surface in all of the chord line. This behaviour is maintained for all the flight configurations required for this optimisation case. In the case of the Mach 2.0 and angle of attack of 6° flight configuration, the increase in the lift and the drag reduction are not as notorious as the others, due to the pressure distribution similarities of the two aerofoils. Only from the leading edge to a point at 20%

of the chord line, the aerofoil predicted by the DNN presents a higher pressure distribution as it is observed in Figure 6.7. This due to the larger drop of pressure on the upper surface compared with the NACA 66-206.

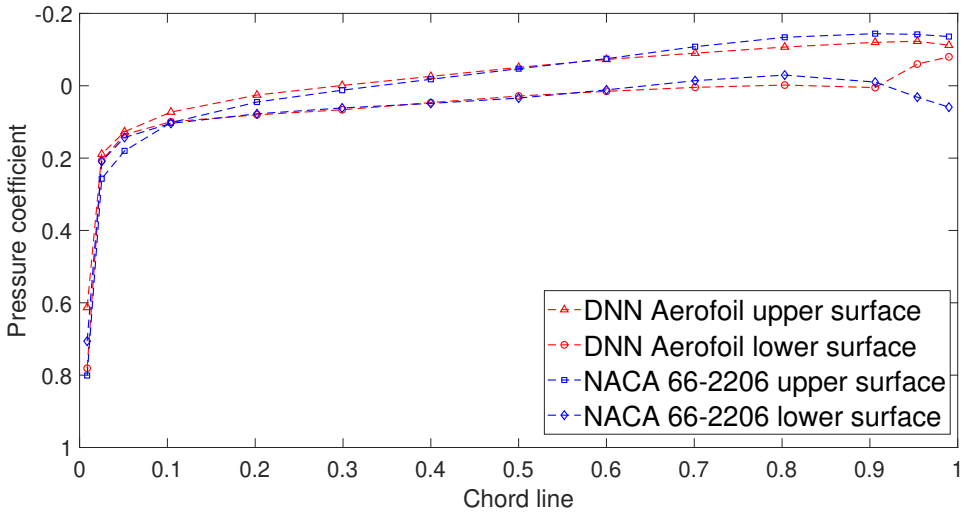

Figure 6.6: Pressure coefficient distribution comparison between NACA 66-206 and the DNN aerofoil prediction at Mach = 2 and AoA = 2°. For a 12 objectives aerodynamic design requirements.

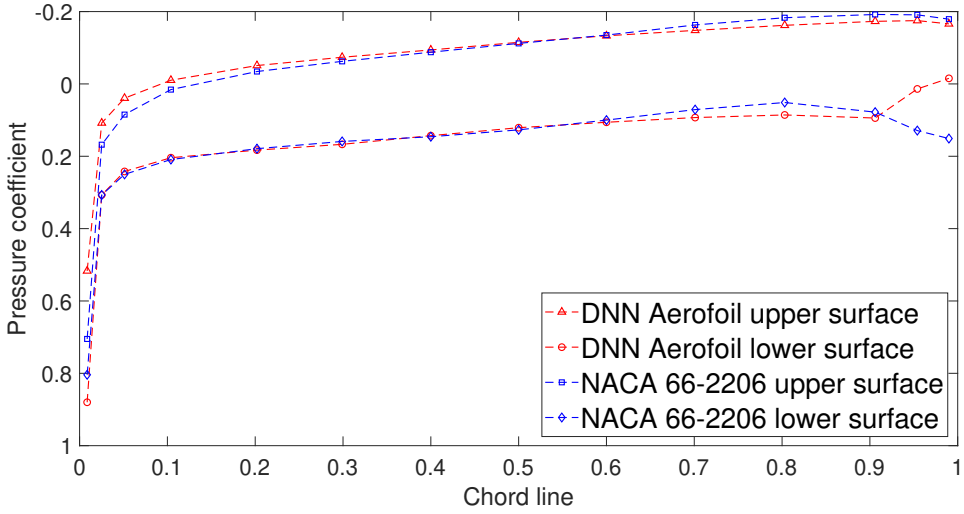

Figure 6.7: Pressure coefficient distribution comparison between NACA 66-206 and the DNN aerofoil prediction at Mach = 2 and AoA = 6°. For a 12 objectives aerodynamic design requirements.

The influence of the drag reductions and the increase in the lift coefficients for this case study are portrayed in Figure 6.8. In Figure 6.8 the lift/drag ratios for all the flight

conditions are evaluated, and it is shown that the ratio is improved in the aerofoil predicted by the DNN. However, when evaluating Figures 6.6 and 6.7, there is a notorious drop in the pressure coefficient on the lower surface of the aerofoil predicted by the DNN at a point about 90% of the chord line. This may have a direct relationship to the aerofoil geometry as it is observed in Figure 6.5, the abrupt change in the geometry as mentioned before, affects the flow near the trailing edge leading to a loss of pressure on the intrados of the aerofoil, this can lead to aerodynamic inefficiencies in further investigation. The pressure contours of the NACA 66-206 and the aerofoil inverse designed using the DNN, at Mach 2 and an angle of attack of 2° are portrayed in Figures 6.9 and 6.10.

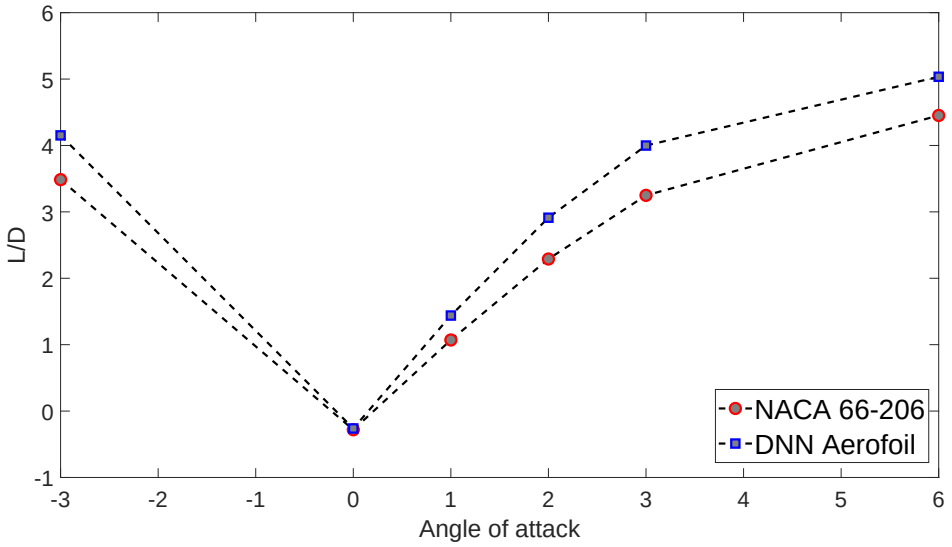

Figure 6.8: Lift/Drag ratio evaluation of the NACA 66-206 and the DNN aerofoil prediction. For a 12 objectives aerodynamic design requirements.

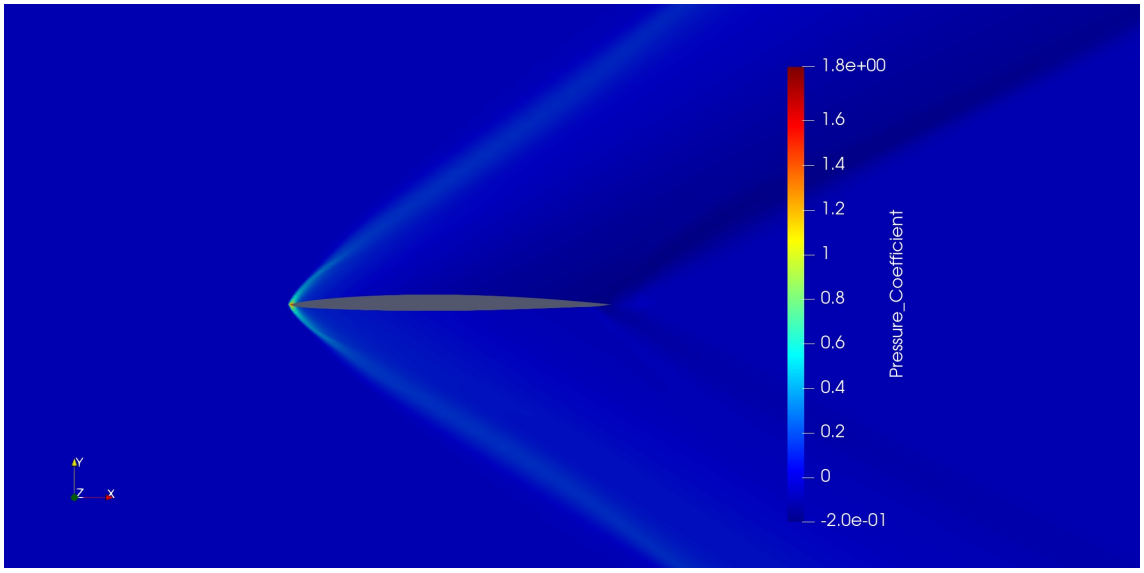

Figure 6.9: Pressure contour at Mach = 2 and AoA = 2°. Aerofoil predicted by the DNN

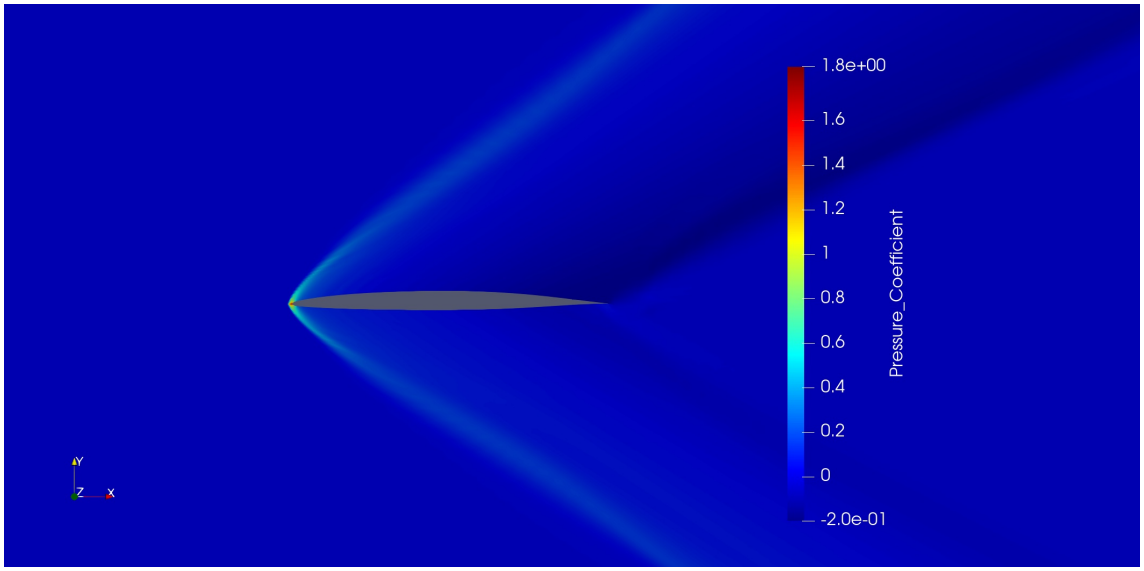

Figure 6.10: Pressure contour at Mach = 2 and AoA = 2°. NACA 66-206

6.3 16 objectives aerodynamic design

The objective of this case study is to inverse design an aerofoil based on the NACA 66-206, with a reduction in the drag coefficient C_d by 25% while maintaining or improving the lift coefficient C_l for 4 Mach numbers, and 2 angles of attack. This makes an inverse

Table 6.3: NACA 66-206 aerodynamic coefficients, DNN aerofoil aerodynamic coefficients, and drag coefficient reduction for a 16 objectives aerofoil design.

AoA	**Mach**	**NACA** C_l	**NACA** C_d	**DNN** C_l	**DNN** C_d	C_d **reduction%**
1	0.8	0.592	0.011	0.524	0.008	27.272
1	1.0	0.137	0.043	0.141	0.034	20.930
1	2.0	0.030	0.028	0.037	0.024	14.285
1	3.0	0.014	0.025	0.020	0.023	8.000
2	0.8	0.942	0.036	0.895	0.027	25.000
2	1.0	0.274	0.050	0.282	0.040	20.000
2	2.0	0.069	0.030	0.075	0.026	13.333
2	3.0	0.038	0.026	0.043	0.024	7.692

design problem for 16 parameters (lift and drag coefficients for 2 different angles of attack and a 4 Mach numbers). The aerofoil original lift and drag coefficients are specified in Table 6.3, where the NACA C_l and NACA C_d are the lift and drag coefficients of the NACA 66-206, and DNN C_l and DNN C_d stands for the lift and drag coefficients of the inverse designed aerofoil. The design requirements are defined in the case bellow:

- NACA C_d reduction by 25%. At Mach numbers: 0.8, 1.0, 2.0 and 3.0, and angles of attack: 1°, 2°
- DNN $C_l \geq$ NACA C_l. At Mach numbers: 0.8, 1.0, 2.0 and 3.0, and angles of attack: 1°, 2°

The shapes of both aerofoils can be observed in Figure 6.11. For this study case with a considerable high number of flight conditions, the aerofoil generated by the DNN accomplished to reduce the drag coefficient in all of the flight conditions required for this inverse design. However, in the case of the transonic Mach 0.8, for both 1° and 2° angles of attack, there is a relatively small decrease in the lift coefficient.

