

CRANFIELD UNIVERSITY

PEDRO JOSE PATON POZO

DYNAMIC DATA DRIVEN APPLICATIONS SYSTEMS
(DDDAS) FOR MULTIDISCIPLINARY OPTIMISATION
(MDO)

SCHOOL OF AEROSPACE, TRANSPORT AND
MANUFACTURING
Thermal Power

MSc

Academic Year: 2015–2016

Supervisor: Dr Timoleon Kipouros
August 2016

CRANFIELD UNIVERSITY

SCHOOL OF AEROSPACE, TRANSPORT AND
MANUFACTURING
Thermal Power

MSc

Academic Year: 2015–2016

PEDRO JOSE PATON POZO

Dynamic Data Driven Applications Systems (DDDAS) for
Multidisciplinary Optimisation (MDO)

Supervisor: Dr Timoleon Kipouros
August 2016

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

© Cranfield University 2016. All rights reserved. No part of
this publication may be reproduced without the written
permission of the copyright owner.

Abstract

Nowadays, the majority of optimisation processes that are followed to obtain new optimum designs involve expensive simulations that are costly and time consuming. Besides, designs involving aerodynamics are usually highly constrained in terms of infeasible geometries to be avoided so that it is really important to provide the optimisers effective datum or starting points that enable them to reach feasible solutions.

This MSc Thesis aims to continue the development of an alternative design methodology applied to a 2D airfoil at a cruise flight condition by combining concepts of Dynamic Data Driven Application Systems (DDDAS) paradigm with Multiobjective Optimisation. For this purpose, a surrogate model based on experimental data has been used to run a multiobjective optimisation and the given optimum designs have been considered as starting points for a direct optimisation, saving number of evaluations in the process. Throughout this work, a technique for retrieving experimental airfoil lift and drag coefficients was conducted. Later, a new parametrisation technique using Class-Shape Transformation (CST) was implemented in order to map the considered airfoils into the design space. Then, a response surface model considering Radial Basis Functions (RBF) and Kriging approaches was constructed and the multiobjective optimisation to maximise lift and minimise drag was undertaken using stochastic algorithms, MOTSII and NSGA. Alternatively, a full direct optimisation from datum airfoil and a direct optimisation from optimum surrogate-based optimisation designs were performed with Xfoil and the results were compared.

As an outcome, the developed design methodology based on the combination of surrogate-based and direct optimisation was proved to be more effective than a single full direct optimisation to make the whole process faster by saving number of evaluations. In addition, further work guidelines are presented to show potential directions in which to expand and improve this methodology.

Keywords

DDDAS; Airfoil optimisation; surrogate model; multiobjective optimisation; data mapping; DDDOM; experimental data; CST parametrisation; Xfoil.

I hear and I forget. I see and I
remember. I do and I understand.

Confucius

Contents

Abstract	v
Table of Contents	ix
List of Figures	xiii
List of Tables	xv
List of Abbreviations	xvii
Acknowledgements	xxi
1 Introduction	1
1.1 Project Motivation	1
1.2 Project Aim and Objectives	3
1.3 Report Outline	4
2 Review of DDDAS for MDO	5
2.1 Introduction	5
2.2 Dynamic Data Driven Application Systems	5
2.2.1 DDDAS in Collaborative UAV's Swarms	9
2.2.2 DDDAS in Structural Health Monitoring	11
2.2.3 DDDAS in Materials Modelling	17
2.3 Data Driven Design Optimisation Methodology	19
2.3.1 Conv-Div Nozzle	21
2.3.2 Electronic device cooling	22
2.3.3 Submerged subsonic inlet	23
2.4 DDDOM Applied to an Airfoil	24
3 Research Methodology	27
3.1 Introduction	27
3.2 Experimental Data	28
3.3 Airfoil Mapping	32

3.3.1	Development of a CST model for 2D Airfoil Mapping	34
	Model Remarks	37
3.3.2	Effect of the order of the parametrisation	38
3.3.3	Source of Target Airfoils	40
3.4	Surrogate Model Building	42
3.4.1	Selection of Acceptable Mapped Airfoils	43
3.4.2	Radial Basis Functions (RBF) Surrogate	44
3.4.3	Kriging Surrogate	46
3.5	Multiobjective optimisation	49
3.5.1	Algorithms used	49
	Multi-Objective Tabu Search Algorithm (MOTSII)	50
	Non-Sorting Genetic Algorithm (NSGAI)	51
	Nimrod/O	52
3.5.2	Surrogate-Based Optimisation	53
3.5.3	Xfoil Direct Optimisation	55
3.6	Post-Processing	56
3.7	Outlook of the whole model	57
4	Results Analysis	59
4.1	Airfoil Mapping Accuracy	59
4.2	Selection of Airfoils included in the Surrogate	61
4.3	Surrogate Based Optimisation	64
4.3.1	Effect of running Stochastic optimisations	65
4.3.2	Kriging Surrogate	65
4.3.3	RBF Surrogate	66
4.3.4	Comparison of the two surrogate approaches	67
4.4	Xfoil Direct Optimisation	68
4.5	Validation of the methodology proposed	69
4.6	Optimum shapes and pressure distributions	73
4.7	Analysis of results with Parallel Coordinates	75
5	Conclusions and Further Work Directions	79
5.1	Conclusions	79
5.2	Further Work suggestions	82
A	Mapped Airfoils	91
A.1	Available Data Airfoils	91
A.1.1	NACA 4-Digits	91
A.1.2	NACA 5-Digits	93
A.1.3	NACA 6-Series	94
A.1.4	NACA 7-Series	99
B	Vector b - Mapping CST MATLAB function	101

C	CST Point writer MATLAB function	105
D	L2 norms of Mapped Airfoils	107
	D.1 Upper Surfaces	108
	D.2 Lower Surfaces	116
E	Data of Airfoils for Surrogate decision	123
F	RBF Surrogate Fortran Code	131
G	Kriging Surrogate MATLAB Code	139
H	Nimrodo/O input file	143
I	CST Points FORTRAN function	145
J	Matrix A	151
K	Parallel Coordinates	153

List of Figures

1.1	Whole Process	2
2.1	Schematic DDDAS diagram [29]	6
2.2	Swarm application architecture needed for swarm control [28]	10
2.3	Feedback control loop in the DDDAS framework [27] [28]	11
2.4	Example of DDDAS framework applied to composite structures	12
2.5	Multiple spatial scales included in the model.[3]	14
2.6	Control done in Structure.[4]	15
2.7	Schematic of DDDAS developed for [30].	16
2.8	Data flow model in [17]	17
2.9	Virtual shaker layout and 3D multiscale porous SMA. [14]	18
2.10	DDDAS Feedback control loop used in [13]	19
2.11	Data Driven Design Optimisation Methodology (DDDOM) [24]	21
2.12	Conv-Div Nozzle DDDOM application used in [24]	22
2.13	DDDOM model for electronic device cooling in [39]	23
2.14	Subsonic inlet model used in [40]	24
2.15	DDDOM applied to an airfoil in this project	25
3.1	Example of an airfoil available experimental data in [1]	31
3.2	Example of chart reading and the available accuracy [1]	32
3.3	Example of FFD Technique	33
3.4	CST: Example of Achievable Shapes [26]	34
3.5	Order of the CST Parametrisation	39
3.6	L_2 norm vs Order of the CST Parametrisation	40
3.7	Comparison of CST cleaning map of a dirty geometry	41
3.8	CST Parametrisation process	41
3.9	Empirical semi-variograms, γ^*	47
3.10	Non-dimensional Semi-variogram model used	48
3.11	Tabu Search Algorithm Memories[22]	51
3.12	NSGAI Procedure [11]	52
3.13	Surrogate-based optimisation	55
3.14	Whole Process	58

4.1	Example of the accuracy of the mapping vs the order of the parametrisation for upper surfaces	61
4.2	Pressure Distribution comparison after parametrisation	64
4.3	Effect of Stochastic Optimisation in the Kriging Surrogate model	65
4.4	Kriging surrogate-based optimisation	66
4.5	RBF surrogate-based optimisation	67
4.6	Pareto Optimal fronts of the surrogate-based optimisations for 5000 evaluations with MOTSII Algorithm	68
4.7	Xfoil direct optimisations	69
4.8	Optimisation steps of the methodology proposed	70
4.9	Pareto fronts of the Proposed Methodology	72
4.10	Xfoil optimal solutions	74
4.11	b_{1U} effect in Parallel Coordinates	75
4.12	b_{10L} effect in Parallel Coordinates	76
4.13	b_{9U} effect in Parallel Coordinates	77

List of Tables

2.1	List of DDDAS related workshops and conferences [12]	8
3.1	Conditions of the experimental data	43
3.2	Criteria for including an airfoil in the surrogate	44
3.3	b vector search space	53
3.4	Surrogate Optimisation settings	54
3.5	b vector of datum airfoil	54
4.1	Airfoils Included in the Surrogate	62
4.2	Rejected Airfoils	63
4.3	Benefits in the use of the method developed	73

List of Abbreviations

Abbreviations

SATM	Schools of Aerospace, Transport and Manufacturing
DDDAS	Dynamic Data Driven Application Systems
MDO	Multidisciplinary Optimisation
NSGAI	Non-dominated Sorting Genetic Algorithm II
MOTSII	Multi-Objective Tabu Search II
CST	Class Shape Transformation
FFD	Free Form Deformation
CFD	Computational Fluid Dynamics
RBF	Radial Basis Functions
RSM	Response Surface Model
ICCS	International Conference of Computer Science
DyDESS	Dynamic Data-driven Environmental Systems Science
MIT	Massachusetts Institute of Technology
NSF	National Science Foundation
AFOSR	Air Force Office of Scientific Research
SIAM	Society for Industrial and Applied Mathematics
WSC	Winter Simulation Conference
MoSES	Modeling, and Environmental Systems Workshop

CAOS	Cooperative Autonomous Observing Systems
FSI	Fluid and Structure Interaction
HPC	High-Performance Computing
SHM	Structural Health Monitoring
SMF	Surrogate Management Framework
SMA	Shape Memory Alloys
GPU	Graphics Processing Unit
MPI	Message Passing Interface
NACA	National Advisory Committee for Aeronautics
TDT	Langley two-dimensional low-turbulence pressure wind tunnel
SVD	Singular Value Decomposition
MTM	Medium Term Memory
LTM	Long Term Memory
STM	Short Term Memory
LE	Leading Edge
TE	Trailing Edge

Nomenclature

C_l	Lift Coefficient
C_d	Drag Coefficient
C'_l	Re-evaluated Lift Coefficient
C'_d	Re-evaluated Drag Coefficient
C_m	Moment Coefficient
α	Angle of Attack
L_2	Square root of the sum of deviations between mapped and original
N_i	airfoil CST Class type exponent

Δz	Trailing Edge thickness
c	Airfoil Chord
x	Airfoil x coordinate
z	Airfoil z coordinate
b_i	CST weight factor
$S(x)$	CST Shape Function
n	Bernstein Polynomials order
λ	RBF Surrogate weights
γ^*	Empirical Semivariogram
γ	Modelled Semivariogram
M	Mach number
P	Parent population
Q	Offspring population
R	P and Q combined population
F	NSGAI Best non-dominated solutions

Subscripts

0	Initial conditions
U	Upper Surface
L	Lower Surface
t	Generation t

Acknowledgements

First of all, I would like to thank and express my profound gratitude to my supervisor Dr. Timos Kipouros for his continuous support and willingness to help in every stage of this thesis, answering emails even on Sundays.

I would also like to thank my family, especially my parents for their encouragement and unfailing support in the distance and also to my aunt, who helped me in a different but necessary way. Without their help nothing of this would have been possible.

Chapter 1

Introduction

1.1 Project Motivation

The behaviour prediction of complicated systems is difficult to analyse accurately. Particularly, when real-time dynamic states take part in the process, even the most elaborated models produce simulations that do not succeed in predicting the real behaviour of the described system. If somehow the application simulations manage to incorporate new data dynamically into the system, either coming from archival or from on-line measurements of the real system, it would lead to a more accurate analysis, prediction and control of the final outcome.

These features are enhanced with the Dynamic Data Driven Application Systems (DDDAS) paradigm, which stands for a dynamic symbiosis between simulations and experiments by means of allowing the running calculations to change upon the incoming real data as well as the experimental measurements being steered by the simulation models, focusing, for instance, on a given subset of the measurement space and thus reducing costs and collection times.

In this research project, the concepts underlying the DDDAS paradigm have

been applied in combination with Multi-objective Design Optimisation (MDO) to try to counterbalance the inaccuracy of the engineering models by introducing experimental data into the design process.

Typically, in designs involving fluid dynamics, well-established optimisation design processes simulate the model iteratively until a given criteria is met, resulting in an expensive, time and resource consuming process. The work done in this thesis stands up for a different path in the optimisation process as illustrated in Figure 1.1. Therefore, if a surrogate model is built with experimental data and the optima solutions of the surrogate-based optimisation are set as a starting points for a direct CFD simulation, the total time of the design process will be decreased. In addition, the optimisation scenario will be more guided because the datum condition of the CFD optimisation will be based on real data.

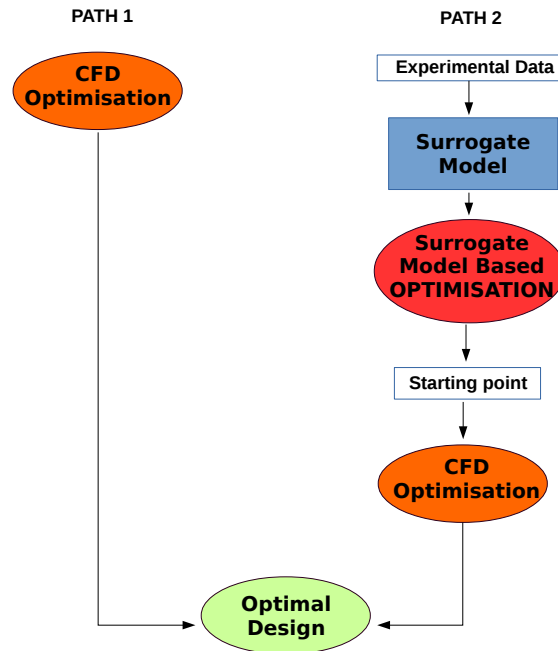


Figure 1.1: Whole Process

Here, only static experimental data has been included in a 2D airfoil design process. However, the methodology developed will be able to help in future research so as to extend the method to real-time acquisition of data with, for example, the use of the emerging rapid prototyping technologies, conforming a synergistic feedback control-loop between running simulations and wind tunnel measurements.

1.2 Project Aim and Objectives

This project, which constitutes a new iteration refinement in the work initiated by A. Agirre-Mentxaka in his MSc Thesis, aims for the development of a methodology to construct a surrogate model based on experimental data that represents the aerodynamic performance of an airfoil, so that a following surrogate-based optimisation can be performed.

Thereby, the main objectives of the project have been to:

1. Undertake a thorough research in the State of the art regarding the DDDAS paradigm and its exploitation in Aerospace Engineering.
2. Implement a new technique in order to map experimental data into the design space.
3. Construct a surrogate model based on experimental data that integrates the performance of an airfoil.
4. Define a multi-objective optimisation process standing on the surrogate model in order to find starting point designs to launch shorter direct optimisations.
5. Carry out a traditional multi-objective optimisation utilising GNU General

Public License Xfoil program so as to compare with the surrogate-based approach.

6. Establish work conclusions and further work guidelines for the continuous development of this methodology.

1.3 Report Outline

Following this introductory Chapter 1, Chapter 2 reviews the DDDAS paradigm and its latest applications to Aerospace engineering. Next, in Chapter 3, the whole research methodology process that have been followed during this thesis is explained in detail. Right after the originated results and their interpretation are presented in Chapter 4. Finally, in Chapter 5 the conclusions derived of this work and future work guidelines are given.

Chapter 2

Review of DDDAS for MDO

2.1 Introduction

This chapter mainly describes the concept of Dynamic Data Driven Applications Systems (DDDAS). Firstly, the general concept of DDDAS is introduced and examples of its potential applications in the aerospace engineering field are presented. Afterwards, a particular application of DDDAS, known as Data Driven Design Optimisation Methodology (DDDOM), is explained and also exemplified. Finally, a general description of how these methodologies apply to a 2D airfoil optimisation process, which is the scope of this work, is detailed.

2.2 Dynamic Data Driven Application Systems

The DDDAS concept, firstly introduced by F. Darema in the DDDAS Workshop [16], can be described as a “symbiotic feedback control system” that is able to utilise simulations in a dynamic manner in order to control and guide experimental measures in terms of, for example, determining where, when and how it is optimum and more

efficient to collect additional data. On the other hand, based on the experimental measurements, the applications/simulations can dynamically be steered. [29]

As [10] stated, most of the models, simulations and traditional processes used to date are serialised, unsynchronized and uncooperative: they do not capture instantaneous events and reactions of the real-world changing conditions since most of them work with static data input from traditional experimentations. In Figure 2.1, a diagram representing the DDDAS concept is shown, where the five main components of this paradigm are depicted. As [29] explains, the human part interacts both with the application models, measurement infrastructures and software support systems through dynamic computation infrastructures. These computational infrastructures encompass the computing machines and their interactions, such as monitors or computers. The application models include the algorithms and mathematical models. To the part of measurement infrastructures belong the laboratory equipment, including sensors, instruments, probes, data storage, rigs, etc. Finally, the software support is composed of the interactive visualisation and automatic steering between simulations and experiments, having the ability to select the proper algorithm and component in every moment.

