# INTELLIGENT DATA-DRIVEN AERODYNAMIC ANALYSIS AND OPTIMIZATION OF MORPHING CONFIGURATIONS

A Dissertation
Presented to
The Academic Faculty

By

José Messias Magalhães Júnior

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Robotics,
School of Aerospace Engineering

Georgia Institute of Technology

May 2024

**INTELLIGENT DATA-DRIVEN AERODYNAMIC ANALYSIS AND OPTIMIZATION OF MORPHING CONFIGURATIONS**

Dissertation Committee:


Prof. Kyriakos G. Vamvoudakis (advisor)
School of Aerospace Engineering
*Georgia Institute of Technology*

Prof. Seth Hutchinson
Institute for Robotics and Intelligent Machines
*Georgia Institute of Technology*

Dr. Gustavo Luiz Olichevis Halila
Technology Development
*Embraer S.A. - Brazil*

Prof. Lakshmi N. Sankar
School of Aerospace Engineering
*Georgia Institute of Technology*

Prof. Daniel P. Schrage
School of Aerospace Engineering
*Georgia Institute of Technology*



Date approved: December 8, 2023

”The Lord is my strength and my shield; my heart trusts in Him, and He helps me.”

*Psalm 28:7*

For my dear parents Jose Messias Magalhães and Edevanildes Duarte Magalhães.

Thank you God.

## ACKNOWLEDGMENTS

The completion of my Ph.D. journey at Georgia Institute of Technology was challenging, long and above of all, rewarding. I would like to thank all professors, students, staff and friends that I have met on this journey and that contributed to completing my cycle in this great school.

I would like to give thanks to my advisor Professor Kyriakos G. Vamvoudakis who accepted me as his PhD student and provided the directions, support and all resources needed to do this work.

I would like to thank Dr Gustavo Halila, Dr Daniel Schrage, Dr Lakshni Sankar and Dr Seth Hutchinson for being members of my committee and devoting their time to read and give feedback about my research. Special thanks to Dr Halila, a friend that I made on this journey. Due to his guidance, I was able to have a further understanding about Aerodynamics and Shape Optimization.

I would like to thank the sponsors who supported me during these years as PhD student: Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES - Brazil) and Aerospace Engineering department at Georgia Tech.

I thank the staff of the school of AE and Robotics program at Georgia Tech, in particular Oksana Gomas and Howard Simpson.

I would like to thank all great people that I met at Georgia Tech and made my life happier and lighter on campus: Harold Nikoue, Nicolas Shu, Mark Mote, Raphael Cohen, Thanakorn Khamvilai, Chams Mballo, Olatunde Sanni.

I express my gratitude to the members of the ICPS Lab: Joshua Netter, Nejat Tukenmez, Filippos Fotiadis, Nick-Marios Kokolakis. Members of the next-door lab: Adam Garlow and Anika Kansky. And member of the front door lab: Dhruv Purushotham.

I am grateful to my great Brazilian friends: Emilio, Denise, Carlos, Cris, Alessandra, Priscila, and special thanks to John Queiroz, a man who I learned to admire and who

became my family in USA.

I am grateful for having met Christie Ross and her family. Their love and support made my last year so happy.

Finally, I am blessed for having a such great family that supported me all these years. Thank you Mom, Thank you Dad !

## TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# LIST OF ACRONYMS

AD ow open-source computational uid dynamics solver

ANK Approximate Newton-Krylov

ASO Aerodynamic Shape Optimization

CFD Computational Fluid Dynamics

CPS Cyber-Physical Systems

DNN Deep Neural Network

DOF Degree of Freedom

EASA European Aviation Safety Agency

ELU Exponential Linear Unit

FAA Federal Aviation Administration

FFD Free-Form Deformation

FTA Fault Tree Analysis

HF-Model High-Fidelity Model

HPC High-Performance Computing

MAPE Mean Absolute Percentage Error

MGDA Multi-Gradient Search Algorithm

MTD Moving Target Defense

MUSCL Monotone Upstream-Centered Scheme for Conservation Laws

NoC Network-on-Chip

PACE Partnership for an Advanced Computing Environment

PTC Pseudo-Transient Continuation

PyGeM Python Geometrical Morphing

RANS Reynolds-Averaged Navier–Stokes

ReLu Recti ed Linear Unit

SA Spalart-Allmaras

UAV Unmanned Aerial Vehicle

## SUMMARY

The aeronautical industry is continuously looking for more ef cient aircraft and provide a reduction on fuel or power consumption while guaranteeing safety, optimality, and stability. The advances of composite materials enable building morphing structures that adapt to a variety of  ight and environmental conditions. Airplanes that use morphing technologies can achieve optimal performance and minimize the drag over the entire  ight envelope and operate even in dangerous weather conditions.

In this dissertation, we propose a data-driven framework to control morphing airfoils in the subsonic  ight regime, considering high Reynolds numbers to reach, in ef cient and safe way, a shape with improved values of the aerodynamic coef cients. The online solution is based on a data-driven controller combined with a surrogate model and a multi-gradient descent algorithm considering objective functions that are relevant in aerodynamics: increase lift-drag ratio, reduce drag and increase lift. Without full knowledge of the aerodynamic parameters (lift, drag, and pitching moment coef cients), the learning framework searches for an airfoil shape that minimizes a metric of performance associated to drag, lift, and pitching moment coef cients. The solution uses online data to improve the accuracy of the predictions of the aerodynamic coef cients provided by the surrogate model along the trajectory. The optimization framework focuses on subtle airfoil deformations to assure a smooth trajectory between the initial and the  nal shape. Finally, the ef cacy and the robustness of our proposed solution is shown in numerical examples, resulting in a signi cant reduction in the prediction error.

# CHAPTER 1

# INTRODUCTION

## 1.1 Motivation

In the aerospace industry [1, 2], engineers are continuously exploring the design space to produce a more ef cient and safe aircraft. One relevant performance metric can be fuel consumption, which is directly affected by the drag, so that a constrained drag minimization is a constant target in aeronautical design. Drag reduction is a desired target due to fuel burn cutback, which has a positive impact on greenhouse gas emissions. To achieve reduced drag, aerodynamic shape optimization is currently used. Numerical optimization can enhance the consistent exploration of the design space and lead to trade-offs which would be dif cult to achieve without resorting to numerical tools. One strategy to achieve reduced drag is the use of morphing con gurations [3, 4, 5, 6].

Since a variety of ight conditions are usually encountered in daily operations, a single optimal shape is, in general, incapable of minimizing drag over the entire ight envelope. To this end, aircraft leveraging morphing technologies are better suited to attain optimal aerodynamic performance under various ight conditions. The advent of composite materials and advanced control techniques enable the design of morphing structures that can adapt to varying ight conditions with optimal performance. Moreover, morphing con gurations eliminate the presence of gaps and steps which are typical in high-lift devices. Airplanes that employ morphing technologies can achieve optimal performance by minimizing drag over the entire ight envelope and leverage the use of laminar ow technologies [7] by performing shape optimization.

To perform aerodynamic analysis and shape optimization [8], one needs a ow solver that can account for the physical and environmental phenomena of interest. The increasing

computational power allows aerodynamicists to use increasingly complex computational aerodynamics tools. The use of Computational Fluid Dynamics (CFD) became the norm in airplane design [9, 10] [10] but despite the availability of High-Performance Computing (HPC) facilities, traditional CFD computations are still time consuming and computationally inef cient. In the early design phases, the engineers investigate many con gurations and, therefore, the time required to perform each aerodynamic analysis is relevant. To alleviate computational requirements, traditional CFD solvers can be adopted to generate high- delity aerodynamic data that will, in turn, be employed to calibrate potential surrogate models.

## 1.2 Research Contributions

This dissertation aims to develop a framework to control ying machines that employ morphing technologies in high Reynolds number ows in different ight conditions. To achieve this goal, we will explore the use of a morphing airfoil controlled by 20 Free-Form Deformation (FFD) control points by focusing on four aspects: high- delity Computational Fluid Dynamics simulations, surrogate models based on Deep Neural Networks (DNNs), shape optimization based on multi-gradient descent approach and learning algorithm using online measurement data. In this work, we will not evaluate constraints related to structure and weight. These constraints can be added to the proposed solution as geometric constraints to be discussed in future work.

The CFD simulations will be performed using open-source computational uid dynamics solver (AD ow), an open source CFD solver with options to solve Euler, laminar Navier-Stokes, and Reynolds-Averaged Navier–Stokes (RANS) equations in steady, unsteady, and time-spectral modes [11]. The numerical methods and models in AD ow have been validated in [7, 12] against experimental airfoil data. To address the surrogate models, we will train three fully-connected DNNs in different ight conditions using shapes generated from morphing airfoil with the respective aerodynamic associated provided by the

CFD simulations.

Next, we will perform a multi-objective shape optimization based on a proposed multi-gradient descent approach that evaluates variation of the pitching moment coef cient. We will consider objective functions that are relevant in aerodynamics: increase lift-drag ratio, reduce drag, and increase lift.