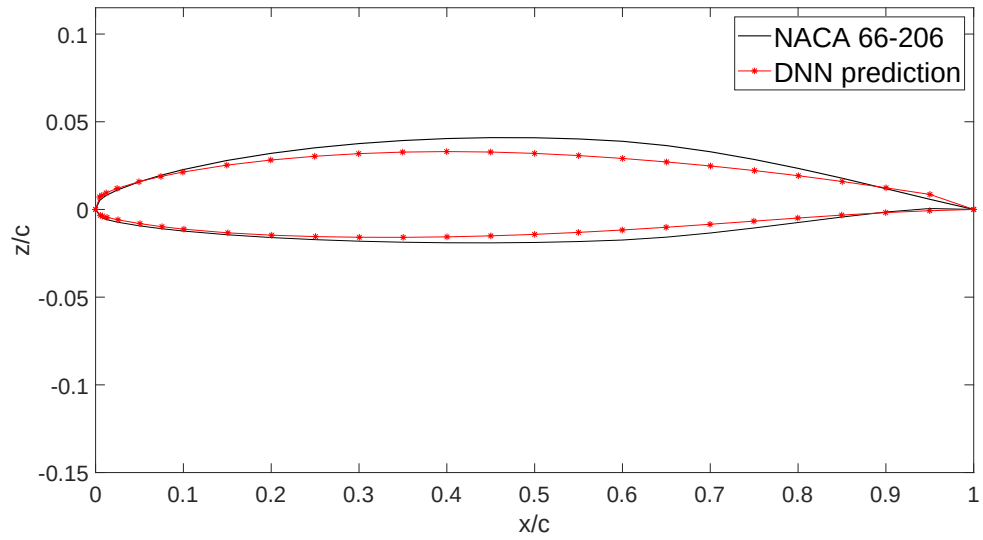

Figure 6.11: Shape comparison between NACA 66-206 and the DNN prediction. Case study: 16 flight conditions aerofoil design.

As shown in Figure 6.12 the pressure increase due to a shock wave generated on the upper surface of the NACA 66-206 is delayed to a point around 75% of the chord, whereas the shock wave on the upper surface of the aerofoil generated by the DNN is located at a point around 65% of the chord line. This delay in the shock wave generation in the NACA 66-206 increase the the area inside the pressure distribution around the aerofoil, thus achieving more lift force than the DNN aerofoil. However, the abrupt increase in the pressure of the upper surface of the NACA 66-206, after the generated shock wave is substantially larger compared with the raise in pressure on the upper surface of the DNN aerofoil. Hence, the shock wave generated by the NACA 66-206 is stronger than the shock wave generated by the DNN aerofoil prediction, for this reason the DNN aerofoil achieved to reduce the drag coefficient up to 27.272%.

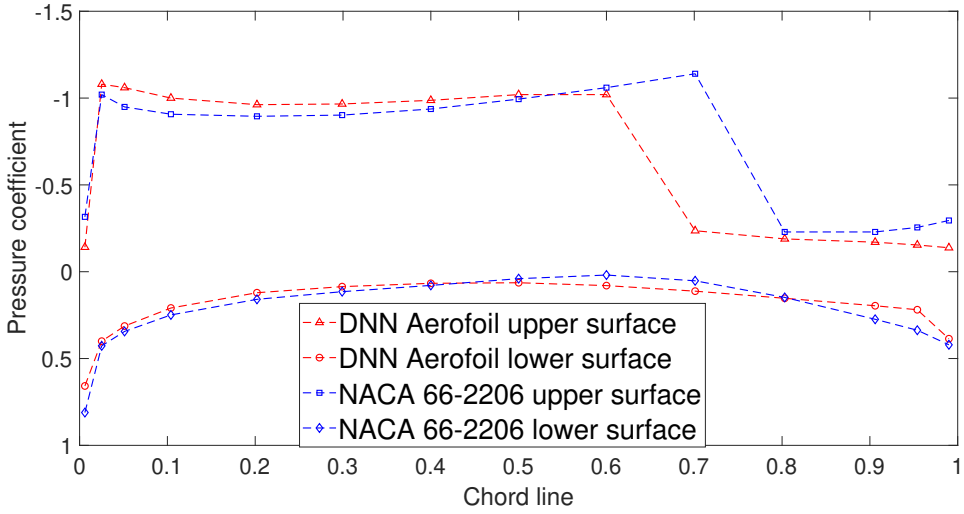

Figure 6.12: Pressure coefficient distribution comparison between NACA 66-206 and the DNN aerofoil prediction at Mach = 0.8 and AoA = 2°. For a 16 objectives aerodynamic design requirements.

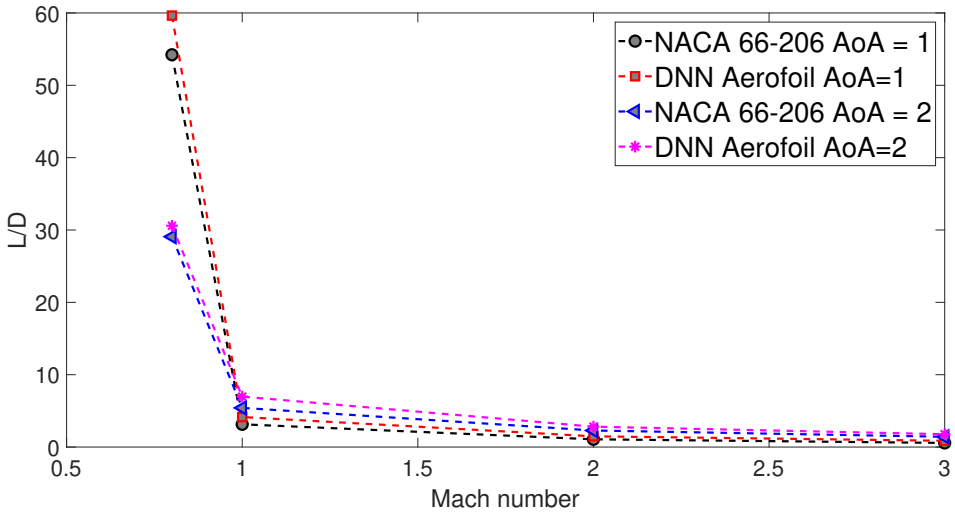

Figure 6.13: Lift/Drag ratio evaluation with the Mach number of the NACA 66-206 and the DNN aerofoil prediction at angles of attack 1° and 2°. For a 16 objectives aerodynamic design requirements.

For Mach numbers 1, 2 and 3; similarly to the previous study cases the increase in the lift coefficients are due to the difference in pressure distribution around the aerofoils. Although for the transonic case of Mach 0.8 the lift coefficient slightly decreased, the behaviour of the pressure distributions around the aerofoil predicted by the DNN presented an improvement against the NACA 66-206. The overall lift/drag coefficients ratio for this

case study is presented in Figure 6.13, it is proved that for both angles of attack 1° and 2° the lift/drag ratio is improved for all the Mach numbers in the inverse design.

6.4 24 objectives aerodynamic design

The objective of this case study is to inverse design an aerofoil based on the NACA 66-206, with a reduction in the drag coefficient C_d by 25% while maintaining or improving the lift coefficient C_l for 2 Mach numbers, and 6 angles of attack. Making the inverse design problem for 24 parameters (lift and drag coefficients at 6 different angles of attack and 2 Mach numbers). The aerofoil original lift and drag coefficients are specified in Table 6.4, where the NACA C_l and NACA C_d are the lift and drag coefficients of the NACA 66-206, and DNN C_l and DNN C_d stands for the lift and drag coefficients of the inverse designed aerofoil. The design requirements are defined in the case bellow:

- NACA C_d reduction by 25%. At Mach numbers: 1.0 and 3.0, and angles of attack: -3°, 0°, 1°, 2°, 3°, 6°
- DNN $C_l \geq$ NACA C_l. At Mach numbers: 1.0 and 3.0, and angles of attack: -3°, 0°, 1°, 2°, 3°, 6°

The results of this aerodynamic multi-objective design are summarised in Table 6.4. The aerofoil predicted by the DNN is presented in Figure 6.14. Unlike the cases studied previously in this paper, this design case presents moderate decrease in the lift coefficients in 5 of the 12 flight configurations required for the design optimisation. These reductions in the lift coefficients are relatively small, being the most substantial a decrease of of 8% for the first flight configuration in Table 6.4.

Table 6.4: NACA 66-206 aerodynamic coefficients, DNN aerofoil aerodynamic coefficients, and drag coefficient reduction for a 16 objectives aerofoil design.

AoA	**Mach**	**NACA** C_l	**NACA** C_d	**DNN** C_l	**DNN** C_d	C_d **reduction%**
-3	1	-0.362	0.062	-0.393	0.053	14.516
-3	3	-0.078	0.030	-0.078	0.023	23.333
0	1	-0.008	0.041	-0.003	0.031	24.390
0	3	-0.008	0.025	-0.007	0.018	28.000
1	1	0.137	0.043	0.130	0.033	23.255
1	3	0.014	0.025	0.016	0.018	28.000
2	1	0.274	0.050	0.262	0.039	13.750
2	3	0.038	0.026	0.039	0.019	26.923
3	1	0.401	0.060	0.393	0.049	18.333
3	3	0.061	0.028	0.063	0.021	25.000
6	1	0.803	0.117	0.772	0.106	9.401
6	3	0.132	0.038	0.136	0.031	18.421

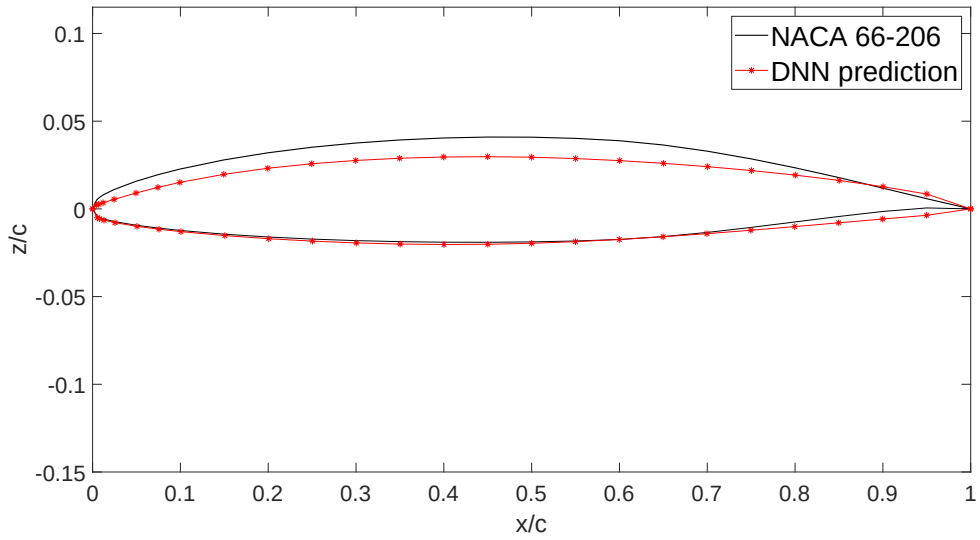

Figure 6.14: Shape comparison between NACA 66-206 and the DNN prediction. Case study: 24 objectives aerofoil design.

For the case of Mach 1 and an angle of attack of 6°, there is a 3.8% lift coefficient reduction. This reduction is accounted by the behaviour of the pressure distribution around the aerofoil generated by the DNN as shown in Figure 6.15. In Figure 6.15 there is a decrease in the pressure on the lower surface of the aerofoil predicted by the DNN from a point around 50% of the chord line to a point around 80% of the chord line. A similar behaviour is presented for the remaining cases where the lift is decreased where the flow

velocity is in the regime of Mach 1. In the case of Mach 3, all of the lift coefficients are improved or maintained, this because the drop in pressure near the leading edge of the aerofoils generated by the DNN is higher than the NACA 66-206, as it is observed in Figure 6.16 for the case of Mach 3 and angle of attack of 3°. Despite the slight reduction in the lift coefficient presented in some of the flight configuration requirements, significant reduction in the drag coefficient was achieved by the DNN aerofoil in all of the case requirements. In all the design objectives the drag was successfully reduced up to 28%. This important drag reduction improves the lift/drag ratio as presented in Figure 6.17. The pressure contours of the NACA 66-206 and the aerofoil inverse designed using the DNN, at Mach 1 and an angle of attack of 6° are portrayed in Figures 6.18 and 6.19.