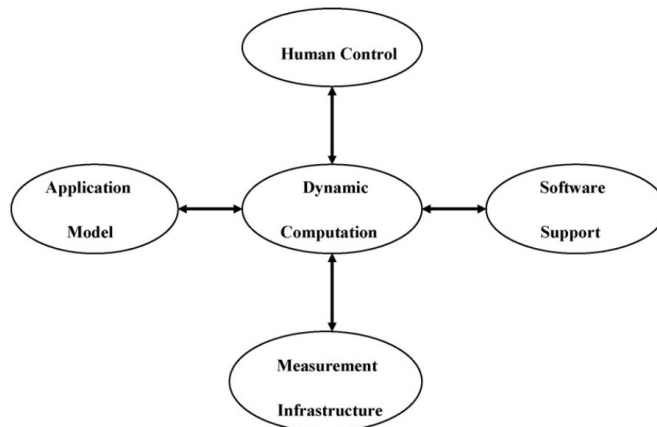


Figure 2.1: Schematic DDDAS diagram [29]

As described in NFS Workshop [16], DDDAS challenges to address four main areas:

- **Applications** that accept data at execution time and allow this data to dynamically steer them. For this, the applications models are required to describe the system at different levels of detail by dynamically selecting the models to use depending on the input data. A good understanding in how data is passed hierarchically is also necessary.
- **Mathematical Algorithms** that are stable and possess convergence robustness under the introduction of dynamic data. They also need to ensure the control of the propagation of measurement errors and uncertainty, specially when data is taken at different discretisation schemes (temporal or spatial scales), is incomplete or is just a small sample.
- **Systems Software** that are able to embody algorithms dependant on the streamed data and resources since DDDAS would employ heterogeneous platforms environments such as embedded sensor for the data acquisition and distributed simulations or specific programs for the pre and post processing of data.
- **Measurements**, in terms of developing interfaces for physical devices (i.e. the sensors used) that take part into the computational grid.

Bearing in mind that DDDAS is quite a new concept, almost all the projects and research are still in development. However, since it was introduced, every year different workshops take place such as the one in the International Conference of Computer Science (ICCS), where different case studies and research directions are exposed. Thus, the presence of DDDAS in conferences, forums and workshops has

increased in the past five years, being short-listed the most important ones in the Table 2.1.

Table 2.1: List of DDDAS related workshops and conferences [12]

MoSES IV	2017	Workshop	Manaus
AFOSR	2016	Workshop	Hartford
NSF PI	2016	Meeting	Washington
ICCS	2016	Workshop	San Diego
ACM SIGSIM PADS	2016	Workshop	Alberta
SIAM	2016	Minisymposium	Boston
ICCS	2015	Workshop	Reykjavik
DyDESS	2014	Conference	MIT
NSF PI	2014	Meeting	New York
ICCS	2014	Workshop	Cairns
WSC	2006	Workshop	Savannah
AFOSR	2013	Workshop	Arlington
ICCS	2013	Workshop	Barcelona
ICCS	2012	Workshop	Omaha
MoSES III	2011	Workshop	Petropolis
ICCS	2011	Workshop	Singapore
AFOSR-NSF	2010	Workshop	Arlington
ICCS	2010	Workshop	Amsterdam
ICCS	2009	Workshop	Baton Rouge
MoSES II	2008	Workshop	Petropolis
ICCS	2008	Workshop	Krakow
ICCS	2007	Workshop	Beijing
MoSES I	2008	Workshop	Petropolis
ICCS	2006	Workshop	Reading
ICCS	2005	Workshop	Atlanta
WSC	2006	Workshop	Arlington
NSF	2006	Workshop	Arlington
NSF	2000	Workshop	Arlington

The application of DDDAS is wide. It can be applied on fields like transport, biology, geology, social/behaviour sciences, manufacturing processes, hazard prevention, business predictions, system software, atmospheric events and engineering design optimisation among others. It is in the latter case, with particularly focus

on aerospace applications, in which the following subsections will be centred since the scope of this thesis is that of engineering nature. Nevertheless, extensive related work and research in the other areas can be found in [12] for every DDDAS event listed in the Table 2.1. Moreover, in [2], examples of the application of DDDAS such as in wind turbine fault diagnosis, wildlife modelling or volcanic ash hazard are described.

2.2.1 DDDAS in Collaborative UAV's Swarms

A challenging application for the DDDAS paradigm is to UAVs (Unmanned Aerial Vehicles) technology for all types of missions including surveillance, reconnaissance, search-rescue missions, sensing for weather prediction or deploying of materials, amongst others. Typically, UAVs will work in big swarms covering large area terrains so the need of near real-time dynamic control, command, re-tasking and efficient mission planning is highly important. Thus, for all these potential applications, the UAV swarm main task is to uninterruptedly send sensor data to central locations, responding to the possible detection of occurrences by means of adapting their sensor activities. In addition, they may have to coordinate and self-organise with all other UAVs that can be of very different sizes and capabilities in order to, for example, collect video or data images from different angles.

An example of an ongoing DDDAS application in this field is developed by G. Madey et al in [27],[37] and [28]. In their recent work, they present two application designs in order to incorporate the DDDAS for swarm control. Firstly, they created an application architecture pretending to be a real UAVs swarm that reports its performance to a central control as a single aggregated statistic, thus enabling the application to be optimized through simulation (Figure 2.2).

Secondly, they presented a DDDAS swarm control framework that improves the previous swarm application via simulations using real-time data. These simulations can either be initialized with real-time data from the application or via swarm control parameters coming from simulated agents. Afterwards, the results of the simulations, in terms of a swarm performance parameter, are examined for a central controller that determines the best way of controlling the swarm in the given conditions, as shown in the Figure 2.3, where the continuous feedback loop is represented.

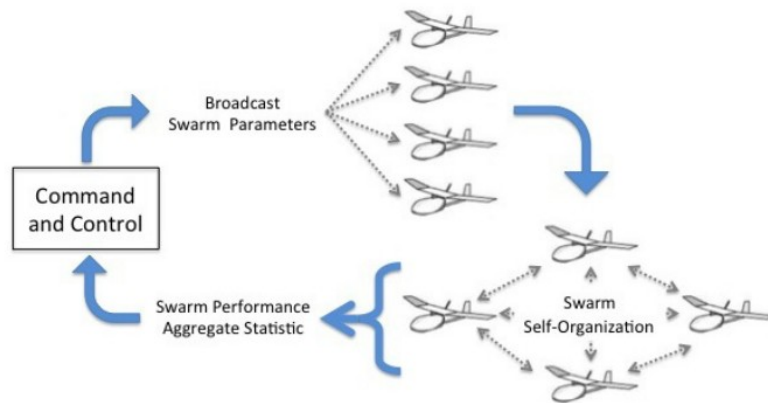


Figure 2.2: Swarm application architecture needed for swarm control [28]

Another important research path in this field is being developed at MIT supported partially by AFOSR DDDAS Program, Lincoln Laboratory, MISTI and NSF [9]. The project, known as CAOS (Cooperative Autonomous Observing Systems), develops DDDAS for cooperative UAVs aiming at atmospheric applications (climate, ecology, volcanic emissions...) using small UAVs that are able to cover terrain scales up to 100 square kilometres in few hours at a small cost.

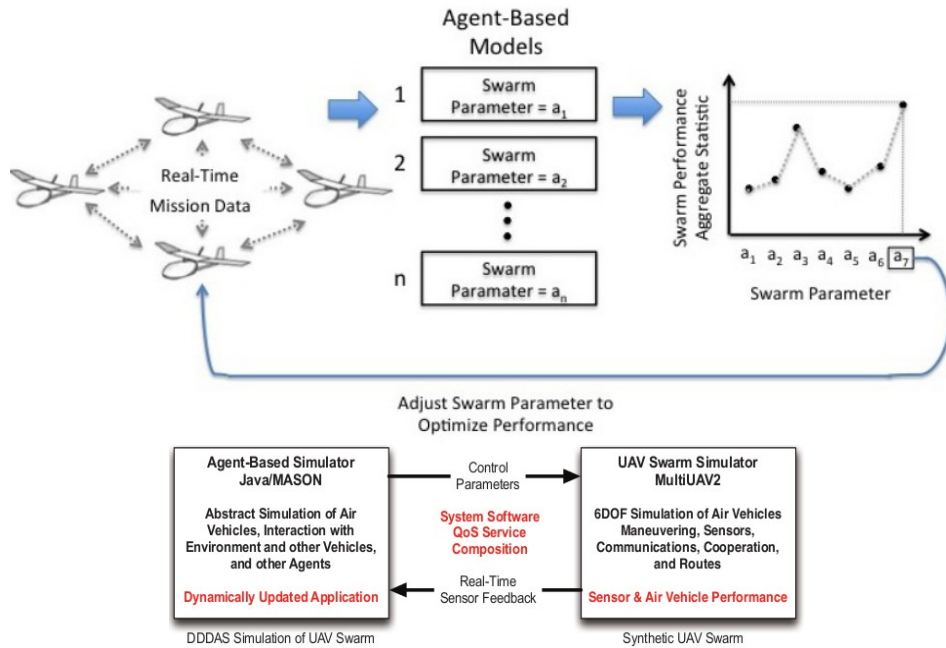


Figure 2.3: Feedback control loop in the DDDAS framework [27] [28]

Related to the self-aware UAVs, the work undertaken by Willcox et al reported in [38] has to be highlighted. The main objective of the research was to construct a Multiscale DDDAS framework that included a parametric model of an Orion UAV. That model was composed of surrogate models used to evaluate the effect on the structure damage propagation for real-time decisioning in UAVs, which links with the following section in terms of applying DDDAS for structural health monitoring.

2.2.2 DDDAS in Structural Health Monitoring

Nowadays the use of composite materials in the design of aircraft is increasing thanks to the improvements in durability and decrease in weight that they bring. Therefore, in order to reduce maintenance and operation costs, it is advantageous to have a DDDAS framework that can estimate the initiation and progression of structural damage in complex aerospace composite structures under different operation condi-

tions, depending on the application. In this DDDAS framework, embedded sensors arrays into the structure carry out continuous measurement of data used to dynamically update the computational model that describes the system, producing higher fidelity results in variables that probably are not ready or impossible to measure (Figure 2.4).

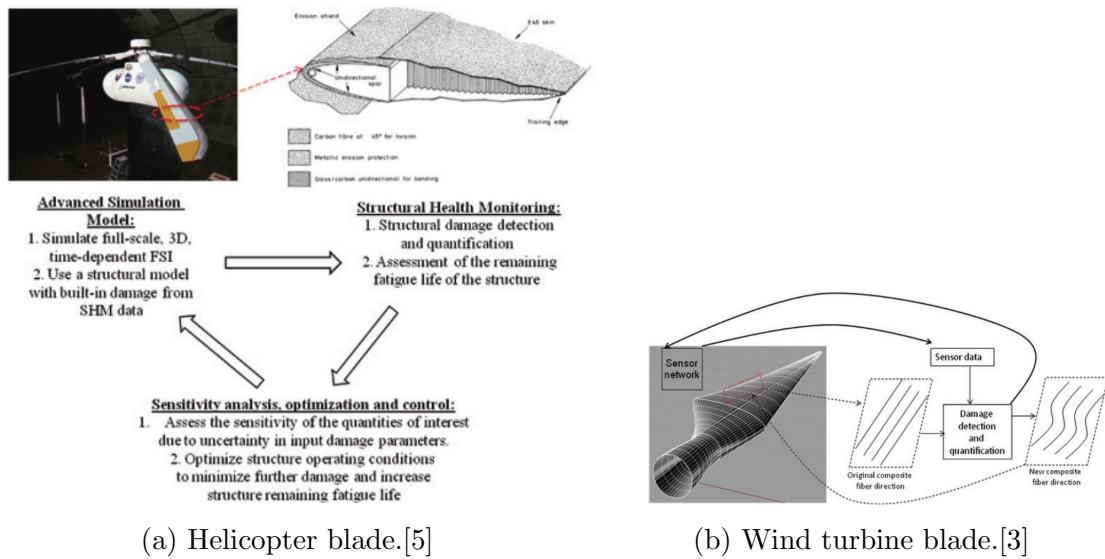


Figure 2.4: Example of DDDAS framework applied to composite structures

An example of a DDDAS framework for composite damage analysis can be found in [5], and it encompasses four main elements:

- **Structural Health Monitoring (SHM) system**, composed of the real manufactured blade with structural defects, ultrasonic sensor arrays and infrared thermographic imaging to enable defect detection, hydraulic actuator to load the blade under fatigue and strain gauges and accelerometers to determine the response of the piece under testing. Also embedded fibre Bragg sensors and distributed carbon nano tubes (CNT) are used [30].
- **Computational Model** that includes non-linear behaviour of the material

(composite structures), aerodynamics and Fluid-Structure Interacion (FSI) coupling. This model allows complex geometries and is time-dependent. It uses thin-shell Isogeometric Analysis (IGA) combined with continuum damage modelling (CDM) and CAD, being the fluid mechanics simulation based on standard FEM.

- **Sensitivity analysis, Optimiser and Control** to ensure that the structure is operating under safe conditions in order to minimise the growth of the damage. For this, the Fluid-Structure Interacion (FSI) coupling is included to achieve a succesful dynamic control.
- **High Performance Computer (HPC)**. The DDDAS framework is composed of different modules implemented in a HPC environment. However, in order to achieve near real-time performance, multicore, GPUs and other accelerator architectures are included. For the case of the FSI code parallel coupling, Message Passing Interface (MPI) library is used.

The origination of the damage in composites brings the necessity of taking into account different spatial scales, as shown in Figure 2.5. Therefore, [3] proposes to apply DDDAS to all the scales involved in the potential damage. At microscale level, Representative Volume Element (RVE) simulations along with X-ray digital micro-tomography are used to obtain the material properties (Elastic module and failure stresses). At mesoscale level, small-scale experiments along with simple geometry simulations are performed in order to construct the damage model, and this is done by means of a Surrogate Model optimisation that minimises the misfit of both experimental and computational results. Finally, in the macroscale, is where the DDDAS applies all its potential. In this level, the accelerometers and strain

gauge experimental data are used to adjust the forces and boundary conditions of the model.

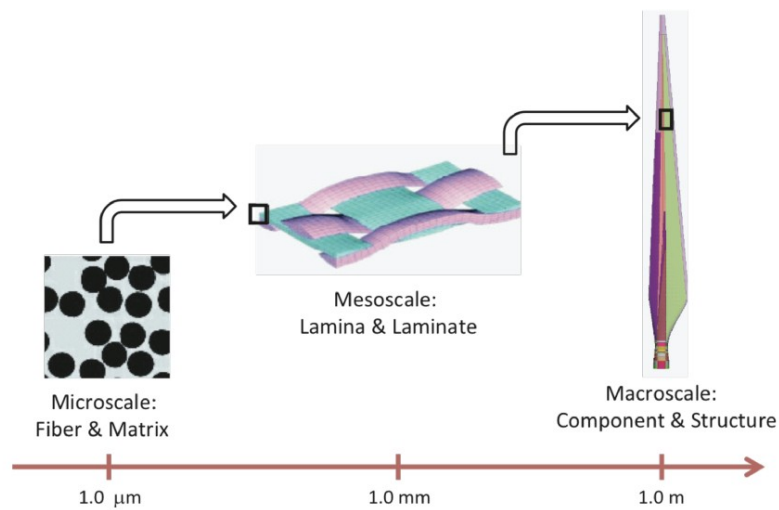


Figure 2.5: Multiple spatial scales included in the model.[3]

As [3] states, the location of the damage zones that are predicted by the computational model can be used to choose where future sensors may be placed, presenting a closed feedback loop between the real structure and the model that represents it. In addition, with the predicted response combined with the experimental data, control orders can be sent to the keep the aero-structure out of the damage zone, as shown in the Figure 2.6

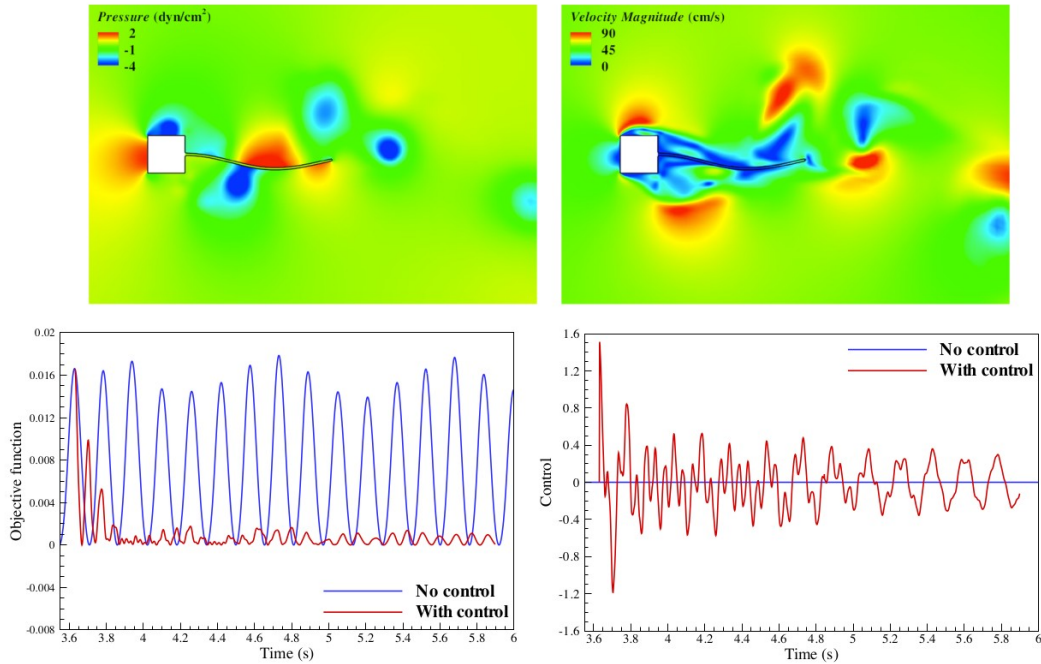


Figure 2.6: Control done in Structure.[4]

Another example of work performed in structure health monitoring can be found in [30]. They have developed a DDDAS model (depicted in 2.7) that addresses the damage evolution in a sheet of a aerospace carbon fibre epoxy composite. The matrix of this sheet is fed with CNT sensors that provide an electrically conducting network. The test specimen is loaded until failure and the strain is measured using digital image (in further studies they will include the measurement of changes in electrical resistivity). According to the model in Figure 2.7, using statistics and Bayesian analysis, the state of the material (material parameters and damage situation) is inferred and used to follow a particular set of actions such as refining the model mesh near to regions where damage is increasing, controlling the load (i.e. update the flight manoeuvre to diminish the probability of further harm), applying healing (by embedding vascular networks for infiltration of healing agents [6]) or concentrating the experimental measurement in the affected regions. It important to bear in mind

the importance of implement statistical analysis when it comes to make decisions because of the present uncertainties: the data has been only measured at some points, it has noise, the model is not perfectly representing the reality, etc. For these reasons Bayesian filtering simplified to a extended Kalman filtering is intended to be applied in [30].