Finally, we will propose an online learning algorithm that improves the performance level and the ef ciency of the shape morphing process during the ight by using online measurements to improve the prediction of $c_l$, $c_d$ and $c_m$.

The overall contributions of this dissertation are as follows,

- A data-based framework for the operation of ying machines that employ morphing technologies.
- An online learning algorithm to reduce the prediction errors of the surrogate models and improve the performance level and ef ciency of the morphing process in ight.
- A search algorithm based on multi-gradient descent approach that takes into account three objective functions: increase lift-drag ratio, reduce drag and increase lift.
- Analysis of sensitive and robustness of the proposed learning algorithm.

## 1.3 Organization

The remainder of the dissertation is structured as follows. Chapter 2 describes works related to cyber-physical systems with focus on three aspects: Data-driven decision making; morphing con gurations and safety assessment. Chapter 3 discusses shape optimization and presents the problem formulation. Chapter 4 focuses on the morphing airfoil and on the development of the surrogate models based on simulations using computational uid dynamics. Chapter 5 discusses the multi-gradient search algorithm considering the three cost functions relevant ins aerodynamic. Chapter 6 describes the algorithm proposed in

this work to predict the values of $c_l$, $c_d$, and $c_m$ of a sequence of shapes. Chapter 6 describes the data driven decision making algorithm to be used to prevent the system from reaching undesired conditions. Finally, in Chapter 7 we provide concluding remarks and recommendations for future work directions.

# CHAPTER 2

# SHAPE OPTIMIZATION

## 2.1 Introduction and Related Work

In the aerospace industry [1, 2], engineers are continuously exploring the design space to produce a more ef cient and safe aircraft. One relevant performance metric can be fuel consumption, which is directly affected by the drag, so that a constrained drag minimization is a constant target in aeronautical design. Since a variety of ight conditions are usually encountered in daily operations, a single optimal shape is, in general, incapable of minimizing drag over the entire ight envelope. To this end, aircraft leveraging morphing technologies are better suited to attain optimal aerodynamic performance under various ight conditions. In this study, we evaluate the use of an airfoil able to optimally morph vertically in ight to achieve a shape with reduced drag for a given ight condition. In order to improve the level of safety and ef ciency, a framework is proposed that combines data-driven analysis, optimization, and control-theoretic tools to optimally morph airfoil in ight. The NACA 2412airfoil is considered as the baseline con guration to be morphed since it is widely used in low-speed, general aviation airfoils. It is the airfoil used for single engine Cessna aircraft, such as the Cessna 152, which attains maximum ight speeds no greater than 0.2 Mach. Figure 2.1 illustrates the airfoil under analysis.

Figure 2.1: Airfoil NACA 2412

To perform aerodynamic analysis and shape optimization [8], one needs a ow solver that can account for the physical and environmental phenomena of interest. The increasing computational power allows aerodynamicists to use increasingly complex computational aerodynamics tools. The use of CFD became the norm in airplane design [9, 10] but despite the availability of HPC facilities, traditional CFD computations are still time consuming and computationally inef cient. In the early design phases, the engineers investigate many con gurations and, therefore, the time required to perform each aerodynamic analysis is relevant. To alleviate computational requirements, traditional CFD solvers can be adopted to generate high- delity aerodynamic data that will, in turn, be employed to calibrate potential surrogate models.

The work of [13] proposes model-based convolutional neural networks to perform airfoil aerodynamic analysis. The authors in [14] use a B-spline-based generative adversarial network model for shape parameterization associated with a combination of multilayer perceptron, recurrent neural networks, and a mixture of experts for surrogate modeling to enable airfoil aerodynamic predictions over a range of Mach and Reynolds numbers. The investigation in [14] includes aerodynamic shape optimization and other data-driven aerodynamic simulations are developed in [15, 16, 17, 18, 19, 20, 21] and the references therein.

Researchers have shown strong interest in morphing structures over the years. Deter-

mining the geometric parameters of an aerodynamically ef cient airfoil and morphing is challenge. For instance, in [22], the authors propose a methodology to obtain the optimal shape of a morphing block by using machine learning and adaptive dynamic inversion control capable of learning the required shape of a morphing smart block for a given trajectory. With a numerical example, the authors show that the method can learn the commands that produce the optimal shape while in [23], they extend the methodology to an air-vehicle using Q-learning to obtain the optimal shape change policy. In [24] [24], the authors use a Q-learning method combined with analytical aerodynamic calculations to determine the optimal shape by changing the parameters of maximum camber location, airfoil angle of attack, and thickness. The work of [25] developed an aerodynamic model and a dynamic model of a morphing ying wing aircraft by using a constant strength source doublet panel method. The developed model provides a versatile tool for calculating the aerodynamic forces on the morphing aircraft.

The investigation in [26] proposes a deep neural network and reinforcement leaning technique as a control strategy for shape-memory alloy actuators in the context of a morphing wing. A deep deterministic policy gradient (DDPG) algorithm based on the Actor-Critic framework is proposed in [27] to solve the problem of deep reinforcement learning morphing control in continuous action space applied to a bionic bird wing-foldable Unmanned Aerial Vehicle (UAV) model.

In [25], the authors proposed an aerodynamic dynamic model of a morphing ying wing aircraft by using a constant strength source doublet panel method. The developed model provides a versatile tool for calculating the aerodynamic forces of the morphing aircraft. The authors in [28] present a novel active wing morphing aerostructure concept by combining advanced composite lattice-based cellular materials/components with exible wings.

Unlike the aforementioned studies, this dissertation aims to study the advantages of deep neural networks with data-driven shape control to determine in ght an appropriate

optimal morphing shape for efficiency and safety. Furthermore, the trajectory between the initial and final shape is determined to minimize the effect of morphing on the aircraft stability characteristics.

## 2.2 Problem Formulation

Define the morphing airfoil as a discrete-time dynamical system of the form

$$\begin{aligned} x[k+1] &= x[k] + u[k] \\ y[k] &= f(x[k]) = \left[ f_{c_l}(x[k]), f_{c_d}(x[k]), f_{c_m}(x[k]) \right]^T, \quad k = 0, 1, \ldots, L, \; x[0] = x, \end{aligned} \tag{2.1}$$

where $x[k] \in R^{20}$ is the state of the system, $u[k] \in R^{20}$ is the control input that drives the position of the ordered pairs, and $y[k] \in R^3$ is the output vector consisting of the lift, drag, and pitching moment coefficients at a given shape, i.e., $f = \left[ f_{c_l}, f_{c_d}, f_{c_m} \right]^T$, where $f_{c_l} : R^{20} \to R$, $f_{c_d} : R^{20} \to R$, and $f_{c_m} : R^{20} \to R$ denote the unknown nonlinear functions representing the lift, drag, and pitching moment coefficients, respectively. The following assumptions are now needed.

Assumption 1. The system (Equation 2.1) is stabilizable and $x[k]$ can be measured for all values of $k \in \{0, 1, \ldots, L\}$. ∎

Assumption 2. The nonlinear functions $f_{c_l}$, $f_{c_d}$, and $f_{c_m}$ are bounded and continuous. Moreover, small variations on the shapes $x[k]$ for all values of $k \in \{0, 1, \ldots, L\}$ imply small variations on the values of $f_{c_l}$, $f_{c_d}$, and $f_{c_m}$. ∎

Assumption 3. The flight condition, defined by the angle of attack ($\alpha$), Reynolds (Re), and Mach numbers, remains unchanged during the whole morphing procedure. ∎

Let $X \subseteq R^{20}$ be the set of all the states reachable from the origin in $k$ steps of the system (Equation 2.1), i.e., the set of all states $x[k]$ obtained starting from the initial condition

$x^{(0)}$. The purpose of this work is to find an improved shape denoted as $x$ that has an increased value of $c_l$, an increased value of $c_l/c_d$, a reduced value of $c_d$, and a desired $|c_m|$ not greater than the initial value. The improved shape can be found by solving the following multi-objective optimization problem:

$$\begin{aligned}
\underset{u \in R^{20}}{\text{minimize}} \quad & F(x) = \left[ \frac{f_{c_l}(x^{(k+1)})}{f_{c_d}(x^{(k+1)})}, \; f_{c_l}(x^{(k+1)}), \; f_{c_d}(x^{(k+1)}) \right] \\
\text{subject to} \quad & x^{(k+1)} = x^{(k)} + u^{(k)}, \quad \forall k \in \{0, \ldots, L-1\} \\
& f_{c_l}(x^{(L)}) \geq f_{c_l}(x^{(0)}) \\
& f_{c_d}(x^{(L)}) \leq f_{c_d}(x^{(0)}) \\
& |f_{c_m}(x^{(L)})| \leq |f_{c_m}(x^{(0)})| \\
& x^{(0)} = x.
\end{aligned} \tag{2.2}$$

Since $f_{c_l}(\cdot)$, $f_{c_d}(\cdot)$, and $f_{c_m}(\cdot)$ are unknown nonlinear functions, a surrogate model must be constructed. This problem can be summarized as follows.