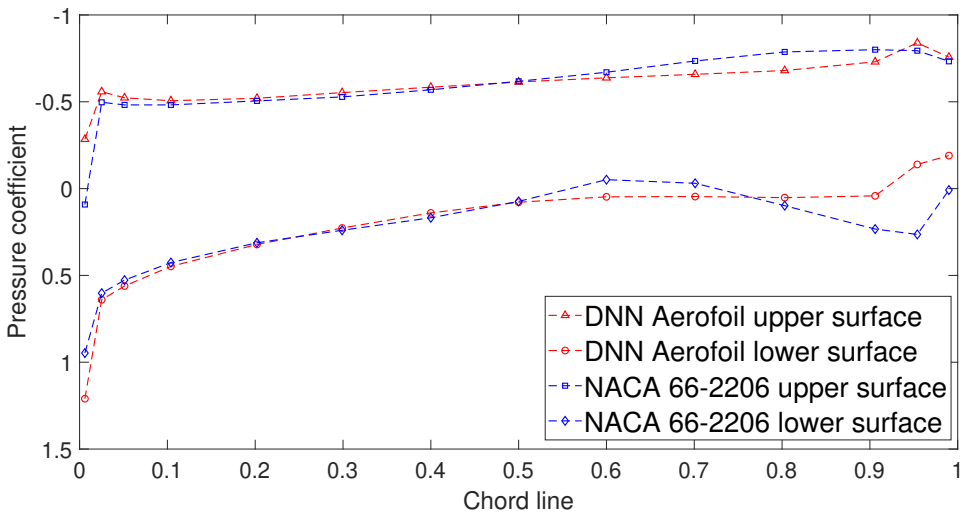

Figure 6.15: Pressure coefficient distribution comparison between NACA 66-206 and the DNN aerofoil prediction at Mach = 1 and AoA = 6°. For a 24 objectives aerodynamic design requirements.

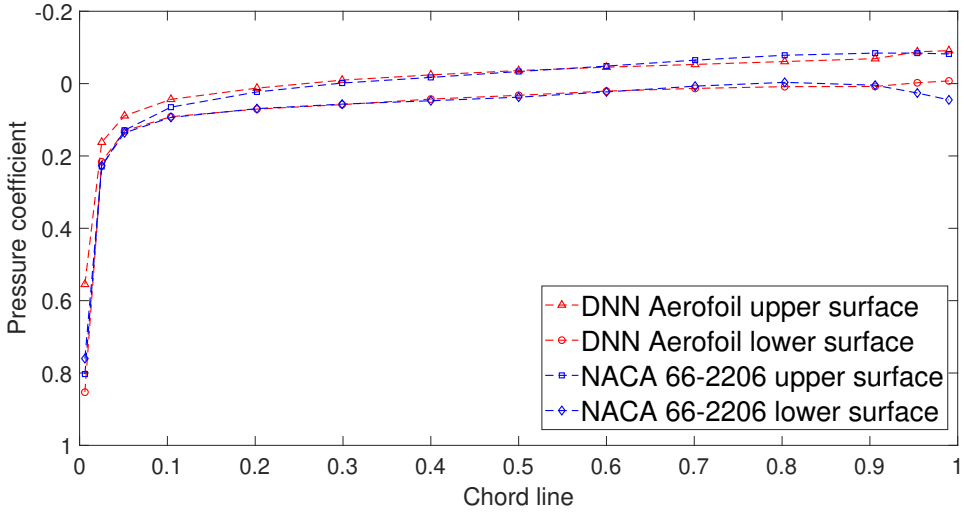

Figure 6.16: Pressure coefficient distribution comparison between NACA 66-206 and the DNN aerofoil prediction at Mach = 3 and AoA = 3°. For a 24 objectives aerodynamic design requirements.

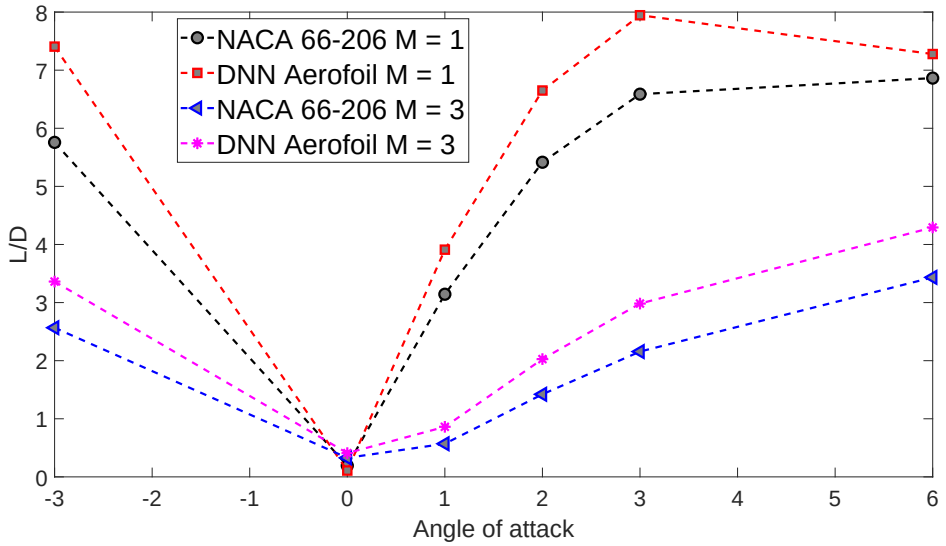

Figure 6.17: Lift/Drag ratio evaluation with the angle of attack of the NACA 66-206 and the DNN aerofoil prediction at Mach numbers of 1 and 3. For a 24 objectives aerodynamic design requirements.

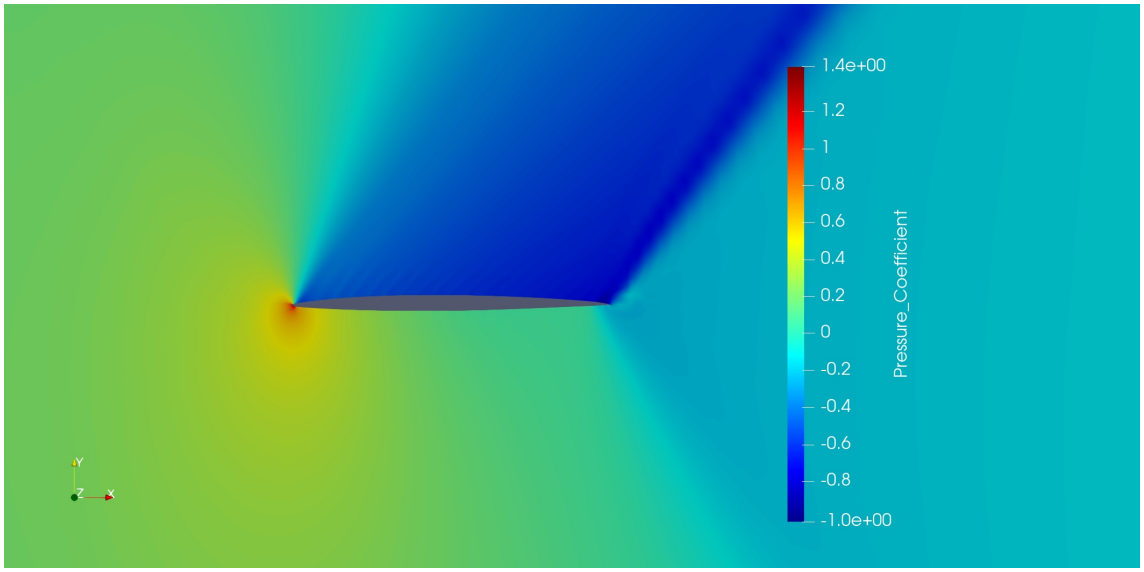

Figure 6.18: Pressure contour at Mach = 1 and AoA = 6°. Aerofoil predicted by the DNN

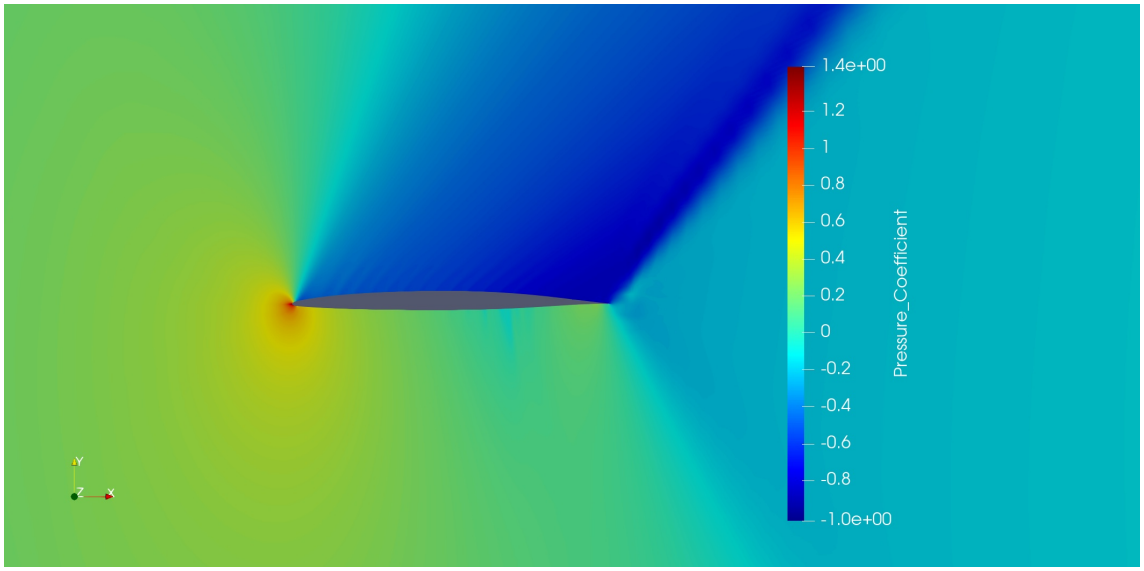

Figure 6.19: Pressure contour at Mach = 1 and AoA = 6°. NACA 66-206

Chapter 7

Conclusions

The present thesis aimed the research of the DNN's capability to inverse design two-dimensional aerofoils for a specific lift and drag coefficient at different flight conditions. Through this thesis a framework based on DNN, for the inverse design of aerofoils with up to 24 different combinations of Mach numbers and angles of attack was established. Taking a step forward to enhance the design process by introducing deep learning models for transonic and supersonic aerofoils. For this research, different neural network's hyperparameters were modified and analysed to obtain the most efficient network structure, varying the network optimisers, number of layers, learning rate, and number of neurons on each layer; a DNN composed by 8 layers and 80 neurons on each layer, with the Adam optimiser and a learning rate of 0.0001 was found to be the most effective for this research.

Despite the fact that in most of the flight configuration requirements stated in the results section, the aerofoil generated by the DNN did not reached in totality a reduction in the drag coefficient of 25%, the inverse design approach developed in this research does accomplish to obtain aerofoils shapes for a high number of flight conditions, which in most of the cases have improved aerodynamic coefficients compared with the baseline aerofoil. In addition, it can be observed that in all of the inverse design cases, there is a tendency of decreasing the thickness of the baseline aerofoil to reduce the drag coefficient. Thus, it can be concluded that the DNN learned from the dataset that the aerofoils

with lower drag coefficients at high Mach numbers have a tendency to be thinner. This outcome is of special importance because it indicates that, taking into consideration that the dataset does not contain any information regarding geometry constrains or aerodynamic constrains such as moment coefficients. The network have the capacity to relate different aspects of the geometry such as the thickness to the aerodynamic coefficients, and can produce an initial guess of a geometry based on desired lift and drag coefficients. Nevertheless, further research which includes implementing geometry and aerodynamic constraints must be done in order to improve the outcome of the DNN. It is worth mentioning that direct design optimisation using gradient-based or gradient-free algorithms could have led to more accurately results in terms of finding optimum shapes for a single flight condition. However, for the application studied in this thesis which involves a wide range of flight conditions, further study needs to be done in order to assess the different types of design optimisation algorithms for this application.

Another aspect to take into consideration is that the difference between the DNN predicted aerofoil aerodynamic coefficients and the CFD computede aerodynamic coefficients, is a function of the error distribution displayed in Figure 5.7 and it is influenced by the construction of the database. For this research the database contains 395 aerofoils samples, which is a small number considering the dimensions stated in the results section, where the dataset dimension accounts for all the CST coefficients and the aerodynamic coefficients. The error distribution displayed in Figure 5.7 can be improved by expanding the number of samples within the database. In spite of the fact that the accuracy of the surrogate model implemented in this thesis has room for improvement via modification of the database size, the inverse design method applied in this research aids the designer to get an aerofoil that fairly fits a high number of aerodynamic coefficients requests, up to 24 coefficients. In addition, it was shown that for all the different design requirements cases in Chapter 6 the aerofoils generated by the DNN did improve the aerodynamic coefficients of the original NACA 66-206 by enhancing the lift to drag ratio. This introduces a wide range of applications for the network designed during this research and shows the

prospect of its use in aerodynamic inverse design.

7.1 Future work

The study conducted in this thesis was the implementation of DNN for inverse aerodynamic design of transonic/supersonic aerofoils, assuming an inviscid flow. Future work requires profound investigation on the effect of the DNN outcome using the CST method with less design parameters, meaning a reduction in the Bernstein polynomials. A reduction in the design parameters is translated in a reduction on the dataset dimension, which is used to train the DNN. Thus, reducing the design parameters have an impact on the accuracy output of the prediction from the DNN. In addition, future works requires the study of the effect on the DNN output from different parameterisation scheme such as the PARSEC method. It was encountered the limitations of the CST method which can be tackled by the use of a different parameterisation method. For future work the implementation of structural constraints such as minimum thickness is taken into consideration, and the effect such constraints have in the accuracy of the output of the DNN. A comprehensive investigation of the DNN for aerofoil inverse design with structural constrains can offer a more realistic output from the DNN. In addition, future works requires investigation of the DNN accuracy when including the momentum aerodynamic coefficient, this will improve the outcome to a more realistic aerofoil inverse design.