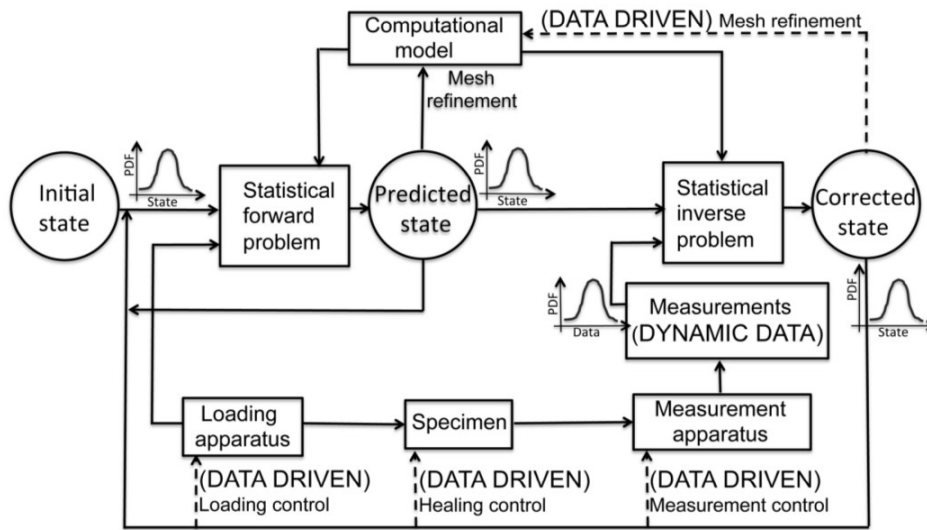


Figure 2.7: Schematic of DDDAS developed for [30].

Finally, the last reference that has been examined related to aerospace structural health monitoring is found in [17]. As sketched in Figure 2.8, they used Full Order Models (FOM) solvers that are able to construct Reduce Order Models (ROM) that are faster and easily adaptable to parameter changing in on-line or off-line modes of operations, reaching near real-time simulation conditions. In the Figure 2.8, the path 01-02-03-04 is executed off-line in order to compute and store ROMs while the path 05-06-07 and 09-08-04 are used for ROM based and sensor driven simulations to establish decision support. On the other hand, the path 09-10-05 is in charge of modelling the effects in the sensor subsystem.

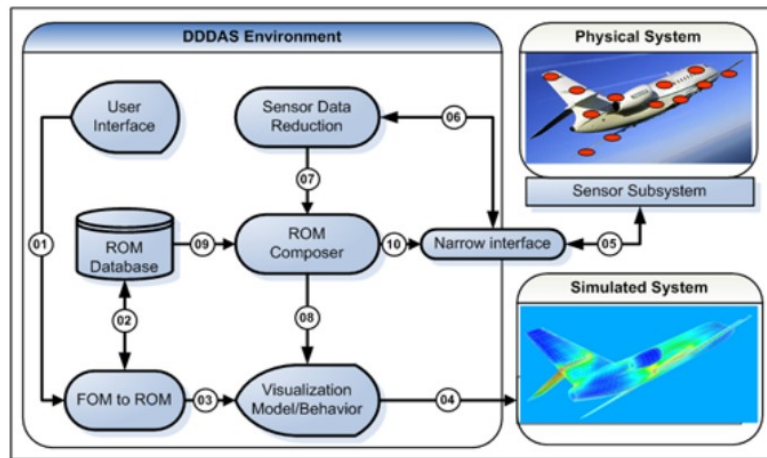


Figure 2.8: Data flow model in [17]

2.2.3 DDDAS in Materials Modelling

Another DDDAS recent interesting application that can link with DDDAS-material healing framework modules described above is in material modelling and manufacturing, with the development of porous Shape Memory Alloys (SMA) technology. This type of alloys can change their microstructure depending on the stress and temperature under they are working, which make them quite attractive for isolation and recovery purposes. They present a big recovery in elastic strain and a good damping of resonance frequencies by means of dissipating large amounts of energy. Thus, if the material has been deformed in the martensitic phase, it can recover its original shape after heating thank to the shape memory effect. Their use in aerospace is desirable because the porosity makes them lighter and, furthermore, viscous flows can be introduced through the pores in order to control the temperature and tune the vibration isolation or and mechanics properties. A research project in this field was initiated in 2012 by [14] with the objective of building a porous SMA-DDDAS framework for real life structural analysis.

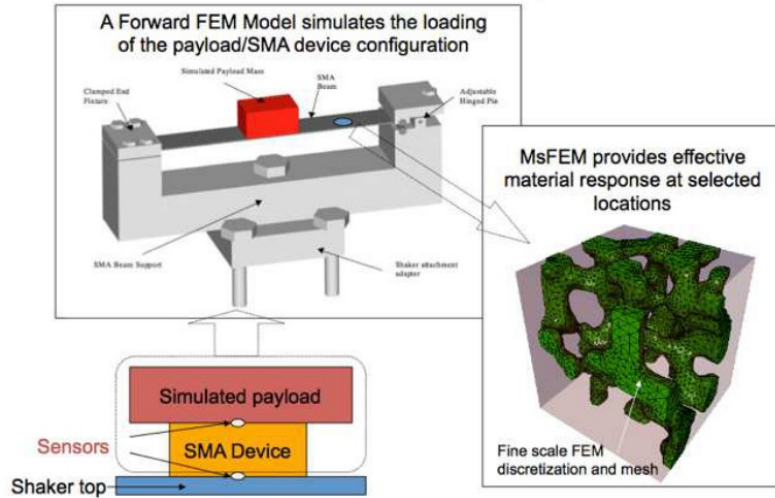


Figure 2.9: Virtual shaker layout and 3D multiscale porous SMA. [14]

The aim of the project is to monitor and change SMA in real time by being able of turning on and off the sensors and heating units, controlling the stresses on the SMA and monitoring the on-line data streams under an unmanned feedback control loop application. They have developed a multiscale porous SMA 3-D model based on thermodynamic potentials such as Gibbs free energy and a finite element model that represents a virtual beam shaker. As shown in Figure 2.9, a mass, which represents a given load, is fixed to the centre of the SMA beam. This system is vibrating periodically over a frequency range so as to find the isolation and damping characteristics that posteriorly are feeding the multi-scale finite element model. The reason of developing a virtual beam instead of using a real shaker is that the project is still in the first stages of development and the use of virtual shake device presents some advantages in terms of flexibility in the modification of the sensors and costs. Nevertheless, some data coming from a similar real shaker from Texas A& M University is being used to modify the SMA model. The future state of the particle is predicted by the DDDAS model and the SMA model is changed subsequently. In Figure 2.10, the complete DDDAS feedback control used in this project is depicted.

The research is still ongoing and the later results can be found in [13], in which further steps will be inclusion of real shaker historical data in order to calibrate the SMA-DDDAS virtual model shaker.

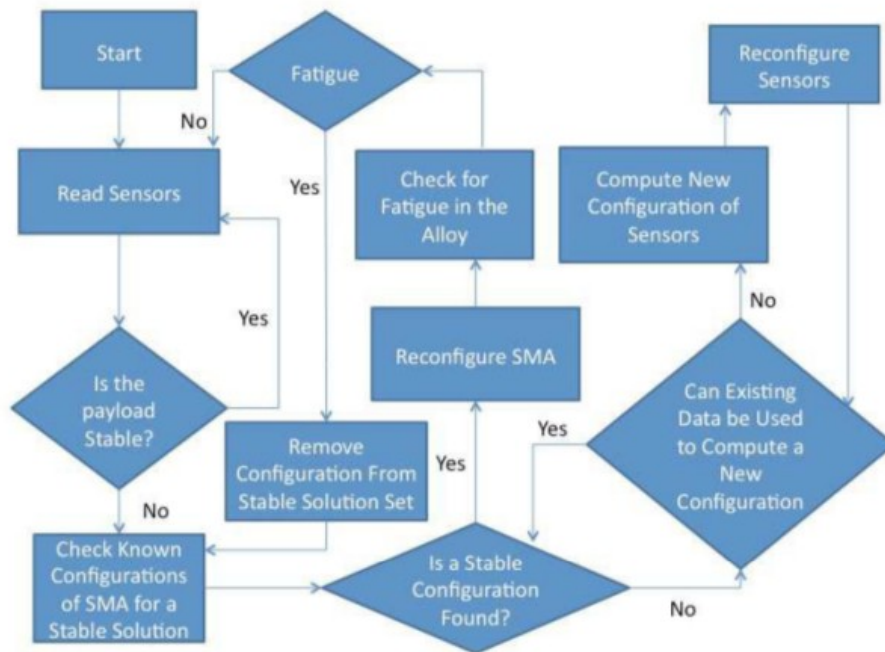


Figure 2.10: DDDAS Feedback control loop used in [13]

2.3 Data Driven Design Optimisation Methodology

Nowadays, a typically established engineering design process is composed by an initial model that is simulated under some software package (i.e. CFD, FEM, etc). If these simulations meet the engineering requirements, a posterior experiment is planned and undertaken, being the results compared with the ones coming from the previous simulations. Eventually after a given number of iterations a satisfactory design is achieved. However, following a different trend to address the design pro-

cess, Knight et al developed another methodology known as Data Driven Design Optimisation Methodology (DDDOM) [24], where DDDAS paradigm is applied to engineering design by means of synergistic incorporation of remote experiments and simulations into an automated design optimisation to achieve better solutions at a lower cost and faster. Regarding their work, DDDOM is composed of six elements:

- **Controller.** The DDDOM Controller duties are to guide and control the design optimisation process. It is written in Perl, thus enabling a robust program control locally and remotely.
- **User Interface.** Operates virtually in any operating system or platform and offers monitoring of the design optimisation process.
- **Optimiser.** Uses a Multi-Objective Design Optimisation (MDO) algorithm in order to seek into the design space and get the Pareto front.
- **Surrogate Model.** Both experiments and simulations are employed to construct the surrogate model of the objective functions. In their work, the authors based their surrogates on Response Surfaces and Radial Basis Function Artificial Neural Networks, taking into account the uncertainty of the results given by experiments and simulations.
- **Experiment.** They are undertaken in real-time.
- **Simulations.** Executed also in real-time either in local or remote stations.

A schematic of the DDDOM is shown in Figure 2.11. The following sections will describe some examples where this methodology has been used such as nozzle optimisation, electronic device cooling and subsonic inlets design.

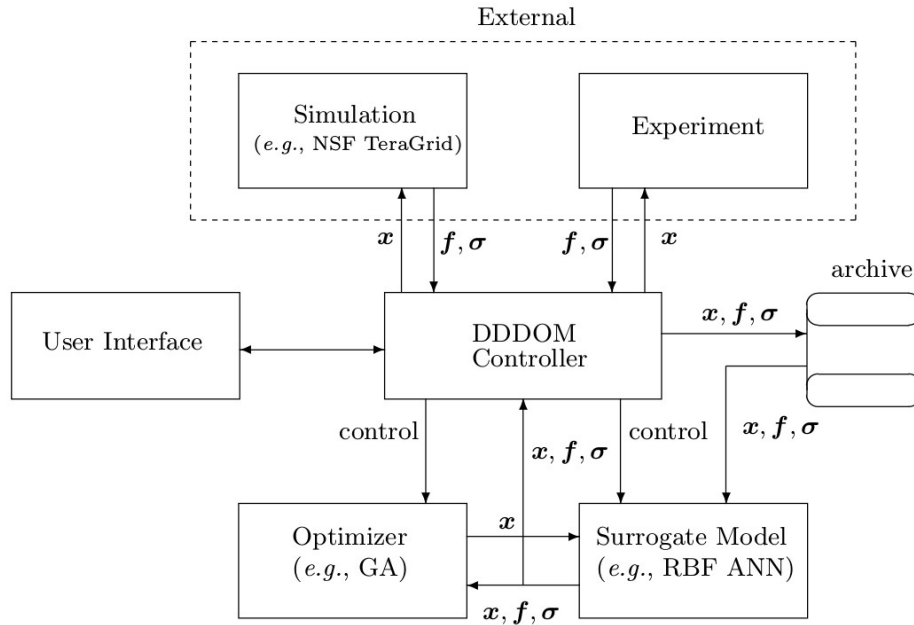
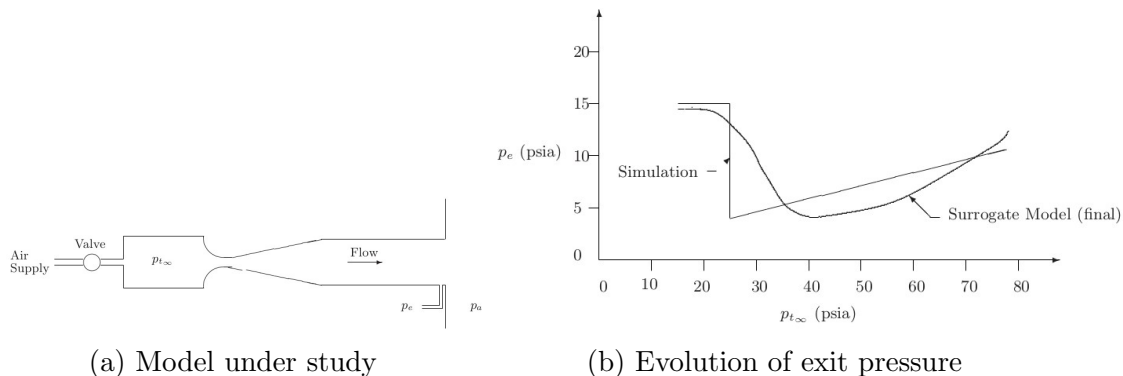


Figure 2.11: Data Driven Design Optimisation Methodology (DDDOM) [24]

2.3.1 Conv-Div Nozzle

This illustrative example was developed by Knight et al [24] in order to introduce the DDDOM. The aim of the optimisation was to find the stagnation pressure in the stagnation chamber that led to the minimum exit pressure for a given ambient pressure. (Figure 2.12 a). It was a single objective problem where the DDDOM Controller built and refined the Surrogate Model using 1D inviscid gas dynamics simulation and real-time experimental data. Firstly, the surrogate is initialised with the simulation code and used by the optimiser to find the minimum back pressure. Secondly, this latter value of pressure is chosen with a 50 % probability amongst other random values and sent to the experiment workstation that is in a separate building. Once the computer on the workstation changes the pressure to the required value, it passes to the Controller the actual value of the pressures in order to update the Surrogate Model until convergence is reached (Figure 2.12 b).



(a) Model under study

(b) Evolution of exit pressure

Figure 2.12: Conv-Div Nozzle DDDOM application used in [24]

2.3.2 Electronic device cooling

DDDOM methodology has been also used for designing cooling systems in electronic devices [23],[39]. In this case, the scope of the optimisation was minimising the pressure drop across the component and maximising the total heat flux, bearing in mind geometrical constraints. Two design variables L_1 and L_2 represented the location of the heat sources and second and third order regression models were utilised to formulate the surrogate model base on experimental and computed data. In Figure 2.13, the circles represent the number of sampled points chosen from the computed model and the squares represent the ones chosen for the experimental measurements, adequately collocated in order to balance the domain of the simulation.