Problem 1. Find a shape with higher $\frac{c_l}{c_d}$, higher $c_l$, lower $c_d$, and feasible $c_m$ that can be tractably estimated given that the functions $f_{c_l}(\cdot)$, $f_{c_d}(\cdot)$, and $f_{c_m}(\cdot)$ are unknown. ∎

Since morphing the wing airfoil can affect the stability of the aircraft, we will morph the shape while we guarantee efficiency and safety. Efficiency means that morphing occurs only when a new shape $x^{(k+1)}$ with improved values of $f_{c_l}(\cdot)$, $f_{c_d}(\cdot)$, and $f_{c_m}(\cdot)$ exists and can be reached; while safety means that the trajectory from the baseline to the new shape will not reach an intermediate shape with undesired values. Thus, the constraints can be mathematically described as follows:

$$\begin{aligned}
f_{c_l}(x^{(k+1)}) &\geq 0.98 f_{c_l}(x^{(0)}) \\
f_{c_d}(x^{(k+1)}) &\leq 1.02 f_{c_d}(x^{(0)}) \\
|f_{c_m}(x^{(k+1)})| &\leq 1.05 |f_{c_m}(x^{(0)})|, \\
\forall k &\in \{1, 2, \ldots, L-1\}.
\end{aligned} \tag{2.3}$$

It is known that a surrogate model has inherent errors on the prediction and thus in order to improve the prediction of $f_{c_l}(p,q)$, $f_{c_d}(p,q)$ and $f_{c_m}(p,q)$ we shall use data gathered along the system's generated trajectories. This problem can be summarized as follows.

Problem 2. Estimate $f_{c_l}(p,q)$, $f_{c_d}(p,q)$ and $f_{c_m}(p,q)$ in an online manner, using data gathered along the trajectory of (Equation 2.1) while preventing the system from reaching any undesired values during the morphing procedure. ∎

## 2.3 Conclusion

In this chapter, we formulate the problem addressed by this dissertation, a multi-objective optimization problem that involves objective functions relevant in Aerodynamic Shape Optimization (ASO) that are: increase the ratio $\frac{c_l}{c_d}(p,q)$, reduce $c_d$ and increase $c_l$. Unlike the classical ASO where we are interested in finding a single shape that optimize an objective function under given constraints, in this work we are interested not only in finding a final shape but also a trajectory to reach this final shape in a safe and efficient way.

# CHAPTER 3

# SURROGATE MODEL

In this section we will deal with Problem 1. The approach used for modeling is based on a Deep Neural Network (DNN) where the training dataset is generated from CFD techniques. Once the surrogate model is obtained, our proposed search algorithm is used to solve (Equation 2.2) and obtain an appropriate morphing shape.

To perform aerodynamic analysis and shape optimization, one needs a flow solver that can account for the physical phenomena of interest. Although the CFD simulations can provide accurate predictions of lift, drag and pitching moment coefficients, $c_l$, $c_d$ and $c_m$, respectively, they still require extensive computational power. This may potentially be an issue during the early design phases, during which many configurations must be explored. Thus, we propose the development of a surrogate model based on a DNN to accurately predict the aerodynamic parameters. This development consists of a generation of 2,000 morphing shapes from a baseline shape and the training of two fully connected neural networks.

We will focus on the aerodynamic shape optimization and we will constrain the maximum airfoil thickness so that it will not be smaller than the lower bound discussed in subsection 3.2.2. Additional considerations from other disciplines and from regulatory requirements should be included in the decision process considering best design practices.

## 3.1 Parametric Method

The FFD technique has been used in shape optimization [7, 29] and is based on the tensor product trivariate Bernstein polynomial [30, 31, 32] that inserts the shape representation into a rectangular box composed by the control points. A subsequent transformation recomputes the position of the solid points with respect to the modified points of the lattice.

The vectors $\vec{S}$, $\vec{T}$, and $\vec{U}$ define the size and the orientation of the control lattice. The lattice is uniformly layered by planes in their directions with several deformable cuboids to embed the object as shown in Figure 3.1. Given several established basis functions (trivariate Bernstein polynomials), the changing positions of the vertexes of the cuboids (FFD control points) cause the inner model to deform [29].

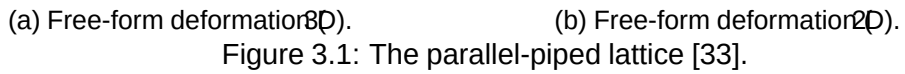
Every object point $X(x,y,z)$ interior to the control lattice has $(s,t,u)$ coordinates in the coordinate system fixed to the FFD box as follows,

$$\vec{X} = \vec{X}_0 + s\vec{S} + t\vec{T} + u\vec{U}. \tag{3.1}$$

Let $P_{ijk}, i = 0, \ldots, l; j = 0, \ldots, m; k = 0, \ldots, n$ be the control points on the lattice. The deformation is represented by a movement of the control points $P_{ijk}$ from their positions. The new position of an arbitrary point $X(x,y,z) = (x(s,t,u), y(s,t,u), z(s,t,u))$ inside the lattice is computed as,

$$X(s,t,u) = \sum_{i=0}^{l} \binom{l}{i} (1-s)^{l-i} s^i \sum_{j=0}^{m} \binom{m}{j} (1-t)^{m-j} t^j \sum_{k=0}^{n} \binom{n}{k} (1-u)^{n-k} u^k P_{ijk}. \tag{3.2}$$

Note that, FFD can then be used to deform an object in 2D or 3D spaces, regardless of the representation of this object. Instead of manipulating the surface of the object directly, we will move the FFD control points to obtain a new shape. The displacement of a point inside the lattice is described by a third order Bézier tensor product. The control lattice of the parallel-piped FFD design used to morph the airfoil shape is shown in Figure 3.1, which is implemented into a standalone Python package named Python Geometrical Morphing (PyGeM).



(a) Free-form deformation (3D). (b) Free-form deformation (2D).

Figure 3.1: The parallel-piped lattice [33].

Using the NACA 2412 airfoil as a baseline configuration, a new airfoil shape is obtained by vertically moving the FFD control points as shown in Figure 3.1b. The $x_s, y_s$ coordinates of the new shape are computed using (Equation 3.2).

## 3.2 Mesh Generation

The computational meshes for the airfoils considered in this work consist of 1,101 points along the surface and 193 points in the off-wall direction. The meshes are generated using a cluster, a cutting edge high performance computing (HPC) resource at Georgia Tech equipped with Dual Intel Xeon Gold 6226 CPUs @ 2.7 GHz (24 cores/node) [34]. The mesh generation approach is implemented in the open-source module pyHyp [35], a hyperbolic mesh generator used to generate high-quality structured mesh. Figure 3.2 shows the computational mesh for the NACA 2412 airfoil.

(a) Far field. (b) Airfoil wall.

Figure 3.2: Computational mesh with 14,029 cells.

### 3.2.1 ADflow-CFD Data Generation

To obtain the values of $c_l$, $c_d$, and $c_m$ (moment coefficient around the quarter chord) for different airfoil shapes, we will use high-fidelity CFD simulations performed with ADflow. ADflow is an open source CFD solver [11] and has options to solve Euler, laminar Navier-Stokes, and RANS equations in steady, unsteady, and time-spectral modes, with multi-block structured and overset meshes. The governing equations are discretized using the finite volume method with first and second order stencils. A Pseudo-Transient Continuation (PTC) strategy is used to convert the flow equations. The inviscid fluxes are discretized by using 3 different numerical schemes: (i) the scalar artificial dissipation scheme [36], (ii) a matrix dissipation scheme [37], and (iii) a Monotone Upstream-Centered Scheme for Conservation Laws (MUSCL) based on the works of [38] and [39]. The viscous flux gradients are calculated by using the Green-Gauss approach while the residual equations can be converged with four distinct algorithms. An Approximate Newton-Krylov (ANK) solver is implemented and can be used as a globalization scheme for the full NK algorithm [40]. We use the Spalart-Allmaras (SA) turbulence model [41] as closure for all CFD simulations that are part of this work. The numerical methods and models in ADflow have been validated in [7, 12] against experimental airfoil data.

For all CFD simulations used in the data generation, we choose a 12th order decay in total residuals magnitude as a relative convergence criterion. The relative convergence is de ned as,

$$\frac{||R_0^{pnq}||_2}{||R_0^{pfsq}||_2}, \tag{3.3}$$

where the superscript fs refers to the free-stream properties. We observe that this stopping criterion is reached, in general, after 2,000 nonlinear iterations for the test cases. For the SA turbulence model, used in this study, the convergence is proved in [40].

### 3.2.2 Shape Generation

For the CFD analysis, we discretize the airfoil using 1,102 points distributed over the surface. This quantity of points is necessary to obtain accurate values of lift, drag, and pitching moment (at the quarter chord point) coef cients, $c_l$, $c_d$, and $c_m$, respectively, for the high Reynolds number ow considered in this work.