Once the future work mentioned above is concluded, the applications of Deep Learning techniques for three-dimensional wing inverse design for transonic/supersonic regimes can be analysed. The research workflow would expand upon the current two-dimensional approach conducted by this thesis. A proper wing shape parameterisation have to be implemented in order to achieve the necessary degrees of freedom of the wing shape configuration and for the necessary dimension reduction of the database. An investigation on different meshing techniques needs be carried out for an efficient capture of the fluid flow properties around the wing, such as shock wave/boundary layer interactions. Thus,

for future work the effects of the viscosity on the transonic and supersonic regime needs to be taken into consideration. A CFD solver will be implemented to solve the Navier–Stokes equations to describe the fluid properties around the supersonic wing. However, it is important to consider the computational cost of conducting CFD simulations of different wing configurations to develop the database, taking into account the accessibility of HPC nodes. Based on the aerodynamic characteristics obtained from the CFD solutions a complete and intelligible three-dimensional database need to be constructed. Yet, a database containing three dimensional geometric parameters and aerodynamic forces suggests a complex dataset matrix array configuration. Finally, a study of different DNN architectures applied for wing inverse design optimisations makes the most attractive part of the future work.

References

[1] Y. Sun, H. Smith, Review and Prospect of Supersonic Business Jet Design, Progress in Aerospace Sciences 90 (2017) 12–38. doi:10.1016/j.paerosci.2016.12.003.

[2] S. Kawa, J. Anderson, S. Baughcum, R. C. Cohen, D. E. Kinnison, P. Newman, J. Rodriquez, R. S. Stolarski, D. W. Waugh, C. Brock, W. Brune, Assessment of the Effects of High-Speed Aircraft in the Stratosphere: 1998, Tech. Rep. NASA/TP-1999-209237, National Aeronautics and Space Administration NASA (1999).

[3] J. Anderson, Fundamentals of Aerodynamics, McGraw-Hill Education, 2010.

[4] I. Abbott, A. Von Doenhoff, Theory of Wing Sections, Including a Summary of Airfoil Data, Dover Books on Aeronautical Engineering Series, Dover Publications, 1959.

[5] A. Kermode, R. Barnard, D. Philpott, Mechanics of Flight, Longman, 1996.

[6] W. Wright, O. Wright, O. Chanute, M. McFarland, The Papers of Wilbur and Orville Wright: 1906-1948, The Papers of Wilbur and Orville Wright, McGraw-Hill, 1953.

[7] B. L. Peandtl, Applications of Modern Hydrodynamics to Aeronautics, Tech. Rep. NACA-TR-116, National Aeronautics and Space Administration NASA (1923).

[8] Norton, F. H, Construction of Models for Tests in Wind Tunnels, Tech. Rep. NACA-TR-74, National Aeronautics and Space Administration NASA (1920).

[9] Margoulis, W, A New Method of Testing in Wind Tunnels, Tech. Rep. NACA-TN-52, National Aeronautics and Space Administration NASA (1921).

[10] F. W. Caldwell, E. N. Fales, Wind Tunnel Studies in Aerodynamic Phenomena at High Speed, Tech. Rep. NACA-TR-83, National Aeronautics and Space Administration NASA (1921).

[11] B. Allen, Eastman N. 'Jake' Jacobs — NASA, visited on 2020-05-16.
URL `https://www.nasa.gov/langley/hall-of-honor/eastman-n-jacobs`

[12] L. W. Habel, J. H. Henderson, M. F. Miller, The Langley Annular Transonic Tunnel, Tech. Rep. NACA-TR-1106, National Aeronautics and Space Administration NASA (1952).

[13] W. C. Moseley, R. T. Taylor, Low-Speed Chordwise Pressure Distribution Near The Midspan Station Of The Slotted Flap And Aileron Of A 1/4-Scale Model Of The Bell X-l Airplane With A 4-Percent-Thick , Aspect-Ratio-4, Unswept Wing, Tech. Rep. NACA-RM-L53L18, National Aeronautics and Space Administration NASA (1954).

[14] F. T. Johnson, E. N. Tinoco, N. J. Yu, Thirty Years of Development and Application of CFD at Boeing Commercial Airplanes, Seattle, Computers and Fluids 34 (2005) 1115–1151. `doi:10.1016/j.compfluid.2004.06.005`.

[15] M. W. Bowman, Boeing 747: A History: Delivering the Dream, Pen and Sword Aviation, 2014.

[16] P. Rubbert, E. Tinoco, The Impact of Computational Aerodynamics on Aircraft Design, American Institute of Aeronautics and Astronautics (AIAA), 1983. `doi:10.2514/6.1983-2060`.

[17] R. L. Bengelink, P. E. Rubbert, The Impact of CFD on the Airplane Design Process: Today and Tomorrow, SAE Transactions 100 (1991) 2059–2068. doi:10.4271/911989.

[18] J. Reuther, S. Cliff, R. Hicks, C. Van Dam, Practical Design Optimization of Wing/-Body Configurations Using the Euler Equations, American Institute of Aeronautics and Astronautics (AIAA), 1992. doi:10.2514/6.1992-2633.

[19] T. R. Barrett, N. W. Bressloff, A. J. Keane, Airfoil Shape Design and Optimization Using Multifidelity Analysis and Embedded Inverse Design, AIAA Journal 44 (9) (2006) 2051–2060. doi:10.2514/1.18766.

[20] R. M. Hicks, E. M. Murman, An Assessment of Airfoil Design by Numerical Optimization, Tech. Rep. NASA-TM-X-3092, National Aeronautics and Space Administration NASA (1974).

[21] R. M. Hicks, P. A. Henne, Wing Design By Numerical Optimization, Journal of Aircraft 15 (7) (1978) 407–412. doi:10.2514/3.58379.

[22] J. Lions, Optimal Control of Systems Governed by Partial Differential Equations, Springer-Verlag, 1971.

[23] A. J. Keane, P. B. Nair, Computational Approaches for Aerospace Design: The Pursuit of Excellence, Wiley Publisher, 2005.

[24] S. Skinner, H. Zare-Behtash, State-Of-The-Art in Aerodynamic Shape Optimisation Methods, Applied Soft Computing 62 (2018) 933–962. doi:10.1016/j.asoc.2017.09.030.

[25] J. F. Kennedy, R. C. Eberhart, Y. Shi, Swarm intelligence, Morgan Kaufmann Publishers, 2001.

[26] S. N. Sivanandam, S. N. Deepa, Introduction to Genetic Algorithms, Springer, 2007.

[27] M. A. Bouhlel, J. T. Hwang, N. Bartoli, R. Lafage, J. Morlier, J. R. Martins, A Python Surrogate Modeling Framework with Derivatives, Advances in Engineering Software 135 (2019) 102662. doi:10.1016/j.advengsoft.2019.03.005.

[28] W. Chen, Z. J. Fu, C. S. Chen, Radial Basis Functions, in: SpringerBriefs in Applied Sciences and Technology, Springer Verlag, 2014, pp. 5–28. doi:10.1007/978-3-642-39572-7_2.

[29] J. Sacks, S. B. Schiller, W. J. Welch, Designs for Computer Experiments, Technometrics 31 (1) (1989) 41–47. doi:10.1080/00401706.1989.10488474.

[30] J. Hastie, Trevor, Tibshirani, Robert, Friedman, The Elements of Statistical Learning, Data Mining, Inference, and Prediction, 2009. doi:10.1007/978-0-387-84858-7.

[31] C. C. Aggarwal, Neural Networks and Deep Learning, Springer. doi:10.1007/978-3-319-94463-0.

[32] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, Backpropagation Applied to Handwritten Zip Code Recognition, Neural Computation 1 (4) (1989) 541–551. doi:10.1162/neco.1989.1.4.541.

[33] B. M. Kulfan, CST Universal Parametric Geometry Representation Method With Applications to Supersonic Aircraft Brenda, Tech. rep., Fourth International Conference on Flow Dynamics (2007).

[34] UIUC Airfoil Data Site, visited on 2020-05-16.
URL https://m-selig.ae.illinois.edu/ads/coord_database.html

[35] A. Jameson, Aerodynamic Design via Control Theory, Journal of Scientific Computing 3 (3) (1988) 233–260. doi:10.1007/BF01061285.

[36] A. Jameson, Optimum Aerodynamic Design Using CFD and Control Theory, in: 12th Computational Fluid Dynamics Conference, American Institute of Aero-

nautics and Astronautics Inc, AIAA, 1995, pp. 926–949. doi:10.2514/6.1995-1729.

[37] A. Jameson, L. Martinelli, N. Pierce, Optimum Aerodynamic Design Using the Navier-Stokes Equations, Theoretical and Computational Fluid Dynamics 10 (1-4) (1998) 213–237. doi:10.1007/s001620050060.

[38] S. K. Nadarajah, A. Jameson, A Comparison of the Continuous and Discrete Adjoint Approach to Automatic Aerodynamic Optimization, in: 38th Aerospace Sciences Meeting and Exhibit, 2000. doi:10.2514/6.2000-667.

[39] A. Dadone, B. Grossman, Progressive Optimization of Inverse Fluid Dynamic Design Problems, Computers and Fluids 29 (1) (2000) 1–32. doi:10.1016/S0045-7930(99)00002-X.

[40] A. Jameson, Aerodynamic shape optimization using the adjoint method, in: VKI Lecture Series on Aerodynamic Drag Prediction and Reduction, von Karman Institute of Fluid Dynamics, Rhode St Genese, 2003, pp. 3–7.

[41] D. J. Mavriplis, A Discrete Adjoint-Based Approach for Optimization Problems on Three-Dimensional Unstructured Meshes, in: Collection of Technical Papers - 44th AIAA Aerospace Sciences Meeting, Vol. 1, 2006, pp. 624–643. doi:10.2514/1.22743.

[42] G. Kuruvila, S. Ta'asan, M. D. Salas, Airfoil Design and Optimization by the One-Shot Method, Tech. Rep. AIAA 95-0478, American Institute of Aeronautics and Astronautics AIAA (1994).

[43] W. K. Anderson, V. Venkatakrishnan, Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation, Computers and Fluids 28 (4-5) (1999) 443–480. doi:10.1016/S0045-7930(98)00041-3.

[44] A. Baeza, C. Castro, F. Palacios, E. Zuazua, 2D Euler Shape Design on Non-Regular Flows Using Adjoint Rankine-Hugoniot Relations, in: 46th AIAA Aerospace Sciences Meeting and Exhibit, American Institute of Aeronautics and Astronautics, Reston, Virigina, 2008. doi:10.2514/6.2008-171.

[45] M. D. Salas, The Curious Events Leading to the Theory of Shock Waves, Tech. Rep. 20060047586, National Aeronautics and Space Administration NASA (2006).

[46] A. Jameson, S. Shankaran, L. Martinelli, Continuous Adjoint Method for Unstructured Grids, in: AIAA Journal, Vol. 46, 2008, pp. 1226–1239. doi:10.2514/1.25362.

[47] J. E. Hicken, D. W. Zingg, Aerodynamic Optimization Algorithm with Integrated Geometry Parameterization and Mesh Movement, AIAA Journal 48 (2) (2010) 400–413. doi:10.2514/1.44033.

[48] R. Hu, A. Jameson, Q. Wang, Adjoint-Based Aerodynamic Optimization of Supersonic Biplane Airfoils, in: Journal of Aircraft, Vol. 49, American Institute of Aeronautics and Astronautics Inc., 2012, pp. 802–814. doi:10.2514/1.C031417.

[49] D. Maruyama, K. Kusunose, K. Matsushima, K. Nakahashi, Aerodynamic Analysis and Design of Busemann Biplane: Towards Efficient Supersonic Flight, Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering 226 (2) (2012) 217–238. doi:10.1177/0954410011404656.

[50] J. Wang, Y. Zheng, J. Chen, F. Xie, J. Zhang, J. Périaux, Z. Tang, Engineering Optimization Single/Two-Objective Aerodynamic Shape Optimization by a Stackelberg/Adjoint Method, Engineering Optimization (5) (2019) 753–776. doi:10.1080/0305215X.2019.1618287.

[51] H. von Stackelberg, S. Von, A. Peacock, The Theory of the Market Economy, Oxford University Press, 1952.

[52] Y. Zhu, Y. Ju, C. Zhang, An Experience-Independent Inverse Design Optimization Method of Compressor Cascade Airfoil, Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy 233 (4) (2019) 431–442. doi:10.1177/0957650918790998.

[53] S. H. Darwish, M. M. Abdelrahman, A. M. Nagib Elmekawy, K. Elsayed, Aerodynamic Shape Optimization of Helicopter Rotor Blades in Hover using Genetic Algorithm and Adjoint Method, AIAA SciTech Forum (004) (2018) 8–12. doi:10.2514/6.2018-0044.

[54] F. Gagliardi, K. C. Giannakoglou, RBF-Based Morphing of B-Rep Models for Use in Aerodynamic Shape Optimization, Advances in Engineering Software 138. doi:10.1016/j.advengsoft.2019.102724.

[55] H. Qin, Dynamic Non-Uniform Rational B-Splines, Ph.D. thesis, Departament of Computer Science, University of Toronto (1995).

[56] A. Hopgood, Intelligent Systems For Engineers And Scientists., CRC PRESS, 2018.