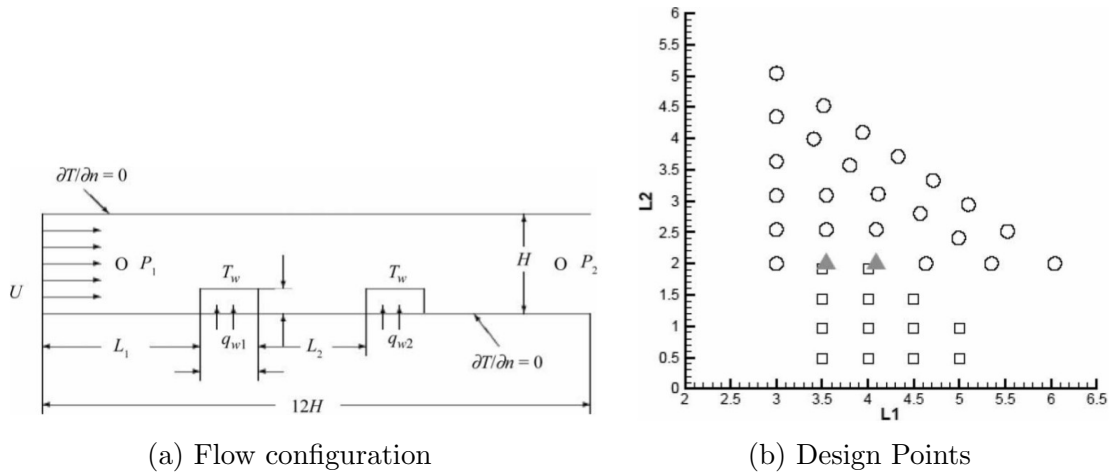
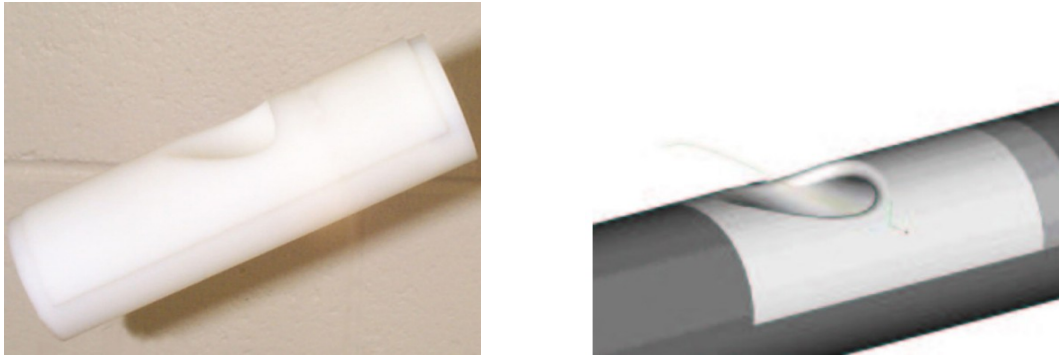


Figure 2.13: DDDOM model for electronic device cooling in [39]

2.3.3 Submerged subsonic inlet

Lastly, another application of DDDOM is for the optimisation of a fluid flow duct that delivers air aero-engines such as turbjets and turbofans [40]. The idea was to minimise the Distortion and Swirl coefficients at the engine inlet face in by changing the inlet geometry and the angle of attack. The most important aspect to highlight in this project is how experiments and simulations worked synergistically: The simulation part took the role of exploring the changes in the inlet geometry while the experimental part did it with the angle of attack. The reason for the experiments to accommodate the angle of attack relied on the fact that the angle change took matter of minutes once the Rapid Prototyping of the model was built (Figure 2.14 a). On the other hand, the simulation part was in charge of varying the geometry, thing that was easier than building a prototype in each evaluation. It can be said that each part worked in the field where its performance was more effective.



(a) Experimental (Rapid Prototyping specimen)

(b) Simulation (CAD model)

Figure 2.14: Subsonic inlet model used in [40]

2.4 DDDOM Applied to an Airfoil

In this project, fundamental concepts of DDDOM have been applied to a 2D airfoil optimisation. As stated before, complex CFD simulations are usually carried out in order to reach a specific solution while experimental testing is left for the end of the process just for validation purposes. This is because running the experiments and building the prototypes are costly steps within the design methodology if measurements results are wanted to be reliable.

In order to avoid this complex scenario of measurements, during this project, already available and reliable experimental data coming from NACA experiments has been used to build a response surface model. This surrogate is built upon the drag and lift experimental coefficients of the included airfoils for a given flight condition. It is also conformed by the design vectors (**b**) coming from a Class Shape-Transformation (CST) parametrisation of the included airfoils.

Once the surrogate is built, it is sent to an optimiser and the results are compared with the results that a direct optimisation with Xfoil will give, as sketched in Figure 2.15.

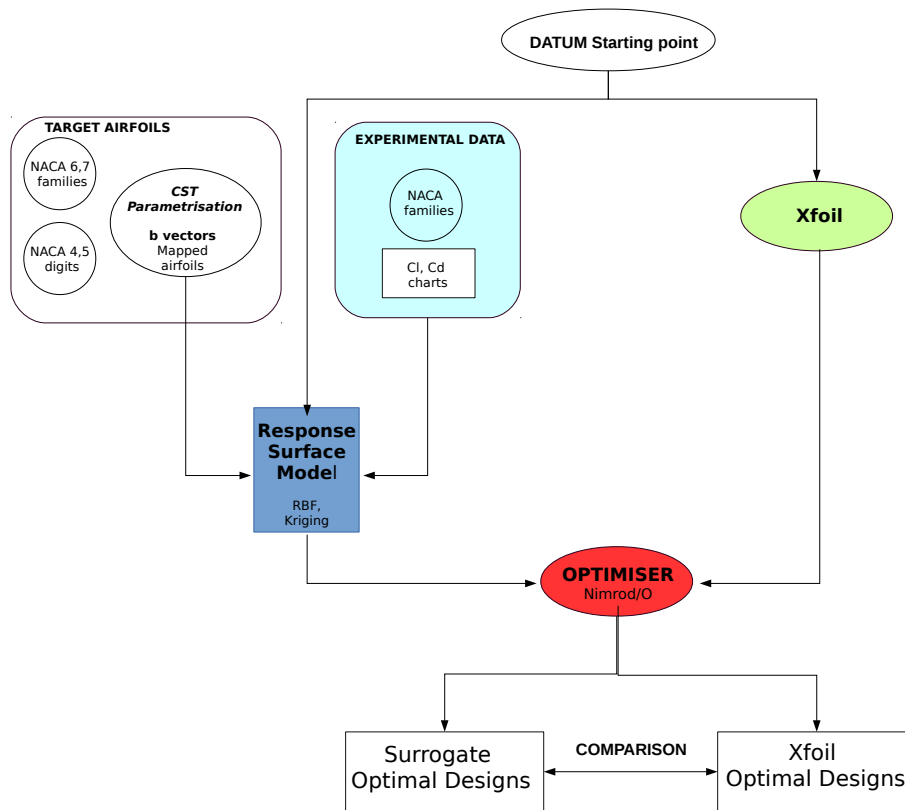


Figure 2.15: DDDOM applied to an airfoil in this project

Chapter 3

Research Methodology

3.1 Introduction

This research project is a second iteration in the work initiated by A. Agirre-Mentxaka [2] in 2015. Despite the main objective remains unchanged, the methodology followed is different. A new research approach has been followed in order to improve and complete the methodology that he started. As explained in the preceding sections, including experimental data and DDDAS concepts into the multi-objective design processes can make the model more efficient and reliable.

Throughout this chapter, the methodology followed to complete this thesis is described. In a first place, the technique used for mapping the airfoils, which represents the keystone and the most important contribution for this work, is explained thoroughly. Secondly, the different approaches considered to construct the surrogate model based on the experimental data and the mapped airfoils are described. Thirdly, the undertaken optimisation process, including an explanation of the chosen algorithms and tools, is presented. Finally, the procedures utilised for the post-analysis of the results are commented, giving a general outlook to the whole model

for closing the chapter.

3.2 Experimental Data

In the early 1930s, the airfoil design process was mainly guided by the experience of the designers with known shapes. It was the National Advisory Committee for Aeronautics (NACA) who developed a systematic manner of naming and classifying the airfoils based on the slope of the airfoil mean camber line and the thickness distribution in relation with this line. They carried out systematic tests of the different families of airfoils under different numbers of angle of attack α and Reynolds in the Langley two-dimensional low-turbulence pressure wind tunnel (TDT). Inside the wind tunnel, that had a section of 3 feet width and 7.5 feet high, real flight conditions by that time were closely approached by testing over a range of Reynolds numbers from 3 to 9 million at a Mach number close to 0.17. All airfoils were tested until stall condition was reached.

Thus, the experimental data used in this research project was obtained by NACA in those experiments, being available in [1]. Amongst others, four main families that were tested and have been included are:

- **NACA Four-digits**, characterised by good stall properties, low maximum C_l and high C_d , i.e. NACA2415, where the maximum camber is 2% (first digit), located at a 40% chord from the leading edge (second digit) and a maximum thickness of 15% (third and fourth digits), being all values set in percentage of the chord.
- **NACA Five-digits**, characterised by poor stall behaviour and high C_d , i.e. NACA23012, where the design lift coefficient in tenths is 0.3 (first digit mul-

tiplied by 3/2), the maximum camber is located at a 15% from the leading edge (second and third digits divided by 2) and a maximum thickness of 12% (fourth and fifth digits).

- **NACA 6-series**, this family of airfoils was developed using improved theoretical methods to specify the desired pressure distribution in order to achieve greater laminar flow than the previous families. Characterised by high maximum C_l , low C_d within small range of operations and poor stall behaviour, i.e. NACA64(1)212, $a=0.6$, where the first digit indicates the series family, the second digit indicates the location of the minimum pressure, the third digit, (1), indicates that drag is kept at a low levels at lift coefficients 0.1 above or below the design C_l which is 0.2, given by the fourth number. The last two digits represent again the thickness. Additionally, the part of $a=0.6$ indicated the percentage of the chord in which the pressure distribution is uniform. If this last term is not given, it is assumed to be equal to 1.
- **NACA 7-series**, this family was derived to maximise laminar flow regions distinguishing between the minimum pressure locations in the upper and lower surfaces, i.e. NACA747A315, where the first digit denotes the series, the second and third indicate the locations of minimum pressure on the upper and lower surfaces, respectively. The letter represents the thickness distribution and mean line form, the fourth digit indicates the design C_l and again, the thickness is given by the last two digits.

As can be appreciated in Figure 3.1, the accuracy in the process of reading the data from the published data is questionable due to the quality of the old report. Thorough research has been done in order to find a database that contained the data available in the charts but nothing has been found in the state of the art. The

data has had to be read directly from the charts. In a first approach and following what was done in [2], the use of a software to extract points from digitalised charts was considered. However, this solution was discarded for the following reasons:

- Most of the charts in [1] were not properly digitalised (they are crooked or even contain ink stains that disallow the identification of the points to be read).
- By using this kind of software, the chart boundary axes have to be set and properly scaled manually each time. Eventually, there is going to be an unavoidable error associated with the action of clicking with the mouse in the desired points, which, in turn, incorporate uncertainty since they are represented with big geometry figures (triangles, squares, circles) that don't show clearly where their point centre is.
- The time needed to process the total amount of charts following this procedure was disproportioned in relation to the improvements of accuracy that it could generate.

For these reasons, the use of software for reading digitalised data was discarded and a faster approach consisting of reading manually the data was used. A systematic procedure was followed bearing in mind that:

- The resolution of the charts to determine the lift coefficient given the angle of attack was 0.1. Once the C_l value was read, the reading was used in the drag coefficient charts that have a resolution of 0.001 (Figure 3.2).
- Typical values of C_l and C_d include two significant figures, which is a higher precision than it could be achieved by reading directly the charts. Nevertheless, for those readings that were clearly located in the half, quarter or third division

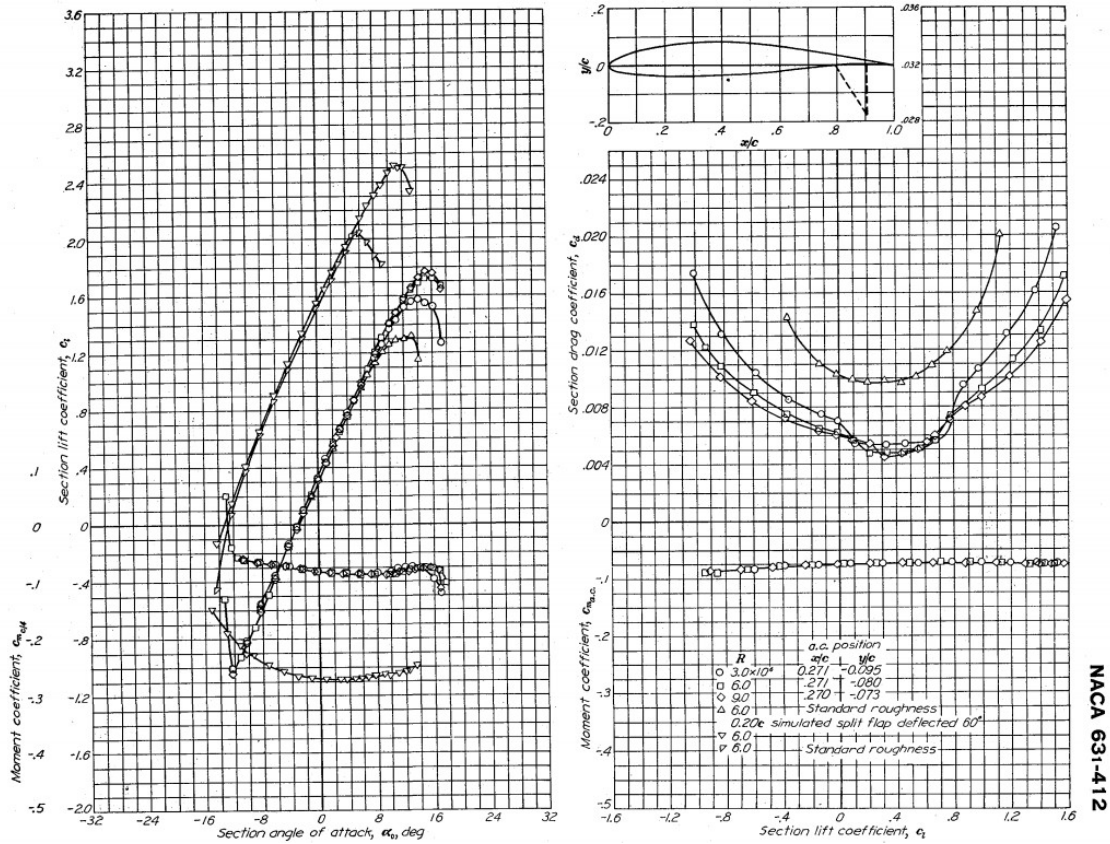


Figure 3.1: Example of an airfoil available experimental data in [1]

of the chart scale, the approximation was done to 0.05/0.025/0.033 for the C_l and 0.0005/0.00025/0.0033 for the C_d lecture.

Data belonging to a total of 83 airfoils belonging to four NACA families was read for a $\alpha = 3^\circ$ and a Reynolds number of $3 \cdot 10^6$. The lift and drag coefficients that were read can be found in Appendix A.

The experimental data doesn't clarify the exact Mach number in which each measurement took place, it's only mentioned that in all cases it was below 0.17. For this reason, it has been considered that for posterior Xfoil simulations the Mach number is 0.16 in order to establish a minimum margin against the upper limit of 0.17. Nevertheless, in Xfoil, once the Reynolds number has been set, choosing a

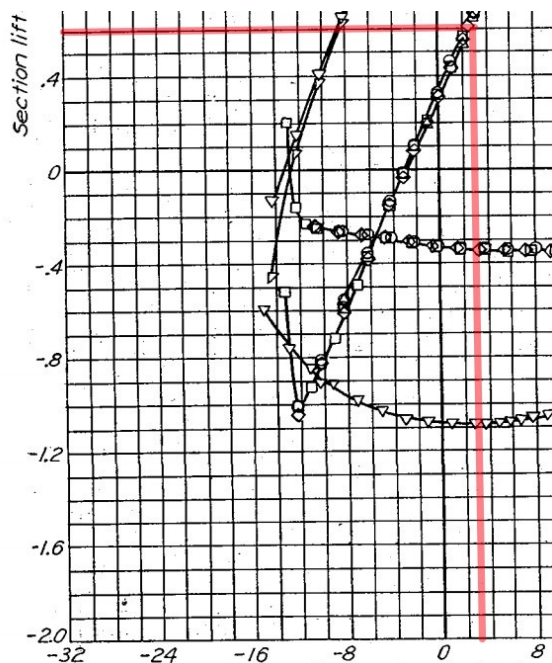


Figure 3.2: Example of chart reading and the available accuracy [1]

Mach number between 0.1 and 0.17 has a negligible effect in the final values of lift and drag coefficients.

3.3 Airfoil Mapping

One important part within any optimisation process involving the design of geometries is the parametrisation technique used for their construction. In the previous work done in [2], FFD technique was used for the parametrisation of the airfoils. This technique consists of conforming a lattice on the given geometry by setting a number of control points which are the design variables. By controlling the position of the control points, the lattice is deformed and the enclosed geometry, which is linked to these points, can be varied (Figure 3.3). However, mapping the airfoils by using this procedure often requires an optimisation process in order to match the datum airfoil geometry by means of varying its control points to the target airfoil,

minimising the difference between them.