To generate the new shapes from the baseline NACA 2412, we consider 20 FFD control points able to independently move in the vertical direction, i.e., 1 Degree of Freedom (DOF) for each control point. The control points are identi ed as $(x_1, x_2, \ldots, x_{20})$ and are evenly distributed as shown in Figure 3.1b. The control points may expand into small variations, resulting in small deformations around the original airfoil. The upper surface is de ned by the control points $(x_1, x_2, \ldots, x_{10})$ and the lower surface is de ned by $(x_{11}, x_{12}, \ldots, x_{20})$ by assuming values in the interval $[-2.5, 2.5]$ as described in Table 3.1. Figure 3.3 presents the upper and lower bounds and the morphed shapes in a linear trajectory obtained using the FFD control points with the geometric constraint. While Figure 3.4 presents the result of the CFD simulation of two different morphed shapes.

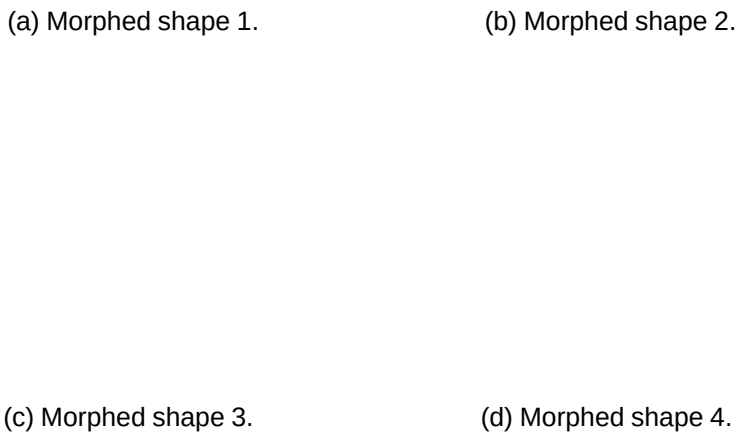
Table 3.1: Boundary of the FFD control points

| | max value | min value | upper bound shape | lower bound shape |
|---|---|---|---|---|
| $x_1, x_2, \ldots, x_{10}$ | 2.5 | -2.5 | 2.5 | -2.5 |
| $x_{11}, x_{12}, \ldots, x_{20}$ | 2.5 | -2.5 | -2.5 | 2.5 |

Figure 3.3: Shapes in a linear trajectory.

Figure 3.4: CFD Simulation of two morphed shapes.

The main goal of morphing the initial shape is to generate shapes with better values of aerodynamic parameters for a given ight condition. Every FFD control point can move independently in y-direction. Figure 3.5 illustrates four morphed shapes generated by moving the FFD control points.



(a) Morphed shape 1. (b) Morphed shape 2.

(c) Morphed shape 3. (d) Morphed shape 4.

Figure 3.5: Examples of morphed shapes under geometric constraints.

## 3.3 Data Set

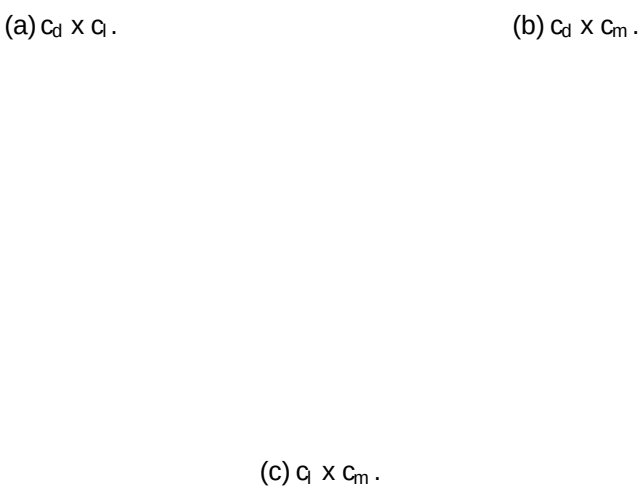
To train the DNNs used, we consider a ight condition de ned as $Re = 6$, Mach $= 0.2$, and $1.0$ and $9,600$ shapes such that $7,680$ shapes are used as training data, and $1,920$ as validation data. Finally, we generate $5,100$ more shapes to use as testing data and to evaluate the accuracy of the predictions. Table 3.2 summarizes the data set used to train the DNNs.

Table 3.2: Data set

| | Training data set | Validation data set | Testing data set |
|---|---|---|---|
| Data points size | 7,680 | 1,920 | 5,100 |

The numerical experiments to obtain $c_l$, $c_d$, and $c_m$ (moment coef cient around the quarter chord ) were performed on a computing node of the Partnership for an Advanced Computing Environment (PACE) cluster at the Georgia Institute of Technology [34]. AD- ow required, on average, $2$ hours to simulate $50$ morphed shapes using $8$ GB memory, and

19 cores. This run-time corresponds to a 12 th order decay in total residuals, which represents a standard engineering-level convergence for CFD analysis. Figure 3.6 describes how the 14;100 datapoints are distributed in terms of the aerodynamic parameters.



(a) $c_d$ x $c_l$.



(b) $c_d$ x $c_m$.



(c) $c_l$ x $c_m$.

Figure 3.6: Distribution of the 14;100 datapoints used to train and evaluate the DNNs.

## 3.4 Surrogate Model

Since the functions $f_{c_l}(p,q)$, $f_{c_d}(p,q)$, and $f_{c_m}(p,q)$ are not known, one can approximate these functions by generating data points by using CFD. To estimate $c_l$, $c_d$, and $c_m$ (moment coef cient around the quarter chord) in a more ef cient way, we propose to use three fully connected DNNs to approximate $f_{c_l}(p,q)$, $f_{c_d}(p,q)$, and $f_{c_m}(p,q)$. The input data of the DNN is the vector $X \in \mathbb{R}^{23}$ where the rst 3 elements de ne the ight condition, i.e., the triple (Re, Mach, ) and the remaining 20 elements de ne a unique airfoil shape.

To estimate $c_l$ and $c_d$ in a more ef cient way, we propose to use two fully connected

DNN to approximate $f_{c_l}(\alpha, M)$ and $f_{c_d}(\alpha, M)$ and predict each aerodynamic coefficient. The input data of the DNN is the vector $x \in \mathbb{R}^{23}$ where the first 20 elements define an unique airfoil shape and the remaining elements define the flight condition, i.e., values of Re, Mach and $\alpha$. The approximation of these functions via neural network with $N$ hidden layers can be mathematically expressed as follows,

$$z_m^{(1)} = \sum_{j=1}^{23} w_{jm}^{(0)} x_j + b_m^{(0)}, \quad m = 1, \ldots, M_1$$

$$a_k^{(i)} = \sum_{m=1}^{M_i} w_{mk}^{(i)} z_m^{(i)} + b_k^{(i)}, \quad k = 1, \ldots, K_i; \; i = 1, \ldots, N-1$$

$$z_m^{(i+1)} = \sigma(a_m^{(i)}), \quad i = 1, \ldots, N-1; \; m = 1, \ldots, M_{i+1}$$

$$\tilde{f}(\alpha) = \sum_{m=1}^{M_N} w_m^{(N)} z_m^{(N)}, \quad \alpha \in \{c_l, c_d\},$$

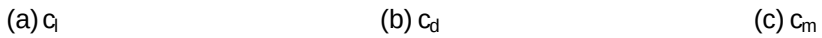
where $x_j$ is the $j^{th}$ element of the input data vector $x$, $\sigma$ is an activation function, $M_i$ and $K_i$ are the number of neurons and number of output respectively for $i^{th}$ hidden layer, $M_{i+1} = K_i$ holds since the two neural networks are fully-connected, $z_m^{(i)}$ is the output value of neuron $m$ of $i^{th}$ hidden layer while $w_{mk}^{(i)}$ and $b_k^{(i)}$ are weight and biases respectively. The goal of hyperparameter tuning and learning is to minimize the absolute value of the difference between the approximation from the neural network and the given data. There are various types of activation functions such as tangent hyperbolic (tanh), sigmoids, and Rectified Linear Unit (ReLu).

The DNNs will be trained by using the training data-sets described in Table 3.2. The training is performed using Exponential Linear Unit (ELU) as activation function with input data being normalized as discussed in our previous work [33], the loss function considered is the Mean Absolute Percentage Error (MAPE).

Table 3.3: DNN architecture

| Layer | DNN ($c_l$) | DNN ($c_d$) | DNN ($c_m$) |
|---|---|---|---|
| Input | 1x23 | 1x23 | 1x23 |
| Hidden | 5x25, 5x50, 5x25, 5x10 | 5x25, 5x50, 5x25, 5x10 | 5x25, 5x50, 5x25, 5x10 |
| Output | 1x1 | 1x1 | 1x1 |
| MAPE | 0.39% | 0.04% | 0.47% |

The error histogram for each parameter using 5,100 shapes from the testing data set is shown in Figure 3.7.