[57] K. C. Giannakoglou, Design of Optimal Aerodynamic Shapes Using Stochastic Optimization Methods and Computational intelligence, Progress in Aerospace Sciences 38 (1) (2002) 43–76. doi:10.1016/S0376-0421(01)00019-7.

[58] M. Birattari, Tuning Metaheuristics, Vol. 197, Springer, 2009.

[59] A. Shahrokhi, A. Jahangirian, Airfoil Shape Parameterization for Optimum Navier-Stokes Design with Genetic Algorithm, Aerospace Science and Technology 11 (2007) 443–450. doi:10.1016/j.ast.2007.04.004.

[60] A. Ribeiro, A. Awruch, H. M. Gomes, An Airfoil Optimization Technique for Wind Turbines, Applied Mathematical Modelling 36 (10) (2012) 4898–4907. doi:10.1016/j.apm.2011.12.026.

[61] A. Jahangirian, A. Shahrokhi, Aerodynamic Shape Optimization Using Efficient Evolutionary Algorithms and Unstructured CFD Solver, Computers and Fluids 46 (2011) 270–276. doi:10.1016/j.compfluid.2011.02.010.

[62] R. Mukesh, K. Lingadurai, U. Selvakumar, Airfoil Shape Optimization Using Non-Traditional Optimization Technique and its Validation, Journal of King Saud University - Engineering Sciences 26 (2014) 191–197. doi:10.1016/j.jksues.2013.04.003.

[63] B. L. Miller, B. L. Miller, D. E. Goldberg, D. E. Goldberg, Genetic algorithms, tournament selection, and the effects of noise, Complex Systems 9 (1995) 193–212.

[64] E. McGookin, Optimisation of Sliding Mode Controllers for Marine Applications: A Study of Methods and Implementation Issues, Ph.D. thesis, Department of Electronics and Electrical Engineering, University of Glasgow (1997).

[65] M. Mitchell, Genetic Algorithms: An Overview, Complexity 1 (1) (1995) 31–39. doi:10.1002/cplx.6130010108.

[66] K. G. Khoo, P. N. Suganthan, Evaluation of Genetic Operators and Solution Representations for Shape Recognition by Genetic Algorithms, Pattern Recognition Letters 23 (13) (2002) 1589–1597. doi:10.1016/S0167-8655(02)00123-X.

[67] K. A. De Jong, W. M. Spears, A Formal Analysis of the Role of Multi-Point Crossover in Genetic Algorithms, Annals of Mathematics and Artificial Intelligence 5 (1) (1992) 1–26. doi:10.1007/BF01530777.

[68] E. McGookin, A Population Minimisation Process for Genetic Algorithms and its Application to Controller Optimisation, Institution of Engineering and Technology (IET), 2005, pp. 79–84. doi:10.1049/cp:19971159.

[69] K. Chiba, S. Obayashi, K. Nakahashi, H. Morino, High-Fidelity Multidisciplinary Design Optimization of Wing Shape for Regional Jet Aircraft, in: Springer (Ed.), Evolutionary Multi-Criterion Optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 621–635.

[70] R. Hassan, B. Cohanim, O. De Weck, G. Venter, A Comparison Of Particle Swarm Optimization And The Genetic Algorithm, AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 46. doi:10.2514/6.2005-1897.

[71] J. H. Kim, B. Ovgor, K. H. Cha, J. H. Kim, S. Lee, K. Y. Kim, Optimization of the Aerodynamic and Aeroacoustic Performance of an Axial-Flow Fan, AIAA Journal 52 (9) (2014) 2032–2044. doi:10.2514/1.J052754.

[72] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197. doi:10.1109/4235.996017.

[73] Z. Lyu, Z. Xu, J. Martins, Benchmarking Optimization Algorithms for Wing Aerodynamic Design Optimization, in: The Eighth International Conference on Computational Fluid Dynamics (ICCFD8), 2014.

[74] S. N. Skinner, H. Zare-Betash, Aerodynamic Design Optimisation of Non-planar Lifting Surfaces, in: 57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2015.

[75] T. Kato, K. Shimoyama, S. Obayashi, Evolutionary Algorithm with Parallel Evaluation Strategy of Feasible and Infeasible Solutions Considering Total Constraint Violation, in: 2015 IEEE Congress on Evolutionary Computation (CEC), 2015, pp. 986–993.

[76] D. J. Zhao, Y. K. Wang, W. W. Cao, P. Zhou, Optimization of Suction Control on an Airfoil Using Multi-island Genetic Algorithm, in: Procedia Engineering, Vol. 99, Elsevier Ltd, 2015, pp. 696–702. doi:10.1016/j.proeng.2014.12.591.

[77] J. Zhang, L. Xu, R. Gao, Multi-island genetic algorithm opetimization of suspension system, Indonesian Journal of Electrical Engineering and Computer Science 10 (2012) 1685–1691.

[78] D. Sasaki, M. Morikawa, S. Obayashi, K. Nakahashi, Aerodynamic Shape Optimization of Supersonic Wings by Adaptive Range Multiobjective Genetic Algorithms, in: Lecture Notes in Computer Science, Springer Verlag, 2001, pp. 639–652. doi:10.1007/3-540-44719-9_45.

[79] D. Sasaki, S. Obayashi, H. J. Kim, Evolutionary Algorithm vs. Adjoint Method Applied to SST Shape Optimization, Tech. rep., Korea Aerospace Research Institute.

[80] P. W. Jansen, R. E. Perez, J. R. R. A. Martins, Aerostructural Optimization of Nonplanar Lifting Surfaces, Journal of Aircraft 47 (5) (2010) 1490–1503. doi:10.2514/1.44727.

[81] M. Masdari, M. Tahani, M. H. Naderi, N. Babayan, Optimization of Airfoil Based Savonius Wind Turbine Using Coupled Discrete Vortex Method and Salp Swarm Algorithm, Journal of Cleaner Production 222 (2019) 47–56. doi:10.1016/j.jclepro.2019.02.237.

[82] C. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[83] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.

[84] K. Jarrett, K. Kavukcuoglu, M. Ranzato, Y. LeCun, What Is the Best Multi-Stage Architecture for Object Recognition?, in: 2009 IEEE 12th International Confer-

ence on Computer Vision, 2009, pp. 2146–2153. doi:10.1109/ICCV.2009.5459469.

[85] X. Glorot, A. Bordes, Y. Bengio, Deep Sparse Rectifier Neural Networks, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Vol. 15, PMLR, 2011, pp. 315–323.

[86] M. Bellman, J. Straccia, B. Morgan, K. Maschmeyer, R. Agarwal, Improving Genetic Algorithm Efficiency with an Artificial Neural Network for Optimization of Low Reynolds Number Airfoils, in: 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition, Orlando, Florida, USA, 2009. doi:10.2514/6.2008-171.

[87] M. Kotinis, A. Kulkarni, Multi-Objective Shape Optimization of Transonic Airfoil Sections Using Swarm Intelligence and Surrogate Models, Structural and Multidisciplinary Optimization 45 (5) (2012) 747–758. doi:10.1007/s00158-011-0719-7.

[88] Y.-Y. Wang, B.-Q. Zhang, Y.-C. Chen, Robust Airfoil Optimization Based on Improved Particle Swarm Optimization Method, Appl. Math. Mech.-Engl. Ed 32 (10) (2011) 1245–1254. doi:10.1007/s10483-011-1497-x.

[89] M. Khurana, H. Winarto, A. Sinha, Application of Swarm Approach and Artificial Neural Networks for Airfoil Shape Optimization, in: 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, American Institute of Aeronautics and Astronautics, Reston, Virigina, 2008. doi:10.2514/6.2008-5954.

[90] T. Kohonen, Self-Organizing Maps, Springer, 2001.

[91] S. Pierret, R. A. Van den Braembussche, Turbomachinery Blade Design Using a NavierStokes Solver and Artificial Neural Network, Journal of Turbomachinery 121 (2) (1999) 326–332. doi:10.1115/1.2841318.

[92] G. E. Farin, Curves and Surfaces for Computer Aided Geometric Design : A Practical Guide, Academic Press, 1993.

[93] G. Sun, Y. Sun, S. Wang, Artificial Neural Network Based Inverse Design: Airfoils and Wings, Aerospace Science and Technology 42 (2015) 415–428. doi:10.1016/j.ast.2015.01.030.

[94] H. Sobieczky, Parametric Airfoils and Wings, in: Recent Development of Aerodynamic Design Methodologies. Notes on Numerical Fluid Mechanics (NNFM), Springer Vieweg Verlag, 1999. doi:10.1007/978-3-322-89952-1_4.

[95] A. Kharal, A. Saleem, Neural Networks Based Airfoil Generation for a Given Cp Using Bezier-PARSEC Parameterization, Aerospace Science and Technology 23 (2012) 330–344. doi:10.1016/j.ast.2011.08.010.

[96] R. W. Derksen, T. Rogalsky, Bezier-PARSEC: An Optimized Aerofoil Parameterization for Design, Advances in Engineering Software 45 (2010) 923–930. doi:10.1016/j.advengsoft.2010.05.002.

[97] V. Sekar, M. Zhang, C. Shu, B. C. Khoo, Inverse Design of Airfoil Using a Deep Convolutional Neural Network, AIAA Journal 57. doi:10.2514/1.J057894.

[98] E. Yilmaz, B. J. German, A Deep Learning Approach to an Airfoil Inverse Design Problem, in: Multidisciplinary Analysis and Optimization Conference, American Institute of Aeronautics and Astronautics, Atlanta, Georgia, USA, 2018. doi:10.2514/6.2018-3420.

[99] V. Braibant, C. Fleury, Shape optimal design using B-splines, Computer Methods in Applied Mechanics and Engineering 44 (3) (1984) 247–267. doi:10.1016/0045-7825(84)90132-4.

[100] V. Sripawadkul, M. Padulo, M. Guenov, A Comparison of Airfoil Shape Parameterization Techniques for Early Design Optimization, in: 13th AIAA/ISSMO Mul-

tidisciplinary Analysis Optimization Conference, Fort Worth, Texas, USA, 2010. doi:10.2514/6.2010-9050.

[101] A. Sóbester, Exploiting Patterns in the Kulfan Transformations of Supercritical Airfoils, in: 9th AIAA Aviation Technology, Integration and Operations (ATIO) Conference, Aircraft Noise and Emissions Reduction Symposium (ANERS), Hilton Head, South Carolina, USA, 2009. doi:10.2514/6.2009-6951.

[102] B. M. Kulfan, J. E. Bussoletti, "Fundamental" Parametric Geometry Representations for Aircraft Component Shapes, in: 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, Virginia, USA, 2006. doi:10.2514/6.2006-6948.

[103] A. J. Emma, C. A. Syvertson, S. Kiwjs, A Study Of Inviscid Flow About Airfoils At High Supersonic Speeds, Tech. Rep. NACA-TR-1123, National Aeronautics and Space Administration NASA (1953).

[104] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, J. Alonso, SU2: An Open-Source Suite for Multiphysics Simulation and Design, AIAA JOURNAL 54 (3). doi:10.2514/1.J053813.

[105] F. Palacios, T. D. Economon, S. R. Copeland, T. W. Lukaczyk, J. J. Alonso, Stanford University Unstructured (SU2): Analysis and Design Technology for Turbulent Flows, in: 52nd Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, National Harbor, Maryland, USA, 2018. doi:10.2514/6.2014-0243.

[106] G. R. Anderson, M. Nemec, M. J. Aftosmis, Aerodynamic Shape Optimization Benchmarks with Error Control and Automatic Parameterization, in: 53rd AIAA Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, Kissimmee, Florida, USA, 2015. doi:10.2514/6.2015-1719.

[107] J. B. Mcdevkt, A. F. Okuno, Static and Dynamic Pressure Measurements on a NACA 0012 Airfoil in the Ames High Reynolds Number Facility, Tech. Rep. NASA-TP-2485, National Aeronautics and Space Administration NASA (1985).

[108] J. Anderson, Computational Fluid Dynamics: The Basics with Applications, McGraw-Hill International Editions: Mechanical Engineering, McGraw-Hill, 1995.

[109] A. Jameson, T. Baker, Solution of the Euler Equations for Complex Configurations, in: 6th Computational Fluid Dynamics Conference Danvers, American Institute of Aeronautics and Astronautics, Reston, Virigina, 1983. doi:10.2514/6.1983-1929.

[110] A. Jameson, Solution of the Euler Equations for Two-Dimensional Transonic Flow by a Multigrid Method, Applied Mathematics and Computation 13 (3-4) (1983) 327–355. doi:10.1016/0096-3003(83)90019-X.

[111] T. K. Sengupta, A. Bhole, N. A. Sreejith, Direct Numerical Simulation of 2D Transonic Flows Around Airfoils, Computers and Fluids 88 (2013) 19–37. doi:10.1016/j.compfluid.2013.08.007.

[112] J. Siegler, J. Ren, L. Leifsson, S. Koziel, A. Bekasiewicz, Supersonic Airfoil Shape Optimization by Variable-Fidelity Models and Manifold Mapping, Procedia Computer Science 80 (2016) 1103–1113. doi:10.1016/j.procs.2016.05.416.

[113] S.-i. Amari, Backpropagation and Stochastic Gradient Descent Method, Neurocomputing 5 (4-5) (1993) 185–196. doi:10.1016/0925-2312(93)90006-O.