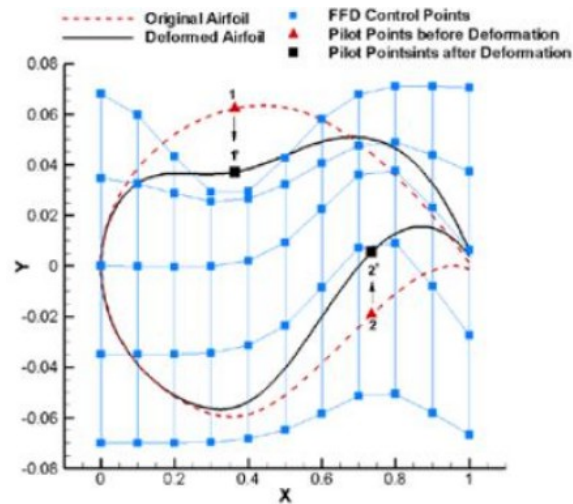


Figure 3.3: Example of FFD Technique

Thereby, a different parametrisation technique based on Class-Shape Transformations (CST) has been used in this project. This alternative method is more suitable for the characterisation of aerodynamic surfaces, because with a few set of variables, all kind of aerodynamic shapes can be constructed: wings, missiles, airfoils, nacelles, etc. (Figure 3.4). Moreover, if CST is used for mapping target geometries, any optimisation process, unlike with FFD, is required since the process is carried out by the solving a matricial system of equations. Thus, the mapping of the 83 included airfoils takes matter of seconds whereas if FFD optimisation is used, every airfoil takes an average of 30 minutes time [2]. Furthermore, this technique is more robust, numerically stable and produce smoother and more realistic geometries. Extensive development of the method can be found in [25] and [26].

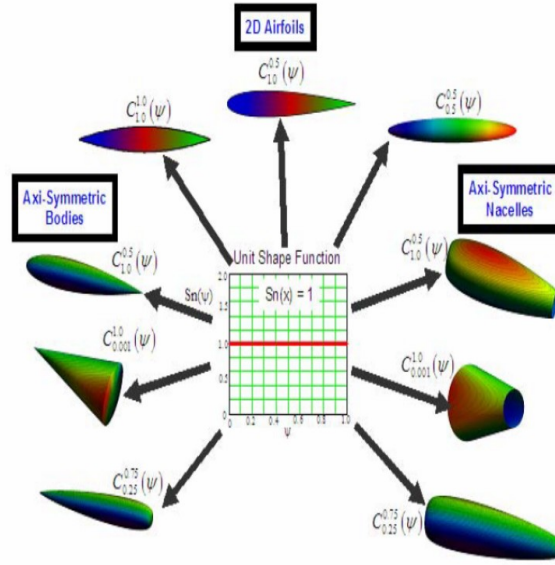


Figure 3.4: CST: Example of Achievable Shapes [26]

3.3.1 Development of a CST model for 2D Airfoil Mapping

According to the aforementioned references, a 2D airfoil geometry can be represented under CST parametrisation using the expression 3.1.

$$\frac{z}{c} \left[\frac{x}{c} \right] = \left(\frac{x}{c} \right)^{N_1} \left(1 - \frac{x}{c} \right)^{N_2} S \left[\frac{x}{c} \right] + \frac{x}{c} \frac{\Delta z}{c} \quad (3.1)$$

Where,

- The term $\left(\frac{x}{c} \right)^{N_1}$ ensures a round nose of the airfoil, by substituting $N_1 = 0.5$.
- The term $\left(1 - \frac{x}{c} \right)^{N_2}$ ensures a pointed end airfoil, by substituting $N_2 = 1$.
These first two terms compose the Class function, which can define other geometries as well, depending on the selection of N_1 and N_2 .
- The term $S \left[\frac{x}{c} \right]$ is known as the shape function, which can be decomposed into

different components determining the specific geometry shape of the airfoil. Normally, Bernstein polynomials that include some coefficients multiplying each one of the polynomials (weight factors, b_i , according to [7]) are used. These weight factors, b_i , work as the design variables to achieve the required geometry.

$$S\left[\frac{x}{c}\right] = \sum_{i=0}^n \left[b_i \cdot \frac{n!}{i!(n-i)!} \cdot \left(\frac{x}{c}\right)^i \cdot \left(1 - \frac{x}{c}\right)^{n-i} \right] \quad (3.2)$$

- The last term, $\frac{x}{c} \frac{\Delta z}{c}$ allows the inclusion of a trailing edge thickness, which can be interesting to consider if, for example, manufacturing tolerances want to be included in the mapping.

Thus, for a single j -point of the total of points that form an airfoil:

$$\frac{z_j}{c} = \left(\frac{x_j}{c}\right)^{N_1} \left(1 - \frac{x_j}{c}\right)^{N_2} \sum_{i=0}^n \left[b_i \cdot \frac{n!}{i!(n-i)!} \cdot \left(\frac{x_j}{c}\right)^i \cdot \left(1 - \frac{x_j}{c}\right)^{n-i} \right] + \frac{x_j}{c} \frac{\Delta z}{c} \quad (3.3)$$

If this is rearranged in a matricial form, it is possible to take into account all the points conforming the airfoil, bearing in mind that it has to be split in lower and upper surfaces:

$$\begin{bmatrix} z_1/c \\ z_2/c \\ \cdot \\ \cdot \\ \cdot \\ z_j/c \end{bmatrix} = \begin{bmatrix} \left(\frac{x_1}{c}\right)^{N_1} \left(\frac{x_1}{c}\right)^{N_2} \sum_{i=0}^n \left[b_i \cdot \frac{n!}{i!(n-i)!} \cdot \left(\frac{x_1}{c}\right)^i \cdot \left(1 - \frac{x_1}{c}\right)^{n-i} \right] \\ \left(\frac{x_2}{c}\right)^{N_1} \left(\frac{x_2}{c}\right)^{N_2} \sum_{i=0}^n \left[b_i \cdot \frac{n!}{i!(n-i)!} \cdot \left(\frac{x_2}{c}\right)^i \cdot \left(1 - \frac{x_2}{c}\right)^{n-i} \right] \\ \cdot \\ \cdot \\ \cdot \\ \left(\frac{x_j}{c}\right)^{N_1} \left(\frac{x_j}{c}\right)^{N_2} \sum_{i=0}^n \left[b_i \cdot \frac{n!}{i!(n-i)!} \cdot \left(\frac{x_j}{c}\right)^i \cdot \left(1 - \frac{x_j}{c}\right)^{n-i} \right] \end{bmatrix} + \begin{bmatrix} \frac{x_1}{c} \frac{\Delta z}{c} \\ \frac{x_2}{c} \frac{\Delta z}{c} \\ \cdot \\ \cdot \\ \cdot \\ \frac{x_j}{c} \frac{\Delta z}{c} \end{bmatrix} \quad (3.4)$$

And, by renaming the term in the matrix,

$$H_{ji}(x_j/c) = \left(\frac{x_j}{c}\right)^{N_1} \left(\frac{x_j}{c}\right)^{N_2} \cdot \left[\frac{n!}{i!(n-i)!} \cdot \left(\frac{x_j}{c}\right)^i \cdot \left(1 - \frac{x_j}{c}\right)^{n-i} \right] \quad (3.5)$$

We get to the expression 3.6:

$$\begin{bmatrix} z_1/c \\ z_2/c \\ \cdot \\ \cdot \\ \cdot \\ z_j/c \end{bmatrix}_{j \times 1} = \begin{bmatrix} H_{10}(x_1/c) & H_{11}(x_1/c) & \dots & H_{1n}(x_1/c) \\ H_{20}(x_2/c) & H_{21}(x_2/c) & \dots & H_{2n}(x_1/c) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ H_{j0}(x_j/c) & H_{j1}(x_j/c) & \dots & H_{jn}(x_j/c) \end{bmatrix}_{j \times n} \cdot \begin{bmatrix} b_0 \\ b_1 \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{bmatrix}_{n \times 1} + \begin{bmatrix} x_1/c \cdot \Delta z/c \\ x_2/c \cdot \Delta z/c \\ \cdot \\ \cdot \\ \cdot \\ x_j/c \cdot \Delta z/c \end{bmatrix}_{j \times 1} \quad (3.6)$$

Where :

- j is the number of points that conform the airfoil.
- n is the order of the Bernstein binomials.

Eventually, a final expression where matrices \mathcal{H} and \mathbf{dz} are known, is reached:

$$\mathbf{z} = \mathcal{H} \cdot \mathbf{b} + \mathbf{dz} \quad (3.7)$$

If a representation of a given airfoil wants to be made, the design variables that would allow doing that would be the unknown vector \mathbf{b} . If the target airfoil representation is given, which means that the target points \mathbf{z} are known, it is feasible to find \mathbf{b} by inverting \mathcal{H} matrix. If this matrix is not square, the calculation of its pseudoinverse would be needed instead:

$$\mathbf{b} = \mathcal{H}^+ \cdot (\mathbf{z} - \mathbf{dz}) \quad (3.8)$$

A MATLAB function that uses as an input a target airfoil coordinates and the order of the Bernstein Polynomials, n , has been created and is included in Appendix B. Once the design variables vector \mathbf{b} , which will $n + 1$ components, has been calculated, it is possible to build the mapped airfoil with the desired x/c points distribution by using expression 3.7. The point distribution can be whichever, but for a correct representation of an airfoil a cosine point distribution that includes more points at the leading and trailing edges is used. The MATLAB function that performs this task can be found in Appendix C.

Model Remarks

- Every set of equations is solved for the upper and lower surfaces (U and L subscripts, respectively), giving the flexibility of choosing different orders of the Bernstein polynomials and therefore a different number of design variables

\mathbf{b} for each airfoil surface:

Upper side,

$$\mathbf{z}_U = \mathcal{H}_U \cdot \mathbf{b}_U + \mathbf{d}\mathbf{z}_U \quad (3.9)$$

$$\mathbf{b}_U = \mathcal{H}_U^+ \cdot (\mathbf{z} - \mathbf{d}\mathbf{z}_U) \quad (3.10)$$

Lower side,

$$\mathbf{z}_L = \mathcal{H}_L \cdot \mathbf{b}_L + \mathbf{d}\mathbf{z}_L \quad (3.11)$$

$$\mathbf{b}_L = \mathcal{H}_L^+ \cdot (\mathbf{z} - \mathbf{d}\mathbf{z}_L) \quad (3.12)$$

- According to [7], there's no unique CST parameterisation of the airfoil. This uniqueness is given by the spectral condition number of \mathcal{H} matrix. This means that very different \mathbf{b} can lead to similar geometries and vice versa. The ill-condition of the matrix worsens with the increase of the order of the parametrisation, namely, with the increase of the order of the Bernstein polynomials. This is why a compromise solution between a minimum value Bernstein order n to map the target airfoil with low error and a maximum value in order to avoid an ill-condition of the matrix has to be found.

3.3.2 Effect of the order of the parametrisation

As has been explained before, the number of Bernstein polynomials that are included in the parametrisation has an important effect in the accuracy of the mapping. As shown in the Figure 3.5, choosing a low order of Bernstein polynomials leads to a poor mapping (i.e 1 parameter, blue line). On the other hand, if a high order

is chosen, \mathcal{H} matrix gets ill-conditioned and an inaccurate behaviour in terms of over-fitting close to the trailing edge appears (i.e. 15 parameters, brown line).

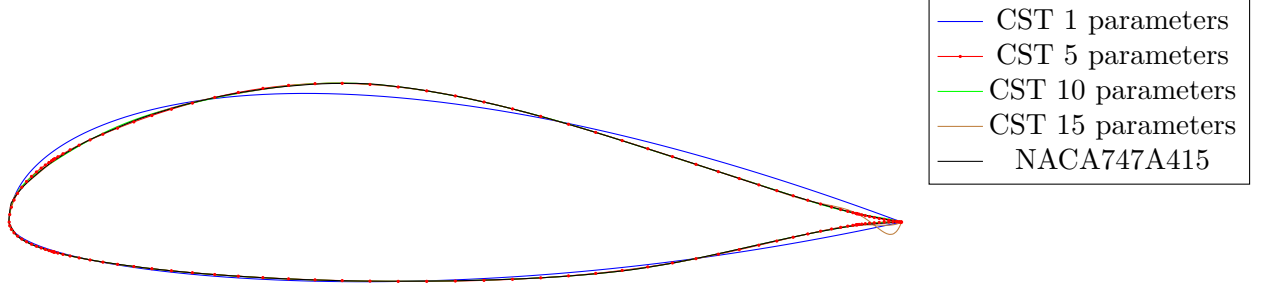
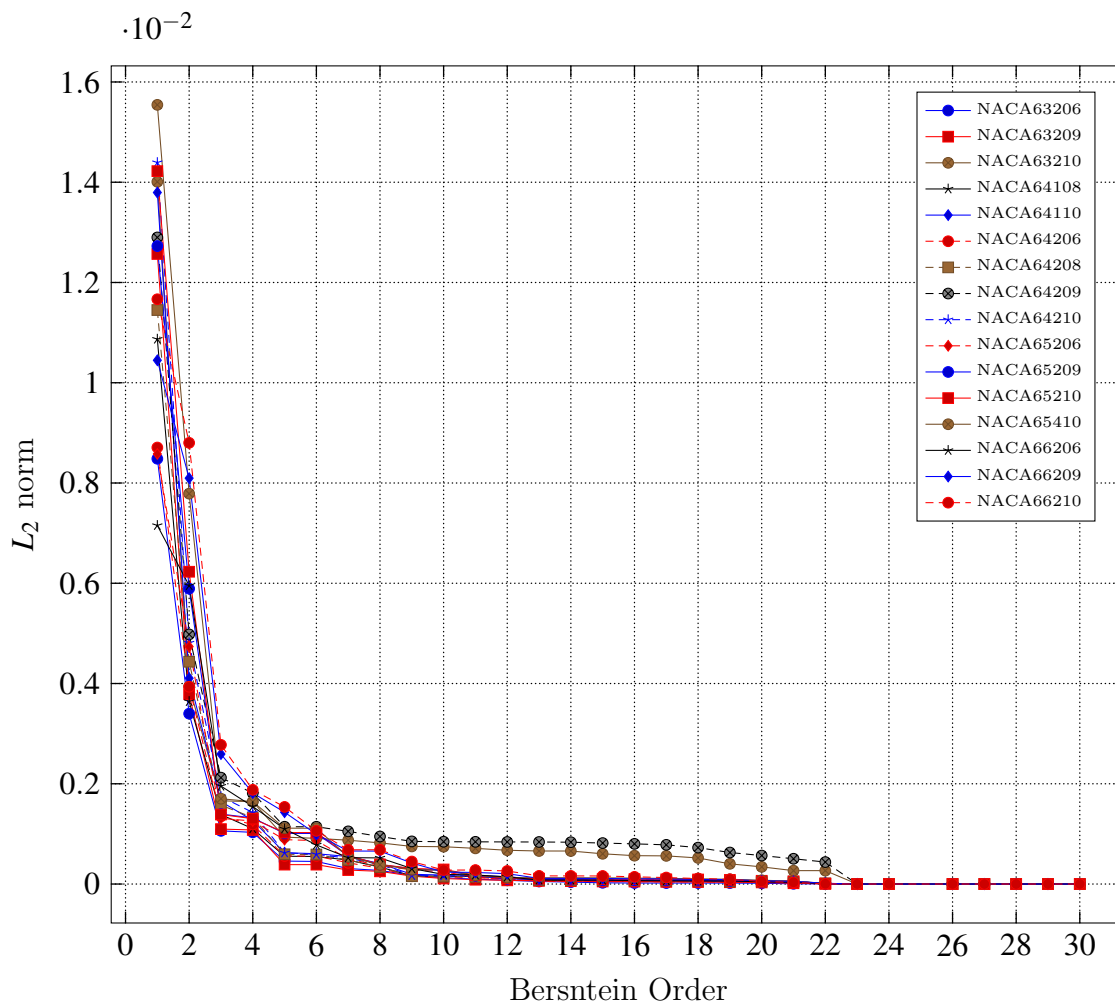


Figure 3.5: Order of the CST Parametrisation

An study of the effect of the order of the CST parametrisation has been carried out for each airfoil for the upper and lower surfaces in order to determine which is the best compromise solution to adopt (Appendix D). The analysis consisted of calculating the L_2 norm, which is the square root of the sum of deviations between mapped and original airfoils. As an example shown in Figure 3.6 for a set of 6-Series airfoils, the deviation falls exponentially until a 5th order parametrisation, when it becomes flat. This fashion is observed for all the airfoils for both surfaces. For this reason, a 5th order CST parametrisation has been chosen for the mapping, which means that a total of 5 + 1 design variables are used for each surface, adding up a total of 12 design variables rearranged in the following manner for the posterior optimisation:

$$\begin{aligned}
 \mathbf{b} &= [b_{1U} \ b_{1L} \ b_{2U} \ b_{2L} \ b_{3U} \ b_{3L} \ b_{4U} \ b_{4L} \ b_{5U} \ b_{5L} \ b_{6U} \ b_{6L}]^T \\
 \mathbf{b} &= [b_1 \ b_2 \ b_3 \ b_4 \ b_5 \ b_6 \ b_7 \ b_8 \ b_9 \ b_{10} \ b_{11} \ b_{12}]^T
 \end{aligned} \tag{3.13}$$

Figure 3.6: L_2 norm vs Order of the CST Parametrisation

3.3.3 Source of Target Airfoils

As described in the Experimental Data section, 83 airfoils belonging to 4 different families have been mapped. For the mapping, which traduces in calculating \mathbf{b} vector, the raw airfoil geometries were needed. For some families, such as NACA 4 and 5 digits, there are analytical expressions that lead to a customizable obtainment of the point. However, for other families such as 6 and 7 series, this procedure is not available so that airfoils coordinate points have been taken from an online database

[32]. The problem encountered when using coordinate points from databases is that in some cases the discretisation has been low (for example only 51 points for a target airfoil) and also, in some cases, some points were wrong and led to dirty geometries. An example of these problems is shown in Figure 3.7a, where it can be appreciated how the target geometry has outlier point and how the clustering of points near to the trailing and leading edges is poor. Despite of these problems, in Figure 3.7b is observed how although the airfoil to map is not a perfect sample, the CST parametrised airfoil has achieved an effective mapping with a correct clustering of points and smooth shape.