(a) $c_l$ (b) $c_d$ (c) $c_m$

Figure 3.7: Error histograms of $c_l$, $c_d$, and $c_m$ with 5,100 shapes

## 3.5 Conclusion

In this chapter, we described the morphing airfoil controlled by 20 FFD points able to move independently in y-axis to obtain a new shape that can improve the values of the aerodynamic parameters. We also described the DNNs to obtain the values of $c_l$, $c_d$, and $c_m$ (moment coef cient around the quarter chord) of a given shape and then perform shape optimization to obtain, in seconds or in a few minutes, a safe trajectory to a given nal shape. A reasonable time of response is necessary to morph ef ciently the shape during the ight. Using directly the AD ow solver to nd the nal shape, and generate a safe trajectory to this shape, will take many minutes or even hours since it takes 20 minutes on average to obtain the aerodynamic parameters of a single shape [33].

For instance, the authors in Ref. [42] used AD ow as CFD solver to obtain $C_L$, $C_D$, $C_M$, and $C_P$ of wing shapes and trained deep neural networks using 135,108 and 47,967 CFD

samples for training and validation, respectively. They performed CFD-based optimization using the ef cient adjoint solver in AD ow, which costs 18.4 CPU hours using a 2.6 GHz processor.

The authors in Ref. [43] performed gradient-based ASO of airfoils in subsonic and transonic ow conditions to investigate the bene ts of including transition to turbulence effects into the optimization process. They performed ASO framework, which uses AD ow as the ow solver, and the optimization runs took up to 72 hours running on 36 processors Intel-Xeon-Gold-6154, 3.0GHz.

# CHAPTER 4

# MULTI-GRADIENT SEARCH ALGORITHM

With the developed surrogate models based on DNNs, in this chapter we deal with the second part of Problem 1 that is to find the shape $x$ to solve the optimization problem described in (Equation 2.2). To address this problem, we use the Multi-Gradient Search Algorithm (MGDA) optimizer to reach a new shape with reduced drag and increased lift. To find the new shape, we solve an optimization problem with three objective functions that can be mathematically described as follows:

$$
\begin{aligned}
\underset{u \in R^{20}}{\text{minimize}} \quad & F(x) = \left[ \frac{f_{c_l}(x^{r+1})}{f_{c_d}(x^{r+1})}, \; f_{c_l}(x^{r+1}), \; f_{c_d}(x^{r+1}) \right] \\
\text{subject to} \quad & x^{r+1} = x^r + u^r, \quad r = 0, \ldots, L-1 \\
& f_{c_l}(x^{r+1}) \geq f_{c_l}(x^0) \\
& f_{c_d}(x^{r+1}) \leq f_{c_d}(x^0) \\
& \| f_{c_m}(x^{r+1}) \| \leq \| f_{c_m}(x^0) \| \\
& x^0 = x \\
& x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \ldots, n \\
& f_{c_l}(x^k) \geq 0.98 f_{c_l}(x^0), \quad k = 1, 2, \ldots, L-1 \\
& f_{c_d}(x^k) \leq 1.02 f_{c_d}(x^0), \quad k = 1, 2, \ldots, L-1 \\
& | f_{c_m}(x^k) | \leq 1.05 | f_{c_m}(x^0) |, \quad k = 1, 2, \ldots, L-1.
\end{aligned}
\tag{4.1}
$$

The multi-objective optimization problem for the morphing airfoil involves three objective functions as described in (Equation 4.1) and it is solved by using the gradient vectors of $f_{c_l}(\,)$, $f_{c_d}(\,)$, and $f_{c_m}(\,)$ represented by $\nabla f_{c_l}$, $\nabla f_{c_d}$ and $\nabla f_{c_m}$ respectively; all gradient

vectors are elements of $\mathbb{R}^{20}$.

Let $\pi$ be a hyperplane defined as $\pi = \{v \in \mathbb{R}^n \mid \nabla f_{c_m}^T \cdot v = 0\}$. The orthogonal projections of $\nabla f_{c_l}$ and $-\nabla f_{c_d}$ on the hyperplane can be expressed as

$$\nabla f_{c_l} \quad \text{Proj}_{\pi}(\nabla f_{c_l}) \quad \nabla f_{c_l} - \frac{\nabla f_{c_m}^T \cdot \nabla f_{c_l}}{\|\nabla f_{c_m}\|_2} (\nabla f_{c_m})$$

$$-\nabla f_{c_d} \quad \text{Proj}_{\pi}(-\nabla f_{c_d}) \quad -\nabla f_{c_d} - \frac{\nabla f_{c_m}^T \cdot \nabla f_{c_d}}{\|\nabla f_{c_m}\|_2} (\nabla f_{c_m})$$

The gradient vectors are orthogonally projected on the hyperplane $\pi$ and therefore $\nabla f_{c_m}^T \cdot \nabla f_{c_l} = 0$ and $\nabla f_{c_m}^T \cdot (-\nabla f_{c_d}) = 0$. Using the hyperplane $\pi$, we search for shapes that improve the value of $c_l$ and $c_d$ with small variations of $c_m$. Fig. Figure 4.1 illustrates the hyperplane $\pi$ and the gradient vectors as vectors of $\mathbb{R}^3$ without loss of generality since only the norms of the vectors and the angle between them do matter.

Figure 4.1: Hyperplane and gradient vectors.

Figure 4.2 shows a trajectory that, regardless the value of $c_{fixed}$, moves toward the maximum value of $f_{c_l}$ while keeping a minimum of $f_{c_m}$ variation.

(a) Evolution of state $\mathbf{x}$ and final airfoil shape.

(b) Evolution of $c_l$, $c_d$, and $c_m$.

Figure 4.2: Example of a trajectory built by using gradient vectors.

## 4.1 Morphing Direction

The morphing direction $\mathbf{r}F(\mathbf{x})$ used is,

$$
\begin{aligned}
\mathbf{r}F(\mathbf{x}) &= w_1 \mathbf{r}f_{c_l}(\mathbf{x}) - w_2 \mathbf{r}f_{c_d}(\mathbf{x}) + w_3 \mathbf{r}f_{c_m}(\mathbf{x}) \\
&\text{subject to:} \\
&w_1 + w_2 = 1 \\
&w_2 \geq 0.1 \\
&w_3 \in \{-0.01, 0, 0.01\} \\
&\|\mathbf{r}f_{c_l}\|_2 \quad \|\mathbf{r}f_{c_d}\|_2 \quad \|\mathbf{r}f_{c_m}\|_2 = 1,
\end{aligned}
\tag{4.2}
$$

where $w_1 \in \mathbb{R}$ and $w_2 \in \mathbb{R}$ are the weights of the gradient vectors used to balance the importance of $f_{c_l}(\mathbf{p},\mathbf{q})$ and $f_{c_d}(\mathbf{p},\mathbf{q})$ in (Equation 4.1) for every iteration when running the search algorithm. For the search algorithm, we do not directly use the gradient of the ratio $\frac{f_{c_l}(\mathbf{p},\mathbf{q})}{f_{c_d}(\mathbf{p},\mathbf{q})}$ as we evaluate each term separately. The correction term $w_3 \mathbf{r}f_{c_m}(\mathbf{x})$ is added to ensure that the constraint $\|f_{c_m}(\mathbf{x}+\mathbf{r}L\mathbf{s})\| \leq \|f_{c_m}(\mathbf{x}+\mathbf{r}0\mathbf{s})\|$ is fulfilled. The partial derivatives used to obtain the gradient vectors $\mathbf{r}f_{c_l}(\mathbf{q})$, $\mathbf{r}f_{c_d}(\mathbf{q})$, and $\mathbf{r}f_{c_m}(\mathbf{q})$ can be calculated using central

difference approximations or automatic differentiation. In this work, we use the central difference approximations to obtain the gradient vectors.

## 4.2 Search Algorithm

For every iteration of the search algorithm the new shape is given by,

$$x[k+1] = x[k] - \alpha \nabla F(x[k]) \quad \alpha \in \{0, \ldots, L-1\} \tag{4.3}$$

where $\alpha$ is defined such that the maximum variation from $x[k]$ to $x[k+1]$ is no greater than 0.1, i.e., $\|x[k+1] - x[k]\|_\infty \leq 0.1$. The values of $\alpha_1, \alpha_2 \in \{0.1, 0.9\}$, $\alpha_3 \in \{-0.01, 0, 0.01\}$, and values of $\alpha$ are chosen such that $x[k+1]$ minimizes the objective function defined in (Equation 4.1). The MGDA can be described by the following pseudo-code.

---

Algorithm 1: Multi-Gradient Search Algorithm

---

Input: DNNs: $f_{cl}(\cdot), f_{cd}(\cdot), f_{cm}(\cdot)$, flight condition: (Re, Mach, $\alpha$) and initial shape $X_0$.