[114] D. P. Kingma, J. L. Ba, Adam: A method for stochastic optimization, in: 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, International Conference on Learning Representations, ICLR, 2015. arXiv:1412.6980.

[115] M. D. Zeiler, ADADELTA: An Adaptive Learning Rate Method, Tech. Rep. 1212.5701, Google Inc. and New York University, USA (2012). arXiv:1212.5701.

[116] J. Duchi, E. Hazan, Y. Singer, Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, Tech. Rep. UCB/EECS-2010-24, EECS Department, University of California, Berkeley (Mar 2010).

[117] G. Montavon, G. B.Orr, K.-R. Mller, Neural Networks: Tricks of the Trade, Springer, 2012.

Appendix A

Python scripts

A.1 Create Directories

```
import numpy
import subprocess

i = numpy.arange(1,396,1)
j = numpy.arange(1,5,1)
k = numpy.arange(1,7,1)

for x in i:
        _testName = '/mnt/c/Users/s293169/Documents/SU2/SU2
            /Data_base/%d' %x
        subprocess.call(['mkdir',_testName])
        for y in j:
                _testName2 = _testName + '/%d' %y
                subprocess.call(['mkdir',_testName2])
                for s in k:
                        _testName3 = _testName2 + '/%d' %s
```

```
                    subprocess.call(['mkdir',_testName3
                        ])
```

A.2 Run SU2 in directories

```
import numpy
import subprocess
import os
import shutil
wd = os.getcwd()

i = numpy.arange(1,396,1)
j = numpy.arange(1,5,1)
k = numpy.arange(1,7,1)

for x in i:
        _testName = '/mnt/c/Users/s293169/Documents/SU2/SU2
            /Data_base/%d' %x
        for y in j:
                _testName2 = _testName + '/%d' %y
                for s in k:
                        _testName3 = _testName2 + '/%d' %s
                        namefile = '%d' %x + '_' + '%d' %y
                            + '_' + '%d' %s + '.cfg'
                        vtk = '%d' %x + '_' + '%d' %y + '_'
                             + '%d' %s + '.vtk'
                        shutil.copy2(wd + '/SU2_CFD.exe',
```

```
        _testName3)
    os.chdir(_testName3)
    subprocess.call([_testName3 + '/
        SU2_CFD.exe', '%s' %namefile])
    subprocess.call(['rm','-r','%s' %
        vtk ,'history.vtk','restart_flow
        .dat','surface_flow.vtk','
        SU2_CFD.exe' ])
    os.chdir(wd)
```

A.3 Read lift and drag coefficients from CFD simulations

```
import sys
import pandas as pd
import numpy
import subprocess
import os

wd = os.getcwd()
pd.set_option('display.max_rows', None)
datafile2 = open("coefficient_complete.txt","w+")
i = numpy.arange(1,396,1)
j = numpy.arange(1,5,1)
k = numpy.arange(1,7,1)


for x in i:
```

```
_testName = '/mnt/c/Users/s293169/Documents/SU2/SU2
    /Data_base/%d' %x
for y in j:
        _testName2 = _testName + '/%d' %y
        for s in k:
                _testName3 = _testName2 + '/%d' %s
                os.chdir(_testName3)
                data = []
                with open('forces_breakdown.dat', '
                    r') as f:
                        d = f.readlines()
                        for z in d:
                                c = z.rstrip().
                                    split(",")
                                data.append(c)
                data = numpy.array(data, dtype='O')
                data = list(data)
                data = str(data)
                data = data.split()
                data = str(data)
                a = numpy.arange(7,15,1)
                CL2 = []
                CD2 = []
                for w in a:

                        CL1 = data[data.index("CL:"
                            ) + w]
                        CD1 = data[data.index("CD:"
```

```
                ) + w]
            CL2.append(CL1)
            CD2.append(CD1)

        CL2 = ''.join(CL2)
        CD2 = ''.join(CD2)
        datafile2.write(CL2 + '␣' + CD2 + '
            \n')

os.chdir(wd)
```

[language=Python, breaklines=true]

A.4 Separate lift and drag coefficients by Mach number

```
import sys
import pandas as pd
import numpy
import subprocess
import os

wd = os.getcwd()
pd.set_option('display.max_rows', None)
datafile2 = open("pressure_mach3.txt","w+")
i = numpy.arange(1,9480,24)
print(i)
for x in i:
        j = x + 83
```

```
        h = x + 62
        k = numpy.arange(h,j,1)
        print(k)
        os.chdir('/mnt/c/Users/s293169/Documents/SU2/SU2/Data_base')
        f = numpy.loadtxt('coefficient_complete.txt',delimiter=None)
        data = []
        for z in f:
                data.append(z)
        b = []
        for idx in k:
                #print(data[idx])
                b = data[idx]
                datafile2.write(str(b).strip("[]") + '\n')
```

A.5 Adjoin aerodynamic coefficients and CST coefficients

```
import sys
import pandas as pd
import numpy
import os

wd = os.getcwd()
pd.set_option('display.max_rows', None)
alpha = numpy.arange(-10,11,1)
datafile = open("features2_mach_2.txt","w+")
#datafile2 = open("features2.txt","w+")
```

```
i = numpy.arange(0,7498,21)
filename = 0
for x in i:


        filename += 1
        j = x + 21
        k = numpy.arange(x,j,1)
        os.chdir('/mnt/c/Users/s293169/Documents/SU2/SU2/
            Data_base')
        aero = numpy.loadtxt('pressure_mach2.txt',delimiter
           =None)
        os.chdir('/mnt/c/Users/s293169/Documents/SU2/SU2/
            Data_base/weights')
        for idx in k:
                weights = numpy.loadtxt('%d' %filename + '.
                    txt')
                parm = list(weights)
                parm_aero = aero[idx]
                datafile.write(str(parm_aero).strip("[]") +
                     (" ") + str(parm).strip("[]").strip(","
                    ) + '\n')
```

Appendix B

MATLAB scripts

B.1 Main Parameterisation

```
clear all; clc;

%% Importing data
imp = importdata('237.txt');      %coordinates text file
    here
m = size(imp,1);
y_up = imp(1:(m/2),2)';
y_lo = imp((m/2)+1:end,2)';
mx = (m/2)-1;
x_up = imp(1:(m/2),1)';
x_lo = imp((m/2)+1:end,1)';

%% Defining the parameters of the airfoil

%CST initial guess
Par_airfoil = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1,
```

```
    0.1, -0.1, -0.1, -0.1, -0.1, -0.1, -0.1, 0.1, 0.1, 0.5,
    1.0];

%% Calling function

[Par_airfoil_opt] = ...
    fmincon(@(t)(CST_fitting(t, y_up, y_lo, x_up, x_lo)),
        Par_airfoil, [], [], [], [], ones(1,20)*-1, ones
        (1,20), []);

z = CST_fitting(Par_airfoil_opt, y_up, y_lo, x_up, x_lo);
[Z1, Z2] = CST(Par_airfoil_opt, x_up, x_lo);

%% Parsec

%PARSEC initial guess
Par_airfoil_parsec = [0.1, -0.1, 0.1, 0.1, 0.1, 0.1, 0.1,
    0.1, 0.1, 0.1, 0.1, 0.1];

[Par_airfoil_opt_parsec] = ...
    fmincon(@(t)(PARSECfitting(t, y_up, y_lo, x_up, x_lo, m)
        ), Par_airfoil_parsec, [], [], [], [], ones(1,12)*-1,
         ones(1,12), []);

z2 = PARSECfitting(Par_airfoil_opt_parsec, y_up, y_lo, x_up
    , x_lo, m);
[Z3, Z4] = PARSEC(Par_airfoil_opt_parsec, x_up, x_lo, m);

```

```
36 %% Data storage (THIS SECTION IS TO STORE NEW CST
       COEFFICIENTS)
37
38 Par_airfoil_opt = Par_airfoil_opt';
39 x_up = x_up';
40 r = [Par_airfoil_opt;(mx + 1); z; x_up;];
41 fileID = fopen('CST_weights_whitcomb.txt','w');      %write
       the CST parameters in the specified text file
42 fprintf(fileID, '%0.30f\n', r);
43 fclose(fileID);
```

B.2 CST_fitting

```
1
2 function [error] = CST_fitting(Par_airfoil, y_up, y_lo,
      x_up, x_lo, m)
3
4 [Z1, Z2] = CST(Par_airfoil, x_up, x_lo);
5 Z = [Z1, Z2];
6 y = [y_up, y_lo];
7 error = mean(abs(y - Z));
8 end
```

B.3 PARSECfitting

```
1 function [error] = PARSECfitting(Par_airfoil, y_up, y_lo,
      x_up, x_lo, m)
2
3 [Z1, Z2] = PARSEC(Par_airfoil, x_up, x_lo, m);
```

```
4 Z = [Z1, Z2];
5 y = [y_up, y_lo];
6 error = mean(abs(y - Z));
7 end
```

B.4 CST scheme

```
1  function [z_up,z_lo] = CST(Par_airfoil, x_up, x_lo)
2
3
4  R_le_up = Par_airfoil(1,1);      %Leading edge radius upper
5  R_le_lo = Par_airfoil(1,2);      %Leading edge radius lower
6  dZe_up = (Par_airfoil(1,3));     %Trailing edge thickness
7  dZe_lo = (Par_airfoil(1,4));
8  Par_up_1 = Par_airfoil(1,5);
9  Par_up_2 = Par_airfoil(1,6);
10 Par_up_3 = Par_airfoil(1,7);
11 Par_up_4 = Par_airfoil(1,8);
12 Par_up_5 = Par_airfoil(1,9);
13 Par_up_6 = Par_airfoil(1,10);
14 Par_lo_1 = Par_airfoil(1,11);
15 Par_lo_2 = Par_airfoil(1,12);
16 Par_lo_3 = Par_airfoil(1,13);
17 Par_lo_4 = Par_airfoil(1,14);
18 Par_lo_5 = Par_airfoil(1,15);
19 Par_lo_6 = Par_airfoil(1,16);
20 theta_up = Par_airfoil(1,17);  %Trailing edge Boattail
       angle Upper
```

```
21 theta_lo = Par_airfoil(1,18); %Trailing edge Boattail
       angle Lower
22
23
24 %% Berstein polynomials
25 n = 7;         %Berstein polynomial order
26 a = 0:1:n;
27 m = size(x_up,2);
28 i = size(a,2);
29 K = zeros(i,1);
30 K(1,:) = 1;
31 K(2:end,:) = factorial(n)./(factorial(a(2:end)).*factorial
       ((n - a(2:end))));
32 b = zeros(i,m);
33 for d = 2:1:n+1
34     b(1,:) = 1;
35     b(d,:) = x_up.^(d-1);
36 end
37
38 b_2 = zeros(i,m);
39 for d_2 = 2:1:n
40     b_2(1,:) = (1 - x_up).^(n);
41     b_2(d_2,:) = (1 - x_up).^(n - (d_2 - 1));
42     b_2(n+1,:) = 1;
43 end
44
45 Sk = b_2.*b.*K;
46 %% Parameters Upper & Lower surfaces Vectorisation
```

```
47
48
49 B_up = [sqrt(2*R_le_up); Par_up_1; Par_up_2; Par_up_3;
       Par_up_4; Par_up_5; Par_up_6; (theta_up) + dZe_up]; %
       Parameters Upper vector
50
51 B_lo = [-sqrt(2*R_le_lo); Par_lo_1; Par_lo_2; Par_lo_3;
       Par_lo_4; Par_lo_5; Par_lo_6; (theta_lo) + dZe_lo]; %
       Parameters Lower vector
52
53 %% CST Class Function
54
55 N1 =  Par_airfoil(1,19);
56 N2 =  Par_airfoil(1,20);
57 C = (x_up.^(N1)).*((1 - x_up).^(N2)); % Class function
58
59 %% Polynomial weights calculation
60
61
62 S_up =Sk.*B_up;
63 S_up_1 = S_up(1,:) + S_up(2,:) + S_up(3,:) + S_up(4,:) +
       S_up(5,:) + S_up(6,:) + S_up(7,:) + S_up(8,:);
64
65 z_up = C.*S_up_1;
66
67 S_lo = Sk.*B_lo;
68 S_lo_1 = S_lo(1,:) + S_lo(2,:) + S_lo(3,:) + S_lo(4,:) +
       S_lo(5,:) + S_lo(6,:) + S_lo(7,:) + S_lo(8,:);
```

```
69
70 z_lo = C.*S_lo_1;
71 end
```