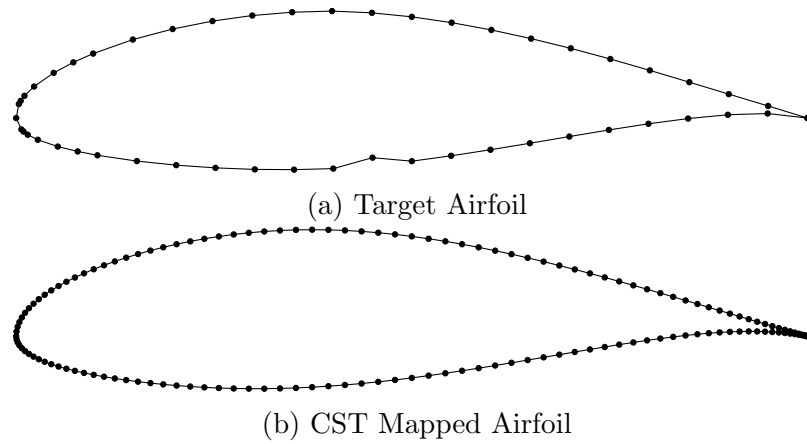


Figure 3.7: Comparison of CST cleaning map of a dirty geometry

The process of the CST parametrisation is summarised in the diagram shown in Figure 3.8.

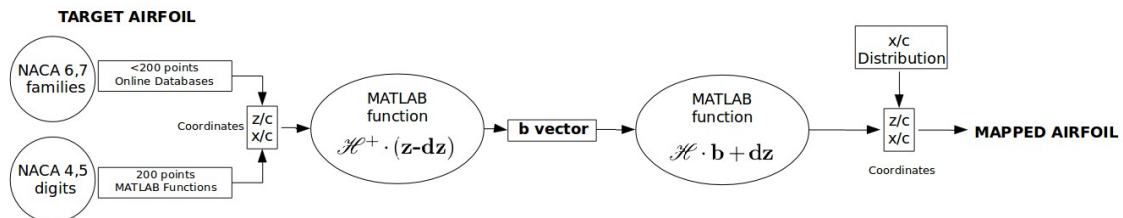


Figure 3.8: CST Parametrisation process

3.4 Surrogate Model Building

A surrogate model mimics the behaviour of the system that is modelling based on a response surface model, being computationally much cheaper than a full simulation procedure.

In this project, due to the lack of reliable experimental data in regions near to maximum lift conditions (and also near to airfoil stall), the surrogate model has been built for a cruise flight condition, specified in Table 3.1. Thus, the surrogate includes a matrix called **A** in which each row is a vector **b** calculated from the mapped airfoil, and a matrix called **Costs** in which each row includes the experimental C_l and C_d of the mapped airfoils. Thereby, matrices **A** and **Costs** have as many rows as included airfoils, bearing in mind that in **Costs**, every i -row has to include the experimental data of the correspondent i -mapped airfoil represented by vector **b**. As shown below, the number of columns in matrix **A** is equal to the design variables which, in turn, is equal to $2(n+1)$, being n the order of the CST parametrisation.

$$A = \begin{bmatrix} b_{1U}^1 & b_{1L}^1 & b_{2U}^1 & b_{2L}^1 & b_{3U}^1 & b_{3L}^1 & b_{4U}^1 & b_{4L}^1 & b_{5U}^1 & b_{5L}^1 & b_{6U}^1 & b_{6L}^1 \\ b_{1U}^2 & b_{1L}^2 & b_{2U}^2 & b_{2L}^2 & b_{3U}^2 & b_{3L}^2 & b_{4U}^2 & b_{4L}^2 & b_{5U}^2 & b_{5L}^2 & b_{6U}^2 & b_{6L}^2 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ b_{1U}^k & b_{1L}^k & b_{2U}^k & b_{2L}^k & b_{3U}^k & b_{3L}^k & b_{4U}^k & b_{4L}^k & b_{5U}^k & b_{5L}^k & b_{6U}^k & b_{6L}^k \end{bmatrix}, \text{Costs} = \begin{bmatrix} C_l^1 & C_d^1 \\ C_l^1 & C_d^1 \\ \cdot & \cdot \\ \cdot & \cdot \\ C_l^k & C_d^k \end{bmatrix}$$

Where k is the number of mapped airfoils (3.14)

Table 3.1: Conditions of the experimental data

Experimental data	C_l, C_d
Reynolds number, Re	$3 \cdot 10^6$
Mach number, M	0.16
Angle of attack, α	3°

It is worth to say that two different techniques have been used to build the response surface model, which are Radial Basis Functions and Kriging. An ordinary method of the latter has been implemented as proposed for further work in [2].

3.4.1 Selection of Acceptable Mapped Airfoils

Although the mapping of the 83 airfoils has been done, not all of them have been included in the surrogate model. The selection criteria considers the CST mapped geometries as well as the target “raw” geometries in order to decide whether an airfoil is selected or not:

- Firstly, the target “raw” geometries coming from either databases [32] (NACA 6,7 Series) or analytical expressions (NACA 4,5 digits) were simulated in Xfoil and the lift and drag coefficients were extracted. For these simulations, the same conditions as in the data from experimental measurements were set, which are $\alpha = 3^\circ$, $Re = 3 \cdot 10^6$ and $Mach = 0.16$.
- In a same fashion, the CST mapped geometries were also simulated in Xfoil.
- Thereby, three different procedures were used to compare the lift and drag coefficients: the experimental data, the target geometries and the mapped geometries simulation.

- The lift and drag coefficients discrepancies were examined between the CST mapped airfoils and the target “raw” geometries, and also between the CST mapped geometries and the experimental data. More value was given to CST mapped airfoil coefficients since the points distribution was better than in the target geometries in terms of better point discretisation in leading and trailing edges.
- For an airfoil to be selected it had to meet the deviation requirements of Table 3.2:

Table 3.2: Criteria for including an airfoil in the surrogate

	Dev. raw vs CST		Dev. CST vs Exp.	
	C_l	C_d	C_l	C_d
Accepted if	<3%	<7%	<15%	<20%
Accepted if	<8%	<25%	<12%	<10%

3.4.2 Radial Basis Functions (RBF) Surrogate

One type of surrogate used in this thesis includes Radial Basis Functions (RBF) interpolation, a relatively easy technique based on the computed distance of two points in a n -dimensional space. An example of application of this technique can be found in [33].

A general expression of RBF is given below, where φ function can be a variety of functions. However, in this thesis, a linear euclidean distance has been used [33].

$$f(x) = \sum_{j=1}^N \lambda_j \varphi_j(r_j) \quad (3.15)$$

The following steps summarise the steps followed to estimate the C_l and C_d of a new input design \mathbf{b} vector using the mapped airfoils, Matrix \mathbf{A} , and the experimental data, Matrix \mathbf{Costs} .

1. The data in Matrix \mathbf{A} is standardised by columns by means of subtracting the mean and dividing by the standard deviation. By doing this, any value outweighs amongst the rest in Matrix \mathbf{A} . The new design \mathbf{b} vector is included at the bottom of Matrix \mathbf{A} and used in this process.
2. Single Value Decomposition (SVD) of Matrix \mathbf{A} is performed in order to work in Principal Component Axes (PCA) [31].
3. By working in PCA, Mahalanobis distances, which are equivalent to Euclidean Distances but in principal axes, are calculated between the new design \mathbf{b} vector and the rest of the airfoil mapped \mathbf{b} vectors.
4. The mapped airfoil \mathbf{b} vectors are reordered in Matrix \mathbf{A} depending on the Mahalanobis distance to the new design so that if a cluster of N mapped airfoils is used, only the closer mapped designs to the new design will be included in the interpolation.
5. A linear system of equations is solved and the weights λ are used to calculate the C_l and C_d for the new design. For example, the interpolation of the lift coefficient for a new design vector of b coefficients taking into account a cluster of N mapped airfoils is shown below, clarifying that in b_a^g , the subscript a represents the number of the mapped airfoil (that can go from 1 to a maximum of 69) and the superscript g represents the component of the \mathbf{b} design vector, that can go from 1 to 12:

$$\begin{bmatrix} \varphi(\|b_1^1 - b_1^1\|) + \dots + \varphi(\|b_1^{12} - b_1^{12}\|) & \varphi(\|b_1^1 - b_2^1\|) + \dots + \varphi(\|b_1^{12} - b_2^{12}\|) & \dots & \varphi(\|b_1^1 - b_N^1\|) + \dots + \varphi(\|b_1^{12} - b_N^{12}\|) \\ \vdots & \varphi(\|b_2^1 - b_2^1\|) + \dots + \varphi(\|b_2^{12} - b_2^{12}\|) & \dots & \varphi(\|b_2^1 - b_N^1\|) + \dots + \varphi(\|b_2^{12} - b_N^{12}\|) \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \dots & \varphi(\|b_N^1 - b_N^1\|) + \dots + \varphi(\|b_N^{12} - b_N^{12}\|) \end{bmatrix} \cdot \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix} = \begin{bmatrix} C_{j1} \\ C_{j2} \\ \vdots \\ C_{jN} \end{bmatrix}$$

With a slight modifications, this RBF surrogate model is programmed in a FORTRAN function provided by Dr. Kipouros and which be found in Appendix F.

3.4.3 Kriging Surrogate

Apart from RBF model, an ordinary Kriging model has been implemented in MATLAB for the surrogate optimisation case of study (included in Appendix G). Typically used in geostatistics, Kriging is an interpolation method where the interpolated values are modelled by Gaussian processes, giving the best linear unbiased estimation and also minimising the error variance between the interpolated and the real value. The analysis of the characteristics of Kriging lies out of the scope of this thesis, however, extensive information and examples of its applications can be found in [8]. The Kriging model implemented in this research project is based on course notes from [36] and the main steps that have been followed are summarised below:

1. **Empirical Semivariograms** .The empirical semivariograms γ^* of the experimental lift and drag coefficients have been calculated using expressions 3.16. These functions describe the spatial correlation of the variables under analysis (Figure 3.9).

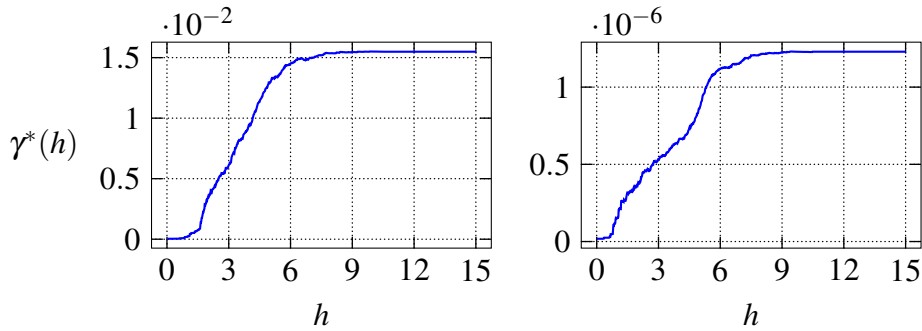
$$\begin{aligned}\gamma_{C_l}^*(h) &= \frac{1}{2n(h)} \sum_{j=1}^{n(h)} [C_l(b_j) - C_l(b_j + h)]^2 \\ \gamma_{C_d}^*(h) &= \frac{1}{2n(h)} \sum_{j=1}^{n(h)} [C_d(b_j) - C_d(b_j + h)]^2\end{aligned}\quad (3.16)$$

Where n is the number of points contained in h for each evaluation.

Since the Kriging model is going to be used for both lift and drag coefficients, it is necessary to have them working in the same semivariogram model, which means that non-dimensional model needs to be used because of the different order of magnitude of both data. For this purpose, both empirical semi-variograms are divided by their maximum value, $\gamma_{\max C_L}^*, \gamma_{\max C_D}^*$, respectively.

$$\gamma_{\max C_l}^* = 0.0155 \quad (3.17)$$

$$\gamma_{\max C_d}^* = 1.2305 \cdot 10^{-6} \quad (3.18)$$



(a) Empirical semi-variogram for C_l

(b) Empirical semi-variogram for C_d

Figure 3.9: Empirical semi-variograms, γ^*

2. **Fitting of empirical semivariograms.** Once the empirical semivariograms have been estimated, they are fitted to a known semivariograms models γ . Thus, γ^* bears the same relationship to γ that a histogram does to a probability distribution. When the distance h become very large, the sample values, in this case the new design vector \mathbf{b} , will become independent of one another and the semivariogram value will the become more or less constant since it will be calculating the difference between sets of independent samples [8]. In this approach a spherical model has been chosen because is the ideal shape, being to geostatistics as the Normal distribution is to statistics (Figure 3.10):

$$\frac{\gamma(h)}{\gamma_{max}^*} = \left(\frac{3h}{2a} - \frac{h^3}{2a^3} \right), \quad h \in [0, a] \quad (3.19)$$

$$\frac{\gamma(h)}{\gamma_{max}^*} = 1, \quad h \in (a, \infty) \quad (3.20)$$

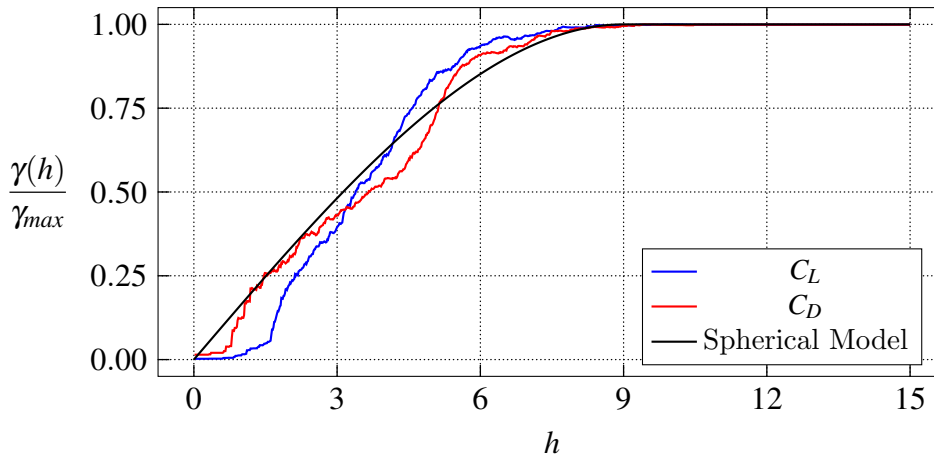


Figure 3.10: Non-dimensional Semi-variogram model used

3. **Weights calculation.** The weights of the interpolation are calculated by solving the equation system coming from the minimisation of the mean square

error expression 3.21 between the interpolated and the real values. In this expression w are the weights of the interpolation, λ is a Lagrange multiplier due to the minimisation and γ is the semivariogram model used:

$$\begin{aligned} MSE &= E\left(\left(\widehat{Z}_0 - Z_0\right)^2\right) \\ MSE &= -\sum_i \sum_j w_i w_j \gamma_{ij} + 2 \cdot \sum w_i \gamma_{i0} + 2\lambda \left(\sum w_i - 1\right) \end{aligned} \quad (3.21)$$

3.5 Multiobjective optimisation

Once the surrogate models have been constructed, the multi-objective optimisation process is carried out. The aim of the multiobjective optimisation has been to find design vectors \mathbf{b} that lead to a minimum drag and maximum lift, which translates in minimising the C_d and maximising the C_l coefficients, for the flight conditions stated in previous sections. Hence, these two coefficients are the two objective functions that are evaluated in order to find set of solutions that are as close as possible to the Pareto optimal-front. In addition, these optimal solutions are desired to be as diverse as possible in order to achieve a good set of trade-off solutions in the multi-objective optimisation.

3.5.1 Algorithms used

The problem under analysis is of aerodynamic nature so it is highly constrained and has a large number of local minima. Therefore, unlike gradient based methods that can get stuck in local optimum regions and be more time consuming, stochastic optimisation has been used in order to find multiple optimum regions in the design

space. For doing this, two different algorithms have been used: Tabu Search and Genetic Algorithms.

Multi-Objective Tabu Search Algorithm (MOTSII)

This family of algorithms performs very well when they are used for aerodynamic optimisation. The particular approach of Tabu Search algorithm used in this thesis was developed by Jaeggi et al [22]. The process is characterised for being sequential and iterative, integrating a systematic local search with stochastic components.