Output: $x^*$, shape with improved $\frac{c_l}{c_d}$, $c_l$, and $c_d$ values.

1: procedure

2: Set $K$, $A_1 = \{0.1, 0.15, \ldots, 0.85, 0.9\}$, $B_1 = \{-0.01, -0.02, \ldots, -0.08, -0.09\}$

3: Set $A_3 = \{0.01, 0.0, -0.01\}$

4: Set target_cl, target_cd, target_ratio_cl_cd, MaxQty

5: $k \leftarrow 0$; $x \leftarrow x_0$

6: stop_search=0, stop_local_search=0

7: while $(k < K$ and $($stop_search $\neq 0))$ do

8: counter $\leftarrow 0$; $x_{list} = [0]$

9: Obtain $[f_{cl}(x), f_{cl}(x), f_{cd}(x), f_{cd}(x), f_{cm}(x)]$

10: for each $\alpha_1 \in B$ do

11: for each $_1 \in A_1$ do

12: for each $_3 \in A_3$ do

13: $_2$ 1 $_1$ ; $\frac{1}{kr\ F(x)k_8}$

14: $r\ F(x)$ $_1 r\ f_{c_l}(x)$ $_2 r\ f_{c_d}(x)$ $_3 r\ f_{c_m}(x)$

15: $x[k+1] = x[k] + r\ F(x[k])$

16: if $\frac{f_{c_l}(x[k+1])}{f_{c_d}(x[k+1])} \geq \frac{f_{c_l}(x[k])}{f_{c_d}(x[k])}$ and $(f_{c_l}(x[k+1]) \geq 0.98 f_{c_l}(x[0]))$ and

17: $(f_{c_d}(x[k+1]) \leq 1.02 f_{c_d}(x[0]))$ and $(\{f_{c_m}(x[k+1]) - f_{c_m}(x[k])\} \leq 1.05\{f_{c_m}(x[k]) - f_{c_m}(x[0])\})$ then

18: $x_{list}[counter] = x[k+1]$

19: counter = counter + 1

20: end if

21: if counter = MaxQty then

22: break

23: end if

24: if $\frac{f_{c_l}(x[k+1])}{f_{c_d}(x[k+1])} \geq$ target_ratio_cl_cd and $(f_{c_l}(x[k+1]) \geq$ target_cl) and

25: $(f_{c_d}(x[k+1]) \leq$ target_cd) then

26: stop_search = 1

27: break

28: end if

29: end for

30: end for

31: end for

32: Evaluate and select cost function: Max$\frac{c_l}{c_d}$ , Max($c_l$) or Min($c_d$).

33: Obtain the best shape $x_{temp}$.

34: $x[k+1] = x_{temp}$; $k = k+1$

35: end while

36: Evaluate the trajectory and obtain $x$.

37: end procedure

---

The subsets $A_1$ and $B_1$ can be chosen randomly, as shown in line 2, such that for every run of the search algorithm, the algorithm provides a different shape $x$ that improves the aerodynamic parameters. Moreover, to avoid local minima, we perform an assessment of the cost functions in every iteration as shown in line 32 of the pseudo-code. It is important to mention that we aim to find shapes, with improved parameters, that are reachable using a trajectory that does not affect the safety conditions as described in (Equation 2.3). Figure 4.3 shows the flow chart of the multi-gradient search algorithm.

Figure 4.3: Flow chart: multi-gradient search algorithm.

To evaluate the ef ciency of the search algorithm using the surrogate model, we generate three trajectories, each trajectory with a focus on an objective function, line 32 of the pseudo-code. For the three trajectories, we reach a  nal shape with signi cant improvement on $\frac{c_l}{c_d}$ , $c_l$, and $c_d$ as described in Table 4.1.

Table 4.1: Analysis of the Three Trajectories

| | Main function | Max $\frac{c_l}{c_d}$ | Max $c_l$ | Min $c_d$ | Description |
|---|---|---|---|---|---|
| Trajectory 1 | $\frac{c_l}{c_d}$ | 23.58% at k=70 | 23.19% at k=70 | 0.34% at k=75 | Trajectory with focus to maximize $\frac{c_l}{c_d}$ |
| Trajectory 2 | $c_l$ | 23.27% at k=77 | 22.88% at k=77 | 0.33% at k=79 | Trajectory with focus to maximize $c_l$ |
| Trajectory 3 | $c_d$, $\frac{c_l}{c_d}$ | 11.29% at k=79 | 10.83% at k=79 | 0.67% at k=22 | Trajectory with focus to minimize $c_d$ and then maximize $\frac{c_l}{c_d}$ |

Figure 4.4 to Figure 4.6 present the trajectories described in Table 4.1 over three plots: plot (a) represents the evolution of the state $x_s$ and the  nal shape after 80 steps; plot (b)

depicts the evolution of $_1$; $_3$, and $_1$; plot (c) shows the evolution of $c_l$; $c_d$, and $c_m$.

(a) Evolution of $X$ and final shape. (b) ; (c) $c_l$; $c_d$; $c_m$

Figure 4.4: Search Algorithm - Trajectory 1 - Maximize $\frac{c_l}{c_d}$.

(a) Evolution of $X$ and the final shape. (b) ; (c) $c_l$; $c_d$; $c_m$

Figure 4.5: Search Algorithm - Trajectory 2 - Maximize .

(a) Evolution of $X$ and the final shape. (b) ; (c) $c_l$; $c_d$; $c_m$

Figure 4.6: Search Algorithm - Trajectory 3 - Min $c_d$ and then Max $\frac{c_l}{c_d}$.

## 4.3 Conclusion

In this chapter, we discussed the use of multi-directional search algorithm to find a shape with improved aerodynamic parameters and obtain a safe trajectory to reach this shape. The proposed search algorithm runs the the surrogate models to calculate the gradients of the functions $f_{c_l}$; $f_{c_d}$; and $f_{c_m}$ and solve the optimization problem that involves three

objective functions: Maximize $\frac{f_{c_l}}{f_{c_d}}$, Minimize $f_{c_d}$, and Maximize $f_{c_l}$. Based on $\nabla f_{c_m}$, we defined the hyperplane, a region in a local neighborhood where there is small variation on $c_m$. The search algorithm can be summarized as follows,

- Consider objective functions that are relevant in aerodynamics: increase lift-drag ratio, reduce drag and increase lift.
- Obtain a shape with improved values of $c_l$, $c_d$, and $c_m$. Ideally, this shape is on the Pareto optimum set [44, 45].
- Obtain a safe trajectory to the shape with improved values of $c_l$, $c_d$, and $c_m$.

The numerical examples show the efficiency of the search algorithm using the surrogate models. It is important to comment that we have the same final shapes shown in Figure 4.4 and Figure 4.5. However, the trajectories to reach this final shape are different and $c_d$ and $c_m$ and this fact should be considered to decide on morphing the shape.

# CHAPTER 5

# DATA-ENABLED PREDICTIVE SHAPE ALGORITHM

In this section, we deal with Problem 2. Although, the optimizer used to solve Problem 1 may provide the desired final shape, it only relies on the trained model, which does not consider real-time feedback data that may exist during the morphing period from the initial shape to the final one. To this end, the data-enabled predictive shape must be applied to solve Problem 2 and improve the system performance. The remaining of this section provides a brief mathematical background of this approach based on behavioral system theory [46, 47].

The algorithm has two components; the first is the time window $T_{ini} + N$, where $T_{ini}$ refers to the length of data collected online that will be used to predict $c_d$, and $c_m$ over the prediction horizon $N$; and the second is the matrix of trajectories, similar to the Hankel and Page matrices. Note that this matrix has as columns real trajectories with length $L$, and in this context, a trajectory is an input-output sequence provided as $\{u_k, y_k\}_{k=0}^{L-1}$. The matrix has full row rank, generated by a sufficiently rich input sequence to be used to recover all the trajectories.

Definition 1. Let $T, L \in \mathbb{Z}_{\geq 0}$ such that $T \geq mL$, then a sequence $\{u_k\}_{k=0}^{T-1}$ where $u_k \in \mathbb{R}^m$ is persistently exciting of order $L$ if the Hankel matrix

$$H_L(u) = \begin{bmatrix} u_1 & u_2 & \cdots & u_{N-L} \\ \vdots & \vdots & \ddots & \vdots \\ u_L & u_{L-1} & \cdots & u_T \end{bmatrix}$$

is full row rank, i.e., $\text{rank}(H_L(u)) = mL$. ∎

Definition 2. An input-output sequence $\{u_k, y_k\}_{k=0}^{N-1}$ is a trajectory of an LTI system $G$ if

there exists an initial condition $x \in \mathbb{R}^n$ with a state sequence $\{x_k, u_k\}_{k=0}^{N}$ such that

$$G \begin{cases} x_{k+1} = Ax_k + Bu_k, \\ y_k = Cx_k + Du_k, \\ x_0 = x, \; k \in \{0, \ldots, N\} \end{cases} \tag{5.1}$$

where $x \in \mathbb{R}^n$ are the states, $u \in \mathbb{R}^m$ are the control inputs, $y \in \mathbb{R}^p$ are the outputs, and system matrices $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$ and $D \in \mathbb{R}^{p \times m}$. Define with the tuple $(A, B, C, D)$ as the minimal realization of $G$. ∎

The following theorem is fundamental for developing a data-enabled predictive shape control [48, 49].