B.5 PARSEC scheme

```
1  function [z_up, z_lo] = PARSEC(Par_airfoil, x_up, x_lo, m);
2  %PARSEC AIRFOIL GENERATOR
3
4  R_le_up = Par_airfoil(1,1);    %Leading edge radius upper
5  R_le_lo = Par_airfoil(1,2);    %Leading edge radius lower
6  alpha = Par_airfoil(1,3);      %Trailindg edge angle
7  beta = Par_airfoil(1,4);       %Trailing edge wedge angle
8  dZe = Par_airfoil(1,5);        %Trailing edge thickness
9  Ze = Par_airfoil(1,6);         %Trailing edge Z position
10 X_up = abs(Par_airfoil(1,7));   %Upper crest X position
11 Z_up = abs(Par_airfoil(1,8));   %Upper crest Z position
12 Z_xxup = Par_airfoil(1,9);      %Upper crest curvature
13 X_lo = abs(Par_airfoil(1,10)); %Lower crest X position
14 Z_lo = Par_airfoil(1,11);       %Lower crest Z position
15 Z_xxlo = Par_airfoil(1,12);     %Lower crest Z position
16
17
18 %% Parameters_Upper & Lower surfaces_VECTORIZATION
19
20 ZX_te = Ze + dZe/2;
21 ZX_te2 = Ze - dZe/2;
22
```

```
23 A_up = [[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]; ...
24       [X_up^(1/2), X_up^(3/2), X_up^(5/2), X_up^(7/2), X_up
             ^(9/2), X_up^(11/2)]; ...
25       [1/2, 3/2, 5/2, 7/2, 9/2, 11/2]; ...
26       [(1/2)*(X_up^(-1/2)), (3/2)*(X_up^(1/2)), (5/2)*(X_up
             ^(3/2)), (7/2)*(X_up^(5/2)), (9/2)*(X_up^(7/2)),
             (11/2)*(X_up^(9/2))]; ...
27       [(-1/4)*(X_up^(-3/2)), (3/4)*(X_up^(-1/2)), (15/4)*(
             X_up^(1/2)), (35/4)*(X_up^(3/2)), (53/4)*(X_up^(5/2)
             ), (99/4)*(X_up^(7/2))]; ...
28       [1.0, 0, 0, 0, 0, 0]];
29
30 A_lo = [[1.0, 1.0, 1.0, 1.0, 1.0, 1.0]; ...
31       [X_lo^(1/2), X_lo^(3/2), X_lo^(5/2), X_lo^(7/2), X_lo
             ^(9/2), X_lo^(11/2)]; ...
32       [1/2, 3/2, 5/2, 7/2, 9/2, 11/2]; ...
33       [(1/2)*(X_lo^(-1/2)), (3/2)*(X_lo^(1/2)), (5/2)*(X_lo
             ^(3/2)), (7/2)*(X_lo^(5/2)), (9/2)*(X_lo^(7/2)),
             (11/2)*(X_lo^(9/2))]; ...
34       [(-1/4)*(X_lo^(-3/2)), (3/4)*(X_lo^(-1/2)), (15/4)*(
             X_lo^(1/2)), (35/4)*(X_lo^(3/2)), (53/4)*(X_lo^(5/2)
             ), (99/4)*(X_lo^(7/2))]; ...
35       [1.0, 0, 0, 0, 0, 0]];
36
37 B_up = [ZX_te; Z_up; tand(alpha - (beta/2)); 0; Z_xxup;
       sqrt(2*R_le_up)];
38 B_lo = [ZX_te2; Z_lo; tand(alpha + (beta/2)); 0; Z_xxlo; -
       sqrt(2*R_le_lo)];
```

```
39
40 %% Coefficients calculation
41
42 V_up = linsolve(A_up,B_up);
43 V_lo = linsolve(A_lo,B_lo);
44
45 m = size(x_up,2);
46 n = size(V_up,1);
47 z1 = zeros(n,m);
48 z2 = zeros(n,m);
49 for n = 1:1:6
50
51     z1(n,:) = V_up(n,:)*(x_up.^(n - 0.5));
52     z2(n,:) = V_lo(n,:)*(x_lo.^(n - 0.5));
53
54 end
55
56 z_up = z1(1,:) + z1(2,:) + z1(3,:) + z1(4,:) + z1(5,:) + z1
       (6,:);
57 z_lo = z2(1,:) + z2(2,:) + z2(3,:) + z2(4,:) + z2(5,:) + z2
       (6,:);
58 end
```

B.6 Create Dataset

```
clear all;
clc;
```

```
imp = importdata('features2_mach_0.8_div.txt');
imp2 = importdata('features2_mach_1_div.txt');
imp3 = importdata('features2_mach_2_div.txt');
imp4 = importdata('features2_mach_3_div.txt');

coefficients = imp(:,1:2);
weigths = imp(:,3:22);
C_L_imp = imp(:,1);
C_D_imp = imp(:,2);
C_L_imp2 = imp2(:,1);
C_D_imp2 = imp2(:,2);
C_L_imp3 = imp3(:,1);
C_D_imp3 = imp3(:,2);
C_L_imp4 = imp4(:,1);
C_D_imp4 = imp4(:,2);

a = 1:21:7497;
a = a';
j = 1:1:357;
row = 1;
for i = 1:21:7497'
    C_D_1(row,1:21) = C_D_imp(i:i+20,:)';
    C_L_1(row,1:21) = C_L_imp(i:i+20,:)';
    C_D_2(row,1:21) = C_D_imp2(i:i+20,:)';
    C_L_2(row,1:21) = C_L_imp2(i:i+20,:)';
    C_D_3(row,1:21) = C_D_imp3(i:i+20,:)';
    C_L_3(row,1:21) = C_L_imp3(i:i+20,:)';
    C_D_4(row,1:21) = C_D_imp4(i:i+20,:)';
```

```
    C_L_4(row,1:21) = C_L_imp4(i:i+20,:)';
    weights_2(row,1:20) = weigths(i,:)';
    row = row+1;
end

C_L_1_train1 = C_L_1(:,8);
C_L_2_train1 = C_L_2(:,8);
C_L_3_train1 = C_L_3(:,8);
C_L_4_train1 = C_L_4(:,8);
C_D_1_train1 = C_D_1(:,8);
C_D_2_train1 = C_D_2(:,8);
C_D_3_train1 = C_D_3(:,8);
C_D_4_train1 = C_D_4(:,8);

C_L_1_train2 = C_L_1(:,11);
C_L_2_train2 = C_L_2(:,11);
C_L_3_train2 = C_L_3(:,11);
C_L_4_train2 = C_L_4(:,11);
C_D_1_train2 = C_D_1(:,11);
C_D_2_train2 = C_D_2(:,11);
C_D_3_train2 = C_D_3(:,11);
C_D_4_train2 = C_D_4(:,11);

C_L_1_train3 = C_L_1(:,14);
C_L_2_train3 = C_L_2(:,14);
C_L_3_train3 = C_L_3(:,14);
C_L_4_train3 = C_L_4(:,14);
C_D_1_train3 = C_D_1(:,14);
```

```
C_D_2_train3 = C_D_2(:,14);
C_D_3_train3 = C_D_3(:,14);
C_D_4_train3 = C_D_4(:,14);



C_L_1_train4 = C_L_1(:,17);
C_L_2_train4 = C_L_2(:,17);
C_L_3_train4 = C_L_3(:,17);
C_L_4_train4 = C_L_4(:,17);
C_D_1_train4 = C_D_1(:,17);
C_D_2_train4 = C_D_2(:,17);
C_D_3_train4 = C_D_3(:,17);
C_D_4_train4 = C_D_4(:,17);

Inputs_1 = [C_L_1_train1, C_D_1_train1, C_L_2_train1,
   C_D_2_train1, C_L_3_train1, C_D_3_train1, C_L_4_train1,
   C_D_4_train1];
Inputs_2 = [C_L_1_train2, C_D_1_train2, C_L_2_train2,
   C_D_2_train2, C_L_3_train2, C_D_3_train2, C_L_4_train2,
   C_D_4_train2];
Inputs_3 = [C_L_1_train3, C_D_1_train3, C_L_2_train3,
   C_D_2_train3, C_L_3_train3, C_D_3_train3, C_L_4_train3,
   C_D_4_train3];
Inputs_4 = [C_L_1_train4, C_D_1_train4, C_L_2_train4,
   C_D_2_train4, C_L_3_train4, C_D_3_train4, C_L_4_train4,
   C_D_4_train4];

Dataset = [Inputs_1, Inputs_2, Inputs_3, Inputs_4,
```

```
weights_2];
```