The algorithm uses four different memories that store the points (design variables) evaluated in the process (Figure 3.11):

- **Short Term Memory (STM)**, in this memory recently visited points are recorded, being considered as Tabu and forbidden to call them again.
- **Medium Term Memory (MTM)**, in this memory Pareto optimal points are saved.
- **Intensification Memory**, after a Hooke and Jeeves (H& J) move, Pareto equivalent points that haven't been selected are kept in this memory for future intensification. More about H& J move can be found in [19].
- **Long Term Memory**, this memory stores information about regions that have been sought so as to execute a search diversification.

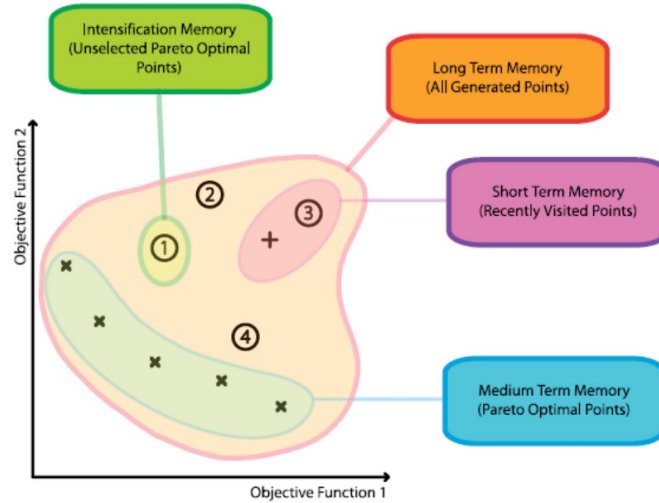


Figure 3.11: Tabu Search Algorithm Memories[22]

Every design variable is divided into a number of n regions and the algorithm keeps a i local counter of iterations that can be either reset when a new point is sent to the MTM or used to reduce the step size, intensify or diversify the search if a maximum number of local iterations is reached.

Non-Sorting Genetic Algorithm (NSGAII)

This version of NSGA algorithm was introduced to overcome its predecessor shortcomings such as a the non-dominated sorting computational complexity, the deficit in elitism and the necessity of the sharing parameter definition. It features a quick non-dominated sorting reducing the computational complexity from $O(MN^3)$ to $O(MN^2)$, being M the number of objectives and N the size of the population. The basic steps that the algorithm follows are summarised below and sketched in Figure 3.12. However, extensive explanation can be found in [11]:

1. Initiation of main loop with a random parent population P_0 .
2. Offspring obtaining Q_0 via binary recombination, tournament and mutation

operations.

3. In generation t , R_t population is obtained by combination of P_t and Q_t and classified following non-domination procedures with the best non-dominated solutions being saved in F_1 .
4. If size of F_1 is smaller than the size of the next generation N , all its members are included in the next generation P_{t+1} . Until N is reached, the following included members will belong to next best fronts F_2, F_3 , etc.
5. Q_{t+1} new population is calculated from P_{t+1} applying genetic operators.

Unlike Tabu Search algorithm, genetic algorithms fix a maximum size of the achievable Pareto front since the number of population members is predefined.

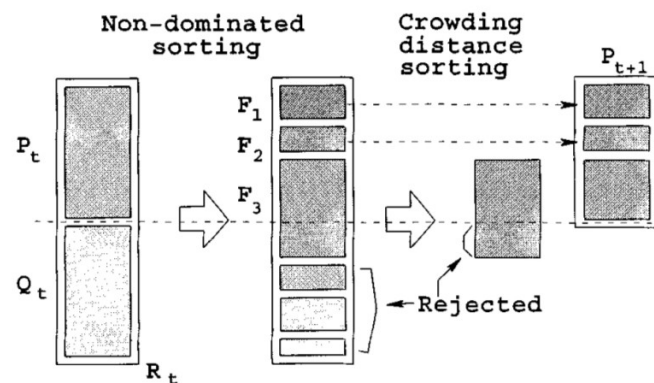


Figure 3.12: NSGAI Procedure [11]

Nimrod/O

The optimisations have been undertaken through the Pareto server at Cranfield University using the Nimrod/O software platform. This platform can perform computational jobs simultaneously via clusters and grid environments and includes the aforementioned algorithms, MOTSII and NSGAI, amongst others. In order to

launch an optimisation, Nimrod/O needs a schedule file with the definition of the search space of the design parameters, the establishment of the tasks to do in order to compute the objective functions and the specification of the optimisation method that is going to be used [35]. An example of a Nimrod/O schedule file for the Kriging surrogate optimisation task can be found in Appendix H.

3.5.2 Surrogate-Based Optimisation

Once the response surface models are built and the optimisation algorithms/interfaces are introduced, the surrogate-based optimisations are launched for both Kriging and RBF models. Having a look at Matrix **A** (Appendix J) which contains the mapped airfoils **b** vectors, it can be inferred that the whole spectre of mapped airfoils lie inside the values given in the Table 3.3.

Table 3.3: **b** vector search space

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}
min.value	0.1	0.1	0.1	0.1	0	0	0	0	-0.1	-0.1	-0.1	-0.1
max.value	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5

For the surrogate optimisation, MOTSII and NSGAI algorithms have been used with their main setting parameters being exposed in Table 3.4. Both optimisations started from a NACA 2421 datum geometry (Table 3.5) although, as it will explained later in results section, the datum condition have minimum effect in the final Pareto front of optimal solutions of the surrogate-based optimisation.

Table 3.4: Surrogate Optimisation settings

MOTSII	
Number of regions	4
Size of Short Term Memory	20
Intensification	15
Diversification	25
Step size reduction	50
Initial step size	0.04-0.06
Step size reduction-factor	0.5
Size of sample	6
Number of evaluations	10000
NSGAI	
Seed	0.3
Population size	20
Generations number	2000
Crossover probability	0.9
Mutation probability	0.1
Crossover index	5
Mutation index	15

Table 3.5: \mathbf{b} vector of datum airfoil

Datum	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	b_{12}
NACA2421	0.3484	0.2589	0.2727	0.2547	0.4046	0.1757	0.229	0.2312	0.3597	0.1604	0.308	0.1923

Once this surrogate-based optimisations are completed, a small sample of the

optimum design points is evaluated in Xfoil and corrected C_l' and C_d' values are calculated. From this new cloud of points, the Pareto optimal front is obtained. Then, one of these re-evaluated values belonging to the new Pareto front is chosen as a starting point for a Xfoil direct optimisation. (Example shown in Figure 3.13)

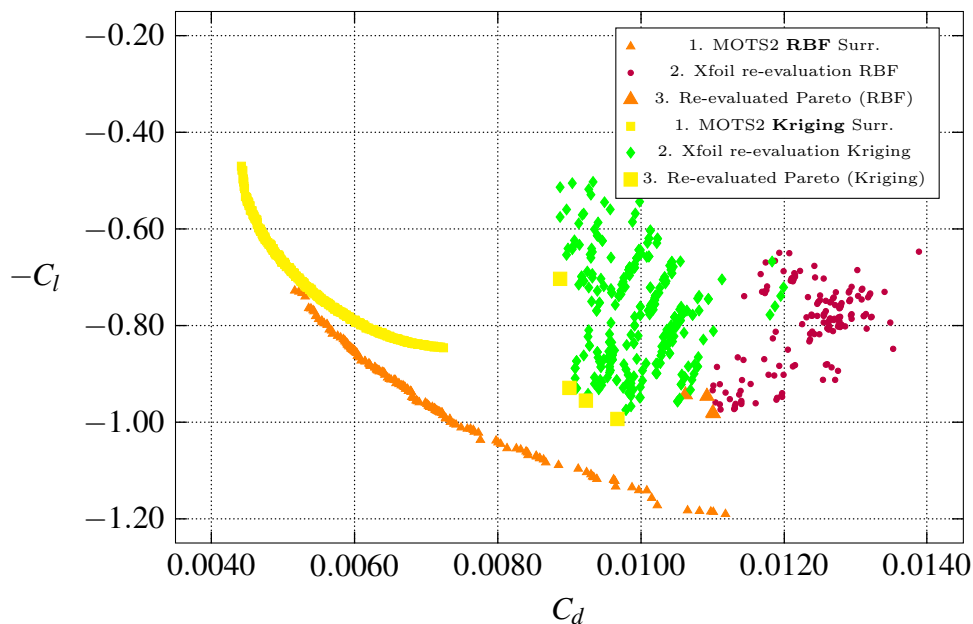


Figure 3.13: Surrogate-based optimisation

3.5.3 Xfoil Direct Optimisation

With the aim of comparing to the surrogate-based optimisations, a parallel direct optimisation has been carried out using the flow solver program Xfoil. This software, developed by M. Drela [15] under GNU General Public License, performs viscous and inviscid analysis of input airfoils geometries using panels method along with boundary layer and compressibility corrections, amongst other capabilities. In order to launch the optimisation in Nimrod/O, an integrated Free Form Deformation + Xfoil package provided by Dr. Kipouros was modified so as to implement the CST parametrisation with the Xfoil code.

Thus, the automatic software gets as an input a design vector \mathbf{b} from the optimiser, builds the CST parametrised airfoil based on \mathbf{b} , performs the viscous flow simulation under the same conditions of the surrogate model and the experimental data (Table 3.1) and calculates the lift and drag coefficients which are the objective functions that are used by Nimrod/O to start the process again by creating a new design vector \mathbf{b} . The CST FORTRAN function that Xfoil package uses in order to build the airfoil geometry based on \mathbf{b} can be found in Appendix I.

It must be said that only MOTSII algorithm has been used for the Xfoil optimisation. The reason for this is that Xfoil, unlike the surrogate models which aren't concerned about physical feasibility and just interpolate from input data, is more restrictive with the input geometries. So, if for example a new design vector \mathbf{b} leads to a non-aerodynamically geometry, Xfoil doesn't converge and the objective functions can't be evaluated or simply doesn't make sense, particularly the drag coefficient. Thereby, this problem is more likely to happen if a genetic algorithm is used because of the limitation in terms of population size in each generation.

The search design space introduced in Xfoil is the same as in the surrogates (Table 3.3) in order make the optimiser to seek for feasible designs that lie inside the margins of the mapped airfoils \mathbf{b} vectors. Also the settings used in MOTSII schedule file are the same as in the surrogates (Table 3.4) and the \mathbf{b} vector considered as datum condition is again from the NACA2421 profile (Table 3.5).

3.6 Post-Processing

After all the mapping and optimisation process, an important part which cannot be undervalued is the post-processing of the results. A very large amount of data and sets of optimum solutions created during the optimisation processes need to be

analysed in order to extract proper conclusions for the work done. Being able to identify the relationships between the design parameters and the objective functions is vital to make final decisions. For this purpose, Parallel Coordinates interactive visualisation method has been used.

The method, introduced by Inselberg [20], [21] facilitates the visualisation of multidimensional design vector and objective functions in a 2D space.

In this thesis, a Parallel coordinates web-application [18] has been used to identify patterns and relations amongst the design parameters (\mathbf{b} vectors) and their physical meaning in the optimum airfoil geometries.

3.7 Outlook of the whole model

In Figure 3.14, the whole model explained in this methodology chapter has been sketched.

1. Experimental drag and lift coefficients from airfoils at a given flight condition have been gathered from NACA measurements.
2. Those airfoils from which the experimental data has been retrieved have been mapped using CST parametrisation technique.
3. A surrogate model based on these experimental data and mapped parameters has been built using two different approaches.
4. A multi-objective optimisation with the surrogate models has been done and Pareto optimal designs has been saved.
5. When surrogate-based optimisations have been obtained, an equally distributed sample of the optimum design points have been evaluated in Xfoil and cor-

rected C'_l and C'_d values have been calculated. Then, one of these re-evaluated values has been chosen as a starting point for a Xfoil direct optimisation in order to accomplish the model described in Figure 1.1.

6. A parallel direct optimisation using Xfoil code has also been done and the optimal designs have also been saved and compared to the surrogate-based optimisation.

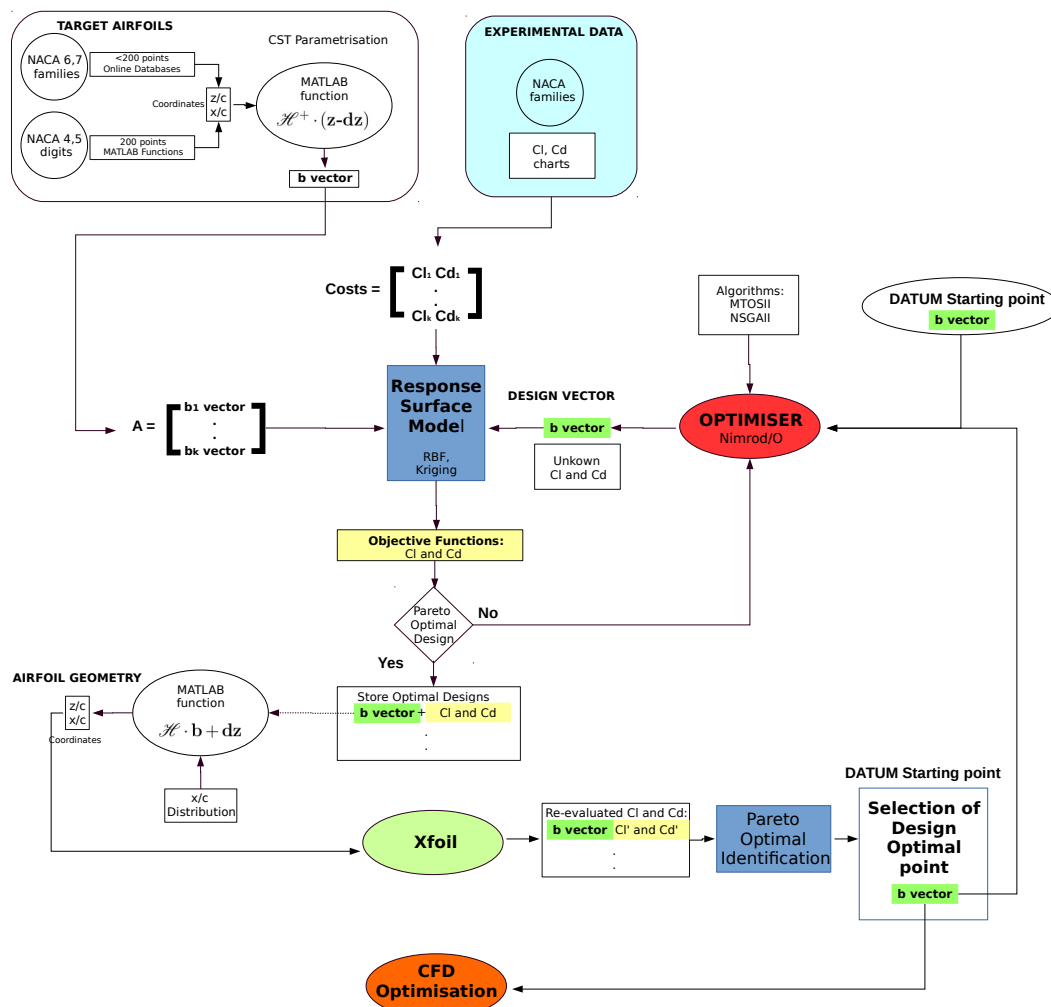


Figure 3.14: Whole Process

Chapter 4

Results Analysis

The results obtained by following the proposed research methodology are presented throughout this chapter. They are classified depending on the phase of the methodology. Firstly, the results related to the airfoil mapping are given. Then, the results regarding the surrogate-based and a direct Xfoil optimisation are discussed. Finally, the outcome of the proposed design methodology, with the results of the combined Xfoil and surrogate-based optimisation are presented.

4.1 Airfoil Mapping Accuracy

The CST parametrisation of 83 airfoils belonging to four different families from which the experimental data was available was carried out, and the mean square error L_2 between the mapped and the target geometries was calculated. This study was performed to see that a 5th order in the CST parametrisation was a good compromise solution. Due to the high amount of mapped airfoils, the charts showing the L_2 norms vs the order of the parametrisation can be found in Appendix D. Here, general comments based on those charts are given depending on the family mapped:

- **NACA 4-digits family:** the L_2 error decreases in a slower fashion with the order of the parametrisation than in other families. In addition, for this family thicker airfoils present higher error, especially in the parametrisation of the lower surfaces.
- **NACA 5-digits family:** this family follows the same trend as NACA 4-digits. Thicker airfoils such as NACA 23024 present a error almost twice as higher as the thinnest airfoil regardless the order of the parametrisation.
- **NACA 6-series family:** this series converges more rapidly to stationary values of L_2 error as the order of the parametrisation increases. Even though thicker airfoils still present more error than the thinner ones, the difference is less remarkable than in previous families.
- **NACA 7-series family:** despite of only being conformed by two airfoils, it can be appreciated that this family follows the same trend as 6-series.

In addition, due to the fact that upper and lower surfaces had to be mapped separately, they present different trends in their L_2 norms. Generally, the upper surface L_2 norm error is higher, almost twice of the value of the lower surface, for orders of parametrisation below 3. This is mainly due to the higher curvature needed for the generation of positive lift on the upper surface, being more difficult to map when low order of parametrisation is set. Nevertheless, this difference disappears when 5th order is reached.