Theorem 1. Suppose that $\{u_k^d, y_k^d\}_{k=0}^{N-1}$ is a trajectory of an LTI system $G$ where $u^d$ is persistently exciting of order $L \geq n$. Then, $\{u_k, y_k\}_{k=1}^{L}$ is a trajectory of $G$ if and only if there exists $g \in \mathbb{R}^{N-L+1}$ such that

$$\begin{bmatrix} H_L(u^d) \\ H_L(y^d) \end{bmatrix} g = \begin{bmatrix} u_{1:L} \\ y_{1:L} \end{bmatrix}.$$

Proof. The proof follows from [48] and it is a direct application to the special case of controllable state-space systems. ∎

According to [46], Theorem 1 provides the sufficient condition to capture all system trajectories based on a known trajectory and initial state.

Define the page matrix of depth $L$ as,

$$P_L(u_T) = \begin{bmatrix} u_1 & u_{L+1} & \cdots & u_{T-L+1} \\ \vdots & \vdots & \ddots & \vdots \\ u_L & u_{2L} & \cdots & u_T \end{bmatrix},$$

which after being generated by a sufficiently rich input sequence, this matrix can be used to recover the whole space of trajectories by replacing the Hankel matrix in (Theorem 1).

The page matrix has no repeated entries, and this fact has important implications when the entries of the matrix are corrupted. Moreover, it is observed that certain robustness and optimality guarantees become tight for Page matrices [50, 51].

From the unknown system (Equation 5.1), we collect sequences of input/output data of length $L = T_{ini} + N$, where $T_{ini}$ denotes the length of the initialization sequence and $N$ denotes the length of the prediction horizon. Define the following matrices,

$$U_L := \begin{matrix} U_{T_{ini}} \\ U_N \end{matrix} = [u_L^1; u_L^2; \dots; u_L^T], \quad Y_L := \begin{matrix} Y_{T_{ini}} \\ Y_N \end{matrix} = [y_L^1; y_L^2; \dots; y_L^T]. \tag{5.2}$$

Using the High-Fidelity Model (HF-Model) based on CFD simulations, we generate trajectories of length $L = T_{ini} + N$ to obtain the input data matrix $U_L = [U_{T_{ini}}^T; U_N^T]^T$ and output data matrix $Y_L = [Y_{T_{ini}}^T; Y_N^T]^T$. For a trajectory of length $L$ to be predicted, we use the following notation $y_{ini} = [y^T[1]; \dots; y^T[T_{ini}]]^T$, $y_N = [y^T[T_{ini}+1]; \dots; y^T[T_{ini}+N]]^T$, $u_{ini} = [u^T[1]; \dots; u^T[T_{ini}]]^T$, $u_N = [u^T[T_{ini}+1]; \dots; u^T[T_{ini}+N]]^T$.

We are interested in finding $u$ for a given reference input $u_r$. Thus, for a given final shape $X_F$, we generate a reference trajectory $(u_r, y_r)$ from baseline shape $X_0$ toward $X_F$ by using the surrogate model. To clarify, denote as state $x \in \mathbb{R}^n$, input $u \in \mathbb{R}^m$, output $y \in \mathbb{R}^p$, and $U_{T_{ini}} \in \mathbb{R}^{T_{ini} m \times T}$, $U_N \in \mathbb{R}^{N m \times T}$, $Y_{T_{ini}} \in \mathbb{R}^{T_{ini} p \times T}$ and $Y_N \in \mathbb{R}^{N p \times T}$.

The prediction of $c_l$, $c_d$, and $c_m$ is given by the solution of the following optimization

problem:

$$
\begin{aligned}
\underset{g;u;y;\ y}{\text{minimize}} \quad & \sum_{k=1}^{N} p(C_{pur}(T_{ini}) \quad k_s; y_r(T_{ini}) \quad k_s q \quad W(p g; \ _y q q \\
\text{s.t.} \quad & \begin{bmatrix} U_{T_{ini}} & u_{ini} & 0 \\ U_N & u_N & 0 \\ Y_{T_{ini}} & y_{ini} & \ _y \\ Y_N & y_N & 0 \end{bmatrix} g \\
& B_L(r(T_{ini}) \quad k_s) \ \alpha \ y_r(T_{ini}) \quad k_s) \ \alpha \ B_U(r(T_{ini}) \quad k_s; \ @(k \ P \ t1; : : : ; N u \\
& u_r(T_{ini}) \quad k_s \ P U; \ @(k \ P \ t1; : : : ; N u \\
& y_r(T_{ini}) \quad k_s \ P Y; \ @(k \ P \ t1; : : : ; N u
\end{aligned}
\tag{5.3}
$$

where, $C_{pur}(s; y_r(s q \quad \| y_r(s) \quad y_r r:s) \|_Q^2 \ \| u_r(s) \quad u_r r:s) \|_R^2; R \quad w_u I_{20}, Q \quad \begin{bmatrix} w_{c_l} & 0 & 0 \\ 0 & w_{c_d} & 0 \\ 0 & 0 & w_{c_m} \end{bmatrix}$,

$w_u \ " \ w_{c_l} \ i \ w_{c_d} \ i \ 0$. $w_u \ " \ 1$, the weights $w_g; w_{c_l}$ and $w_{c_d}$ are de ned using the data set described in section section 3.3. $W(p g; \ _y q \quad w_g \| g \|_2^2 \quad w_{\ y} \| \ _y \|_2^2; w_g; w_{\ y} \ i \ 0$, as the data matrix is corrupted by the error inherent to surrogate model, the regularization parameters $w_g$ and $w_y$ are necessary. $B_U$ and $B_L$ are upper and lower constraints used to guarantee small variations on output according to Assumption 2 and $I_{20}$ is an identity matrix of appropriate dimensions. These constraints are based on online gathered data. Let $y_{min}$ $_{max}$ $r \quad _{c_l}; \quad _{c_d}; \quad _{c_m} s^T$ be the maximum absolute variation on measured $p c_l; c_d; c_m q$ for

two consecutive intermediate shapes, we have,

$$B_U[k] \leq \begin{bmatrix} w^{c_l} & 0 & 0 \\ 0 & w^{c_d} & 0 \\ 0 & 0 & w^{c_m} \end{bmatrix} \begin{bmatrix} c_l \\ c_d \\ c_m \end{bmatrix} + y[T_{ini}],$$

$$B_L[k] \geq \begin{bmatrix} w^{c_l} & 0 & 0 \\ 0 & w^{c_d} & 0 \\ 0 & 0 & w^{c_m} \end{bmatrix} \begin{bmatrix} c_l \\ c_d \\ c_m \end{bmatrix} + y[T_{ini}],$$

where $y[T_{ini}]$ is the most recent measurement of $[c_l, c_d, c_m]$ and $w^{c_l}, w^{c_d}, w^{c_m} \in [0.1]$ are correction factors whose values are equal to one as default but can be used to improve accuracy.

Remark 1. The data-enabled predictive shape algorithm uses a matrix of trajectories, randomly generated, for prediction which might be computed for every new flight condition. The size of this matrix depends on the horizon of prediction. ∎

Remark 2. We assume that the state $x[k]$ is known for all values of $k \in \{0, 1, \ldots, L-1\}$ (Assumption 1) during the entire morphing process. ∎

Remark 3. The value of u[k] is bounded such as $x[k+1] - x[k] \leq 0.1$ so that only small shape variations arise and, therefore, small variations of $[c_l, c_d]$ are produced (Assumption 2). This boundary was defined empirically by evaluating the data points used to build the DNN. ∎

Remark 4. The constraints $B_L[k], B_U[k]$ are based on online measured data and defined such as the variation of $[c_l, c_d]$ are no greater than 2% and variation of $c_m$ no greater than 5%. ∎

To show the efficiency of the data-enabled predictive shape algorithm, we run the algorithm in a trajectory with 25 steps named as "Trajectory 4." Here we consider $T_{ini} = 5$, $N = 10$, i.e., we use 5 time steps to predict 10 steps ahead. The matrix of tra-

jectories $[U_L^T; Y_L^T s^T] \in R^{345 \times 425}$, and the flight condition is defined as (Re=2e6, Mach=0.2, $\alpha$=1.0 ). The matrix of trajectories is compound by 425 trajectories, with each trajectory defined by a sequence of 15 shapes, in total we need 6,375 shapes to build the matrix $[U_L^T; Y_L^T s^T]$. The 6,375 shapes used to build the matrix is a subset of the training data set described in section section 3.3.