Appendix C

Glyph script

C.1 Mesh Glyph script

```
# AIRFOIL GENERATOR using CST parameterisation
# Written by Aaron Feria

package require PWI_Glyph
#READ DATA FILE X - COORDINATES
set xt 237
set yt 237
#$yt

set filename "C:/Users/s293169/Documents/
   Airfoil_data_for_mesh/Coordinates/$yt.txt"
set f [open $filename "r"]
set file_data [read $f]
close $f
```

```
set data [split $file_data "\n"]
#puts "[lindex $data]"
foreach {one} $data {
  lappend col2 [lindex $one 0]
}

set m [llength $col2]

set len [expr {[llength $col2] / 2}]
set left [lrange $col2 0 [expr {$len - 1}]]
set right [lrange $col2 $len end]

#READ DATA FILE CST PARAMETERS
set filename2 "C:/Users/s293169/Documents/
   Airfoil_data_for_mesh/weights/CST_Coeff_1_deg.txt"
set f2 [open $filename2 "r"]
set file_data2 [read $f2]
close $f2
set m2 [llength $file_data2]

#Parameters_Upper & Lower surfaces_VECTORISATION

#UPPER SURFACE
set R_up [lindex $file_data2 0]
set B_up_0 [expr sqrt ($R_up * 2)]
set B_up_1 [lindex $file_data2 4]
set B_up_2 [lindex $file_data2 5]
set B_up_3 [lindex $file_data2 6]
```

```
set B_up_4 [lindex $file_data2 7]
set B_up_5 [lindex $file_data2 8]
set B_up_6 [lindex $file_data2 9]
set dZe_up [lindex $file_data2 2]
set theta_up [lindex $file_data2 16]
set B_up_7 [expr $dZe_up + $theta_up]

#LOWER SURFACE
set R_lo [lindex $file_data2 1]
set B_lo_0 [expr -1 * sqrt($R_lo * 2)]
set B_lo_1 [lindex $file_data2 10]
set B_lo_2 [lindex $file_data2 11]
set B_lo_3 [lindex $file_data2 12]
set B_lo_4 [lindex $file_data2 13]
set B_lo_5 [lindex $file_data2 14]
set B_lo_6 [lindex $file_data2 15]
set dZe_lo [lindex $file_data2 3]
set theta_lo [lindex $file_data2 17]
set B_lo_7 [expr $dZe_lo + $theta_lo]

# BERSTEIN POLYNOMIALS
# BERSTEIN POLYNOMIAL ORDER
set n 7

# FACTORIAL
proc Factorial {x} {
    set i 1; set product 1
    while {$i <= $x} {
```

```
        set product [expr $product * $i]
        incr i
    }
    return $product
}
set b7 [Factorial $n]
set b6 [Factorial 6]
set b5 [Factorial 5]
set b4 [Factorial 4]
set b3 [Factorial 3]
set b2 [Factorial 2]
set b1 [Factorial 1]

set K0 1
set K1 [expr $b7 / $b6]
set K2 [expr $b7 / ($b2 * $b5)]
set K3 [expr $b7 / ($b3 * $b4)]
set K4 [expr $b7 / ($b4 * $b3)]
set K5 [expr $b7 / ($b5 * $b2)]
set K6 [expr $b7 / ($b6 * $b1)]
set K7 1
set a0 [lrepeat $len 1]
set a1 $left
foreach x $left {
        lappend a2 [expr pow($x, 2)]
}
foreach x $left {
        lappend a3 [expr pow($x, 3)]
```

```
}
foreach x $left {
        lappend a4 [expr pow($x, 4)]
}
foreach x $left {
        lappend a5 [expr pow($x, 5)]
}
foreach x $left {
        lappend a6 [expr pow($x, 6)]
}
foreach x $left {
        lappend a7 [expr pow($x, 7)]
}

foreach x $left {
        lappend c0 [expr pow(1 - $x, 7)]
}
foreach x $left {
        lappend c1 [expr pow(1 - $x, 6)]
}
foreach x $left {
        lappend c2 [expr pow(1 - $x, 5)]
}
foreach x $left {
        lappend c3 [expr pow(1 - $x, 4)]
}
foreach x $left {
        lappend c4 [expr pow(1 - $x, 3)]
```

```
}
foreach x $left {
        lappend c5 [expr pow(1 - $x, 2)]
}
foreach x $left {
        lappend c6 [expr pow(1 - $x, 1)]
}
foreach x $left {
        lappend c7 [expr pow(1 - $x, 0)]
}

#Matrix

# ---------------- SK0 --------------------------------
foreach y $a0 z $c0 {
        lappend x0 [expr $y * $z]
}
foreach y $x0 {
        lappend Sk0 [expr $y * 1]
}
# ---------------- SK1 --------------------------------
foreach y $a1 z $c1 {
        lappend x1 [expr $y * $z]
}
foreach y $x1 {
        lappend Sk1 [expr $y * 7]
}
# ---------------- SK2 --------------------------------
```

```
foreach y $a2 z $c2 {
        lappend x2 [expr $y * $z]
}
foreach y $x2 {
        lappend Sk2 [expr $y * 21]
}
# ---------------- SK3 ----------------------------
foreach y $a3 z $c3 {
        lappend x3 [expr $y * $z]
}
foreach y $x3 {
        lappend Sk3 [expr $y * 35]
}
# ---------------- SK4 ----------------------------
foreach y $a4 z $c4 {
        lappend x4 [expr $y * $z]
}
foreach y $x4 {
        lappend Sk4 [expr $y * 35]
}
# ---------------- SK5 ----------------------------
foreach y $a5 z $c5 {
        lappend x5 [expr $y * $z]
}
foreach y $x5 {
        lappend Sk5 [expr $y * 21]
}
# ---------------- SK6 ----------------------------
```

```
foreach y $a6 z $c6 {
        lappend x6 [expr $y * $z]
}
foreach y $x6 {
        lappend Sk6 [expr $y * 7]
}
# ---------------- SK7 ------------------------------
foreach y $a7 z $c7 {
        lappend x7 [expr $y * $z]
}
foreach y $x7 {
        lappend Sk7 [expr $y * 1]
}

# CST CLASS FUNCTION
set N1 [lindex $file_data2 18]
set N2 [lindex $file_data2 19]

foreach x $left {
        lappend c_1 [expr pow($x, $N1)]
}
foreach x $left {
        lappend c_2 [expr pow(1 - $x, $N2)]
}
foreach x $c_1 y $c_2 {
        lappend C [expr $x * $y]
}
```

```
# Z - COORDINATES


#Lower coordinates
foreach x $Sk0 {
        lappend S_lo0 [expr $x * $B_lo_0]
}
foreach x $Sk1 {
        lappend S_lo1 [expr $x * $B_lo_1]
}
foreach x $Sk2 {
        lappend S_lo2 [expr $x * $B_lo_2]
}
foreach x $Sk3 {
        lappend S_lo3 [expr $x * $B_lo_3]
}
foreach x $Sk4 {
        lappend S_lo4 [expr $x * $B_lo_4]
}
foreach x $Sk5 {
        lappend S_lo5 [expr $x * $B_lo_5]
}
foreach x $Sk6 {
        lappend S_lo6 [expr $x * $B_lo_6]
}
foreach x $Sk7 {
        lappend S_lo7 [expr $x * $B_lo_7]
}
```

```
foreach x $S_lo0 a $S_lo1 b $S_lo2 c $S_lo3 d $S_lo4 e
   $S_lo5 f $S_lo6 g $S_lo7 {
        lappend S_lo [expr $x + $a + $b + $c + $d + $e + $f
            + $g]
}
foreach x $S_lo y $C {
        lappend z_lo [expr $x * $y]
}

#Upper coordinates
foreach x $Sk0 {
        lappend S_up0 [expr $x * $B_up_0]
}
foreach x $Sk1 {
        lappend S_up1 [expr $x * $B_up_1]
}
foreach x $Sk2 {
        lappend S_up2 [expr $x * $B_up_2]
}
foreach x $Sk3 {
        lappend S_up3 [expr $x * $B_up_3]
}
foreach x $Sk4 {
        lappend S_up4 [expr $x * $B_up_4]
}
foreach x $Sk5 {
        lappend S_up5 [expr $x * $B_up_5]
}
```

```
foreach x $Sk6 {
        lappend S_up6 [expr $x * $B_up_6]
}
foreach x $Sk7 {
        lappend S_up7 [expr $x * $B_up_7]
}

foreach x $S_up0 a $S_up1 b $S_up2 c $S_up3 d $S_up4 e
   $S_up5 f $S_up6 g $S_up7 {
        lappend S_up [expr $x + $a + $b + $c + $d + $e + $f
            + $g]
}

foreach x $S_up y $C {
        lappend z_up [expr $x * $y]
}


# CREATE AIRFOIL GEOMETRY

# Create upper airfoil surface
set airUpper [pw::Application begin Create]
set airUpperPts [pw::SegmentSpline create]

for {set i 0} {$i < [llength $left]} {incr i} {
        $airUpperPts addPoint [list [lindex $left $i] [
           lindex $z_up $i] 0]
}
```

```
set airUpperCurve [pw::Curve create]
$airUpperCurve addSegment $airUpperPts
$airUpper end

# Create lower airfoil surface
set airLower [pw::Application begin Create]
set airLowerPts [pw::SegmentSpline create]

for {set i 0} {$i < [llength $left]} {incr i} {
        $airLowerPts addPoint [list [lindex $left $i] [
           lindex $z_lo $i] 0]
}

set airLowerCurve [pw::Curve create]
$airLowerCurve addSegment $airLowerPts
$airLower end

pw::Display resetView

set _DB(1) [list $airUpperCurve]
set _DB(2) [list $airLowerCurve]


# CREATE MESH O-GRID

set _TMP(PW_9) [pw::Connector createOnDatabase
   -parametricConnectors Aligned -merge 0 -reject _TMP(
```

```
   unused) [list $_DB(1) $_DB(2)]]
#puts "TMP = $_TMP(PW_9)"
set data1 [split $_TMP(PW_9) "_"]
set _CN(1) [lindex $data1 0]
set _CN(2) [lindex $data1 1]
unset _TMP(unused)
unset _TMP(PW_9)
pw::Application markUndoLevel {Connectors On DB Entities}

set _TMP(PW_10) [pw::Collection create]
$_TMP(PW_10) set [list $_CN(1) $_CN(2)]
$_TMP(PW_10) do setDimension 350
$_TMP(PW_10) delete
unset _TMP(PW_10)

pw::Application markUndoLevel {Dimension}

set _TMP(PW_11) [pw::Collection create]
$_TMP(PW_11) set [list $_CN(1) $_CN(2)]
$_TMP(PW_11) do setRenderAttribute PointMode All
$_TMP(PW_11) delete
unset _TMP(PW_11)
pw::Application markUndoLevel {Modify Entity Display}

set _TMP(mode_5) [pw::Application begin Modify [list $_CN
   (1) $_CN(2)]]
set _TMP(PW_12) [$_CN(1) getDistribution 1]
$_TMP(PW_12) setBeginSpacing 0.01
```

```
unset _TMP(PW_12)
set _TMP(PW_13) [$_CN(2) getDistribution 1]
$_TMP(PW_13) setBeginSpacing 0.01
unset _TMP(PW_13)
$_TMP(mode_5) end
unset _TMP(mode_5)
pw::Application markUndoLevel {Change Spacings}

set _TMP(mode_6) [pw::Application begin Modify [list $_CN
   (1) $_CN(2)]]
set _TMP(PW_14) [$_CN(1) getDistribution 1]
$_TMP(PW_14) setEndSpacing 0.01
unset _TMP(PW_14)
set _TMP(PW_15) [$_CN(2) getDistribution 1]
$_TMP(PW_15) setEndSpacing 0.01
unset _TMP(PW_15)
$_TMP(mode_6) end
unset _TMP(mode_6)
pw::Application markUndoLevel {Change Spacings}

set _TMP(mode_7) [pw::Application begin Create]
set _TMP(PW_16) [pw::Edge createFromConnectors [list $_CN
   (1) $_CN(2)]]
set _TMP(edge_2) [lindex $_TMP(PW_16) 0]
unset _TMP(PW_16)

set _TMP(dom_2) [pw::DomainStructured create]
$_TMP(dom_2) addEdge $_TMP(edge_2)
```

```
$_TMP(mode_7) end
unset _TMP(mode_7)
set _TMP(mode_8) [pw::Application begin ExtrusionSolver [
   list $_TMP(dom_2)]]
$_TMP(mode_8) setKeepFailingStep true

set _DM(1) $_TMP(dom_2)

$_DM(1) setExtrusionSolverAttribute NormalMarchingVector {
   -0 -0 -1}
$_DM(1) setExtrusionSolverAttribute NormalInitialStepSize 0
   .0005
$_DM(1) setExtrusionSolverAttribute SpacingGrowthFactor 1.1
$_DM(1) setExtrusionSolverAttribute StopAtHeight Off
$_DM(1) setExtrusionSolverAttribute StopAtHeight 150
$_TMP(mode_8) run 100

$_TMP(mode_8) run -1

$_TMP(mode_8) end
unset _TMP(mode_8)
unset _TMP(dom_2)
unset _TMP(edge_2)
pw::Application markUndoLevel {Extrude, Normal}

# SELECT SOLVER, DIMENSIONS AND BOUNADRY CONDITIONS

pw::Application setCAESolver {Stanford ADL/SU2} 3
```

```
pw::Application markUndoLevel {Select Solver}

pw::Application setCAESolver {Stanford ADL/SU2} 2
pw::Application markUndoLevel {Set Dimension 2D}

set _TMP(PW_42) [pw::BoundaryCondition getByName {
   Unspecified}]
set _TMP(PW_43) [pw::BoundaryCondition create]
pw::Application markUndoLevel {Create BC}

set _TMP(PW_44) [pw::BoundaryCondition getByName {bc-2}]
unset _TMP(PW_43)
$_TMP(PW_44) apply [list [list $_DM(1) $_CN(2)] [list $_DM
   (1) $_CN(1)]]
pw::Application markUndoLevel {Set BC}

$_TMP(PW_44) setName "airfoil"
pw::Application markUndoLevel {Name BC}

set _TMP(PW_45) [pw::BoundaryCondition create]
pw::Application markUndoLevel {Create BC}

set _TMP(PW_46) [pw::BoundaryCondition getByName {bc-3}]
unset _TMP(PW_45)

set node [$_CN(1) getNode End]
foreach cnctr [pw::Connector getConnectorsFromNode $node] {
    lappend con2 [lindex $cnctr]
```

```
        puts [$cnctr getName]
}
puts "con2 = $con2"
set data3 [split $con2 " "]
set con3 [lindex $data3 2]
puts "con3 = $con3"

set node [$con3 getNode End]
foreach cnctr2 [pw::Connector getConnectorsFromNode $node]
   {
    lappend con4 [lindex $cnctr2]
        puts [$cnctr2 getName]
}
puts "con4 = $con4"
set data4 [split $con4 " "]
set _CN(3) [lindex $data4 1]
puts "CN(3) = $_CN(3)"

$_TMP(PW_46) apply [list [list $_DM(1) $_CN(3)]]
pw::Application markUndoLevel {Set BC}

$_TMP(PW_46) setName "farfield"
pw::Application markUndoLevel {Name BC}

unset _TMP(PW_42)
unset _TMP(PW_44)
unset _TMP(PW_46)
```

```
# EXPORT CAE

set _TMP(mode_10) [pw::Application begin CaeExport [
   pw::Entity sort [list $_DM(1)]]]
$_TMP(mode_10) initialize -strict -type CAE {C:/Users/
   s293169/Documents/SU2/SU2/Data_base/Mesh_pointwise/
   CST_Coeff_1_deg.su2}
$_TMP(mode_10) setAttribute FilePrecision Double
$_TMP(mode_10) verify
$_TMP(mode_10) write

$_TMP(mode_10) end
unset _TMP(mode_10)
```

Appendix D

SU2 Configuration file

```
PHYSICAL_PROBLEM= EULER
MATH_PROBLEM= DIRECT
RESTART_SOL= NO
KIND_TURB_MODEL= NONE
MACH_NUMBER= 0.8
AOA= 1
FREESTREAM_PRESSURE= 101325
FREESTREAM_TEMPERATURE= 273.15
REYNOLDS_NUMBER= 10.3E6
GAMMA_VALUE= 1.4
GAS_CONSTANT= 287.87
REF_ORIGIN_MOMENT_X = 0.25
REF_ORIGIN_MOMENT_Y = 0.00
REF_ORIGIN_MOMENT_Z = 0.00
REF_AREA= 1.0
REF_DIMENSIONALIZATION= FREESTREAM_VEL_EQ_MACH
MARKER_EULER= ( airfoil )
MARKER_FAR= ( farfield )
```

```
MARKER_PLOTTING = ( airfoil )
MARKER_MONITORING = ( airfoil )
MARKER_DESIGNING = ( airfoil )
NUM_METHOD_GRAD= WEIGHTED_LEAST_SQUARES
OBJECTIVE_FUNCTION= DRAG
CFL_NUMBER= 0.5
EXT_ITER= 1000
LINEAR_SOLVER= BCGSTAB
LINEAR_SOLVER_PREC= LU_SGS
LINEAR_SOLVER_ERROR= 1E-6
LINEAR_SOLVER_ITER= 5
MGLEVEL= 3
MGCYCLE= W_CYCLE
MG_PRE_SMOOTH= ( 1, 2, 3, 3 )
MG_POST_SMOOTH= ( 0, 0, 0, 0 )
MG_CORRECTION_SMOOTH= ( 0, 0, 0, 0 )
MG_DAMP_RESTRICTION= 1.0
MG_DAMP_PROLONGATION= 1.0
CONV_NUM_METHOD_FLOW= JST
MUSCL_FLOW= YES
SLOPE_LIMITER_FLOW= VENKATAKRISHNAN
JST_SENSOR_COEFF= ( 0.5, 0.02 )
TIME_DISCRE_FLOW= EULER_IMPLICIT
CONV_CRITERIA= CAUCHY
RESIDUAL_REDUCTION= 6
RESIDUAL_MINVAL= -8
STARTCONV_ITER= 10
CAUCHY_ELEMS= 100
```

```
CAUCHY_EPS= 1E-6
CAUCHY_FUNC_FLOW= DRAG
MESH_FILENAME= 337.su2
MESH_FORMAT= SU2
MESH_OUT_FILENAME= mesh_out.su2
SOLUTION_FLOW_FILENAME= solution_flow.dat
SOLUTION_ADJ_FILENAME= solution_adj.dat
OUTPUT_FORMAT= PARAVIEW
CONV_FILENAME= history
RESTART_FLOW_FILENAME= restart_flow.dat
RESTART_ADJ_FILENAME= restart_adj.dat
```