[Figure 5.1 to Figure 5.3] show the use of the shape algorithm on "Trajectory 4" over 3 cycles: in cycle 1, we move the first 15 steps of the given trajectory and predict other 10 steps; in cycle 2 we move the next 05 steps and predict 10 steps forward; and cycle 3, the final cycle to cover the whole trajectory with 25 steps.

(a) Evolution of state X and the final airfoil shape. (b) Evolution of $c_l$, $c_d$, and $c_m$.

Figure 5.1: Trajectory 4 - Cycle 1.

(a) Evolution of state X and the final airfoil shape. (b) Evolution of $c_l$, $c_d$, and $c_m$.

Figure 5.2: Trajectory 4 - Cycle 2

(a) Evolution of state $X$ and the final airfoil shape. (b) Evolution of $c_l$, $c_d$, and $c_m$.

Figure 5.3: Trajectory 4 - Cycle 3.

The figures above show that the prediction provided by the shape algorithm is more accurate than the one provided by the surrogate model based on DNN. At cycle 3, the maximum error on $c_d$ prediction is about 1.0 drag-count. Table 5.1 summarizes the accuracy of the proposed shape algorithm.

Table 5.1: Prediction error of $c_l$, $c_d$, and $c_m$ for Trajectory 4.

| | | Cycle 1: yr5s to yr14s | | Cycle 2: yr10s to yr19s | | Cycle 3: yr20s to yr24s | |
|---|---|---|---|---|---|---|---|
| | | Mean Abs Error | Max Abs Error | Mean Abs Error | Max Abs Error | Mean Abs Error | Max Abs Error |
| $c_l$ | DNN | 0.46% | 0.47% | 0.40% | 0.47% | 0.25% | 0.42% |
| | Shape Algorithm | 0.02% | 0.03% | 0.03% | 0.04% | 0.09% | 0.12% |
| $c_d$ | DNN | 0.49% | 0.74% | 0.69% | 0.85% | 0.86% | 1.0% |
| | Shape Algorithm | 0.01% | 0.04% | 0.03% | 0.06% | 0.01% | 0.05% |
| $c_m$ | DNN | 1.02% | 2.01% | 2.29% | 4.58% | 4.39% | 6.62% |
| | Shape Algorithm | 0.03% | 0.05% | 0.01% | 0.02% | 0.06% | 0.11% |

It is important to point out that we morph the wing in flight only when the airplane is in a stable and trimmed cruise condition. This procedure is adopted due to safety reasons since morphing the wing during the flight is a disturbance to the system, affecting the stability of the airplane. For simplification, the flight condition defined by the triple (Re, Mach), remains unchanged during the morphing process as described by Assumption 3.

## 5.1 Robustness Analysis

The matrix of trajectories in (Equation 5.3), used to predict any trajectory of the controllable and observable system defined in (Equation 2.1), is generated by using the HF-model;

this matrix might be generated for every new ight condition. As discussed in our previous work [33], real values of $c_l$, $c_d$, and $c_m$ are not always available and therefore the matrix of trajectories is built by using the developed surrogate model.

For the robustness analysis, we run the data-driven shape algorithm over twenty trajectories using the matrix of trajectories provided by the surrogate model, modeled with inherent errors on the parameters' prediction. Figure 5.4 presents the histograms of these errors for the given ight condition.

(a) $c_l$ (b) $c_d$

(c) $c_m$

Figure 5.4: Error histograms of $c_l$, $c_d$, and $c_m$ - Matrix of trajectories.

Table 5.2 presents the robustness analysis of the proposed algorithm for twenty different trajectories. For every trajectory, we consider the use of ve and ten real-time data points for prediction.

Table 5.2: Robustness analysis of the shape algorithm.

| | | Shape Algorithm w/ noise | | | DNN | | |
|---|---|---|---|---|---|---|---|
| | Data points | Max Abs Error in $c_l$ | Max Abs Error in $c_d$ | Max Abs Error in $c_m$ | Max Abs Error in $c_l$ | Max Abs Error in $c_d$ | Max Abs Error in $c_m$ |
| Traj 6 | 5 | 1.42% | 0.10 (drag count) | 0.63% | 12.94% | 0.10 (drag count) | 15.83% |
| | 10 | 0.37% | 0.10 (drag count) | 0.26% | | | |
| Traj 7 | 5 | 0.92% | 0.24 (drag count) | 4.07% | 14.83% | 0.22 (drag count) | 31.22% |
| | 10 | 0.22% | 0.11 (drag count) | 2.30% | | | |
| Traj 8 | 5 | 0.32% | 0.09 (drag count) | 1.27% | 0.72% | 0.08 (drag count) | 1.28% |
| | 10 | 0.26% | 0.03 (drag count) | 1.27% | | | |
| Traj 9 | 5 | 0.06% | 0.78 (drag count) | 0.73% | 0.83% | 1.06 (drag count) | 1.80% |
| | 10 | 0.04% | 0.31 (drag count) | 0.48% | | | |
| Traj 10 | 5 | 0.40% | 1.17 (drag count) | 0.42% | 0.32% | 2.13 (drag count) | 2.32% |
| | 10 | 0.15% | 0.54 (drag count) | 0.16% | | | |
| Traj 11 | 5 | 0.09% | 0.31 (drag count) | 1.99% | 0.47% | 0.74 (drag count) | 2.0% |
| | 10 | 0.05% | 0.17 (drag count) | 1.99% | | | |
| Traj 12 | 5 | 0.49% | 0.32 (drag count) | 8.03% | 0.58% | 1.07 (drag count) | 8.07% |
| | 10 | 0.41% | 0.23 (drag count) | 0.65% | | | |
| Traj 13 | 5 | 0.47% | 0.15 (drag count) | 3.91% | 0.94% | 1.27 (drag count) | 60.35% |
| | 10 | 0.19% | 0.08 (drag count) | 2.97% | | | |
| Traj 14 | 5 | 0.70% | 0.29 (drag count) | 10.53% | 2.83% | 1.56 (drag count) | 102.57% |
| | 10 | 0.22% | 0.39 (drag count) | 6.75% | | | |
| Traj 15 | 5 | 0.58% | 0.05 (drag count) | 3.40% | 5.49% | 2.09 (drag count) | 107.68% |
| | 10 | 0.30% | 0.05 (drag count) | 3.74% | | | |
| Traj 16 | 5 | 0.13% | 0.09 (drag count) | 1.82% | 7.04% | 0.97 (drag count) | 113.81% |
| | 10 | 0.23% | 0.14 (drag count) | 1.12% | | | |
| Traj 17 | 5 | 0.23% | 0.35 (drag count) | 0.22% | 0.65% | 0.11 (drag count) | 0.24% |
| | 10 | 0.18% | 0.11 (drag count) | 0.21% | | | |
| Traj 18 | 5 | 0.35% | 0.21 (drag count) | 1.65% | 1.19% | 0.35 (drag count) | 1.65% |
| | 10 | 0.18% | 0.18 (drag count) | 0.88% | | | |
| Traj 19 | 5 | 0.11% | 0.44 (drag count) | 1.66% | 2.81% | 0.78 (drag count) | 3.54% |
| | 10 | 0.19% | 0.24 (drag count) | 0.47% | | | |
| Traj 20 | 5 | 0.21% | 0.74 (drag count) | 2.22% | 4.56% | 0.97 (drag count) | 5.36% |
| | 10 | 0.14% | 0.66 (drag count) | 0.22% | | | |
| Traj 21 | 5 | 0.24% | 0.49 (drag count) | 0.18% | 0.28% | 0.65 (drag count) | 0.15% |
| | 10 | 0.18% | 0.35 (drag count) | 0.18% | | | |
| Traj 22 | 5 | 0.12% | 1.36 (drag count) | 0.43% | 0.15% | 0.93 (drag count) | 0.39% |
| | 10 | 0.15% | 0.66 (drag count) | 0.43% | | | |
| Traj 23 | 5 | 0.49% | 0.99 (drag count) | 0.17% | 0.49% | 3.72 (drag count) | 0.18% |
| | 10 | 0.51% | 0.22 (drag count) | 0.18% | | | |
| Traj 24 | 5 | 0.33% | 0.25 (drag count) | 0.39% | 0.34% | 0.25 (drag count) | 0.32% |
| | 10 | 0.32% | 0.28 (drag count) | 0.37% | | | |
| Traj 25 | 5 | 0.14% | 1.12 (drag count) | 0.24% | 0.19% | 1.19 (drag count) | 0.25% |
| | 10 | 0.12% | 0.53 (drag count) | 0.18% | | | |

The figures below present the evolution of $c_l$, $c_d$, and $c_m$ for every trajectory described in Table 5.2.

(a) Five real-time data points. (b) Ten real-time data points.

Figure 5.5: Robustness Analysis - Trajectory 6.

(a) Five real-time data points. (b) Ten real-time data points.

Figure 5.6: Robustness Analysis - Trajectory 7.

(a) Five real-time data points. (b) Ten real-time data points.

Figure 5.7: Robustness Analysis - Trajectory 8.