An example of the evolution of the mapping error L_2 for a sample of airfoils is shown in Figure 4.1:

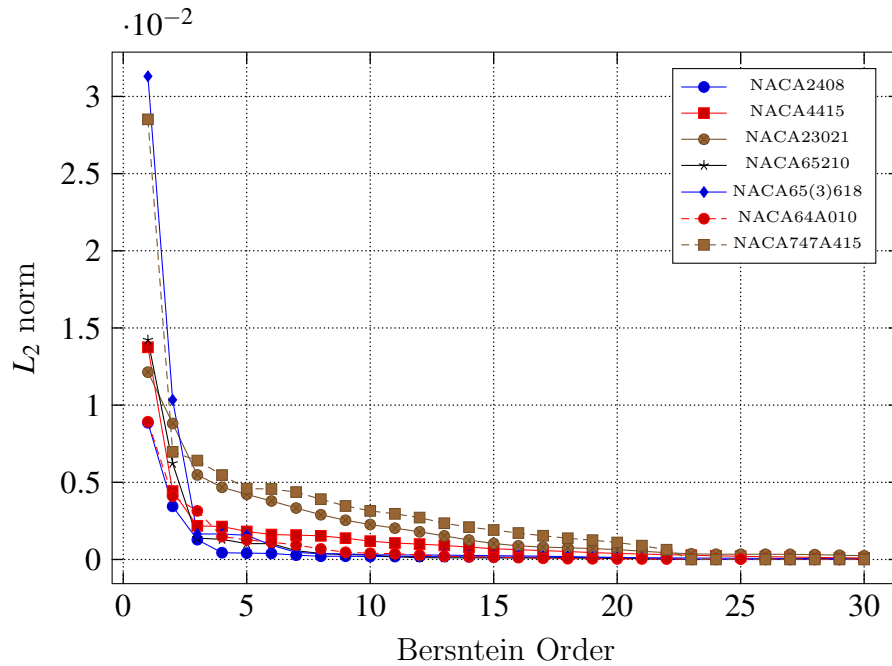


Figure 4.1: Example of the accuracy of the mapping vs the order of the parametrisation for upper surfaces

4.2 Selection of Airfoils included in the Surrogate

According to the criteria explained in Subsection 3.4.1, 69 out of 83 airfoils were selected for being included in the surrogate model. This number represents a 83.13% of the total amount of airfoils that has been mapped. The name of these airfoils is listed in Table 4.1. On the other hand, the full table containing the data required for making the decision whether an mapped airfoil was included or not can be found in Appendix E. Here, in Table 4.2, only the values that do not meet the criteria of Subsection 3.4.1 are shown. Since Xfoil has been proved as an accurate simulation code for low angles of attack and speeds, it is very likely that the experimental data of the rejected airfoils was not accurate enough and perhaps some errors in the measurement processes were introduced, resulting in a more reliable Xfoil performance

prediction. Thereby, as the surrogate model was only based on trustworthy experimental data, they were still rejected.

Table 4.1: Airfoils Included in the Surrogate

NACA 6	NACA 64210	NACA 64(2)215	NACA 65(2)415(a05)
NACA 9	NACA 65206	NACA 64(2)415	NACA 65(3)418(a05)
NACA 1410	NACA 65209	NACA 64(3)218	NACA 65(3)618(a05)
NACA 1412	NACA 65210	NACA 64(3)418	NACA 65(4)421(a05)
NACA 2412	NACA 66206	NACA 64(3)618	NACA 747A315
NACA 2415	NACA 66209	NACA 64(4)221	NACA 747A415
NACA 2418	NACA 63(1)212	NACA 64(4)421	
NACA 2421	NACA 63(1)412	NACA 65(1)212	
NACA 4412	NACA 63(2)015	NACA 65(1)412	
NACA 4415	NACA 63(2)215	NACA 65(2)215	
NACA 4418	NACA 63(2)415	NACA 65(2)415	
NACA 4421	NACA 63(2)615	NACA 65(3)418	
NACA 23012	NACA 63(3)218	NACA 65(3)618	
NACA 23015	NACA 63(3)618	NACA 65(4)221	
NACA 23018	NACA 63(4)221	NACA 65(4)421	
NACA 63206	NACA 63(4)421	NACA 66(1)212	
NACA 63209	NACA 64(1)012	NACA 66(2)215	
NACA 63210	NACA 64(1)112	NACA 66(3)218	
NACA 64110	NACA 64(1)212	NACA 66(4)221	
NACA 64206	NACA 64(1)412	NACA 67(1)215	
NACA 64209	NACA 64(2)015	NACA 65(1)212(a06)	

Table 4.2: Rejected Airfoils

Airfoil	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
	C_l	C_d	C_l	C_d	C_l	C_d	$C_l\%$	$C_d\%$	$C_l\%$	$C_d\%$
NACA 1408	0.65	0.0075	0.445	0.00536	0.446	0.00535	0.2	0.19	31.3	28.7
NACA 2424	0.45	0.0090	0.320	0.00775	0.329	0.00770	2.7	0.65	26.9	14.4
NACA 4424	0.70	0.0100	0.467	0.00868	0.476	0.00859	2.0	1.04	31.9	14.1
NACA 23021	0.45	0.0085	0.323	0.00749	0.307	0.00747	4.9	0.3	31.7	12.1
NACA 23024	0.40	0.0095	0.260	0.00834	0.241	0.00838	7.4	0.5	39.8	11.8
NACA 64108	0.35	0.0065	0.421	0.00647	0.421	0.00632	0.1	2.3	20.3	2.8
NACA 64208	0.50	0.0045	0.508	0.00672	0.507	0.00648	0.1	3.6	1.5	44.0
NACA 65410	0.60	0.0045	0.675	0.00741	0.679	0.00635	0.7	14.3	13.2	41.1
NACA 66210	0.43	0.0060	0.486	0.00720	0.495	0.00653	1.7	9.3	16.4	8.8
NACA 63(3)418	0.68	0.0068	0.714	0.00461	0.702	0.00474	1.7	2.8	4.0	29.8
NACA 65(3)218	0.45	0.0055	0.542	0.00506	0.534	0.00508	1.5	0.4	18.6	7.6
NACA 66(2)415	0.55	0.0045	0.674	0.00583	0.699	0.00473	3.8	18.9	27.2	5.1
NACA 66(3)418	0.60	0.0050	0.714	0.00461	0.702	0.00474	1.7	2.8	17.0	5.2
NACA 66(4)021	0.28	0.0053	0.364	0.00472	0.358	0.00469	1.6	0.6	30.3	10.7

Additionally, an example of the pressure distribution over target and mapped airfoil is presented in Figure 4.2. As stated in previous sections, the CST parametrisation smoothers the “raw” geometry of the target airfoil leading to a smoother pressure distribution. Then, it can be said that for cases where the target geometry has been poor in point distribution, the CST parametrisation has improved the quality of the resulting mapped airfoil.

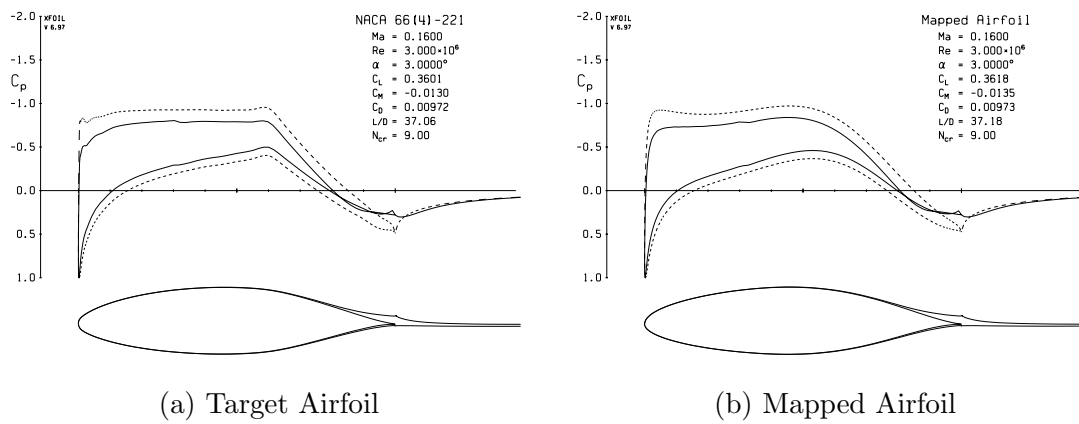


Figure 4.2: Pressure Distribution comparison after parametrisation

4.3 Surrogate Based Optimisation

The results regarding the surrogate-based optimisations are presented in this section. However, since the optimisation processes were stochastic, it was decided to check the effect that different starting points had in the pareto optimal front reached. Five runs with randomly selected starting points were launched for each algorithm (MOTSII and NSGAI) and surrogate method (RBF and Kriging) used. The results reached showed that by using CST parametrisation, the final optimal Pareto fronts were independent of the starting point. This is shown for the Kriging model in Figure 4.3:

4.3.1 Effect of running Stochastic optimisations

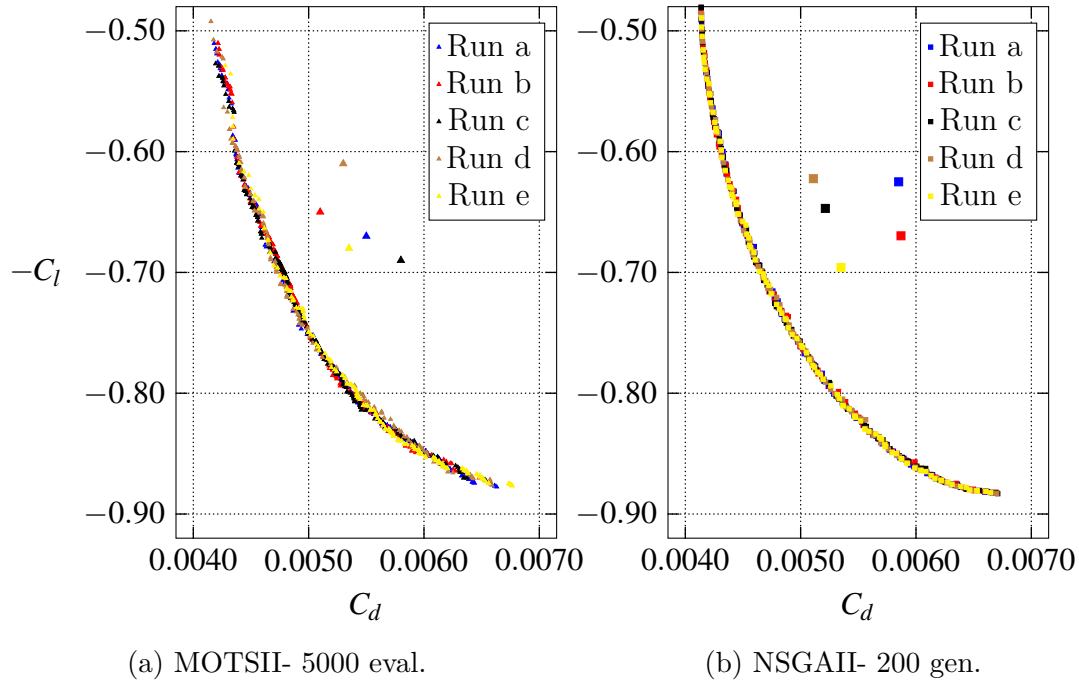


Figure 4.3: Effect of Stochastic Optimisation in the Kriging Surrogate model

4.3.2 Kriging Surrogate

The results of Kriging surrogate-based optimisation for Tabu Search and Genetic algorithm are shown in Figure 4.4. Two numbers of evaluations (1000 and 5000) were chosen for the MOTSII optimisation, leading to a more complete and bigger Pareto front when 5000 evaluations were considered. Since the evaluation of the surrogate model is very cheap, switching from 1000 to 5000 evaluations only took a matter of minutes. The Genetic Algorithm achieved slightly better improvements in lift coefficient than MOTSII although the process took much more time. For this case airfoil NACA2421 was selected as a datum condition.

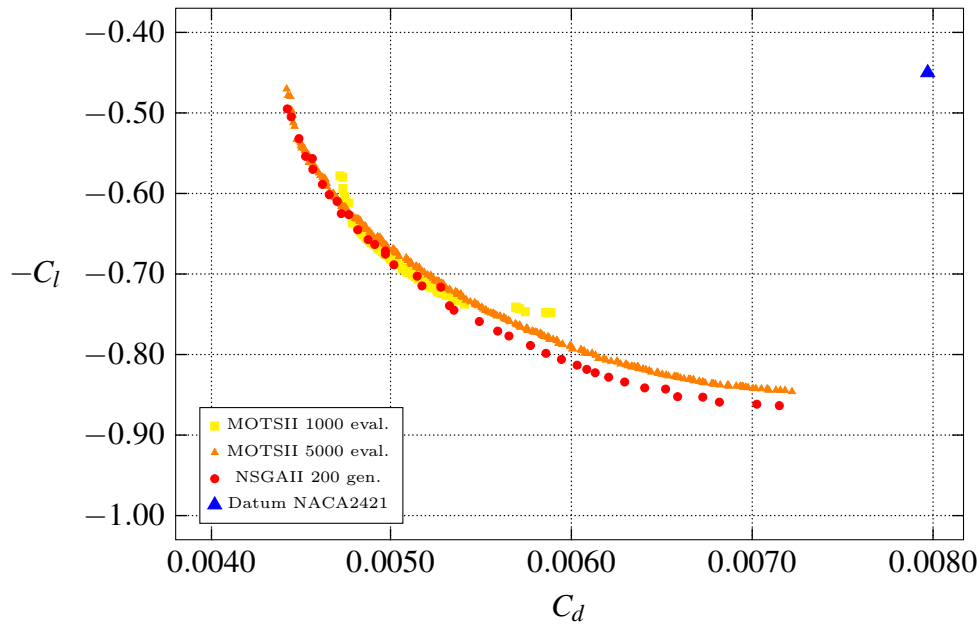


Figure 4.4: Kriging surrogate-based optimisation

4.3.3 RBF Surrogate

The optima Pareto fronts for the RBF surrogate-based optimisation are shown in Figure 4.5 for the two different algorithms considered. As it happened with the Kriging surrogate, a higher number of evaluations from the surrogate under MOTSII algorithm led to a more complete and optimum Pareto front. Again, NACA2421 was selected as a datum condition.

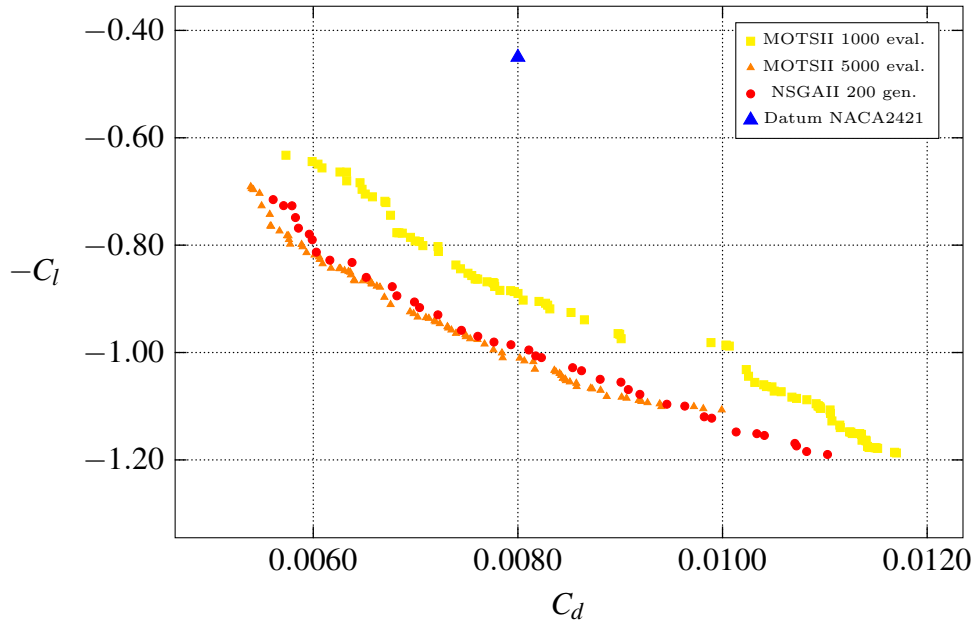


Figure 4.5: RBF surrogate-based optimisation

4.3.4 Comparison of the two surrogate approaches

Once the optimisations were carried out for the two considered surrogate approaches, it was decided that only the solution derived from the MOSTII optimisation was going to be compared. The reasons for that were:

- a. Unlike the Genetic Algorithm, Tabu Search leads to a more efficient search of feasible aerodynamic designs.
- b. It is faster and the number of optimal solutions is not constrained by any population size.
- c. Posterior direct optimisations with Xfoil were only undertaken with MOTSII algorithm because the use of NSGAI was unstable and led to infeasible designs.

The results are compared in Figure 4.6, where it can be observed that the optimisation undertaken with Kriging surrogate improves both lift and drag performance. Conversely, the RBF optimisation did not improve the drag coefficient performance as much as it did with the lift, showing a more disperse distribution. The reasons for the RBF surrogate to behave like this mainly reside in the sophistication of the method, being the Kriging surrogate a more sophisticated Response Surface Model. Therefore, for next comparisons with Xfoil direct optimisation, only Kriging model was used.

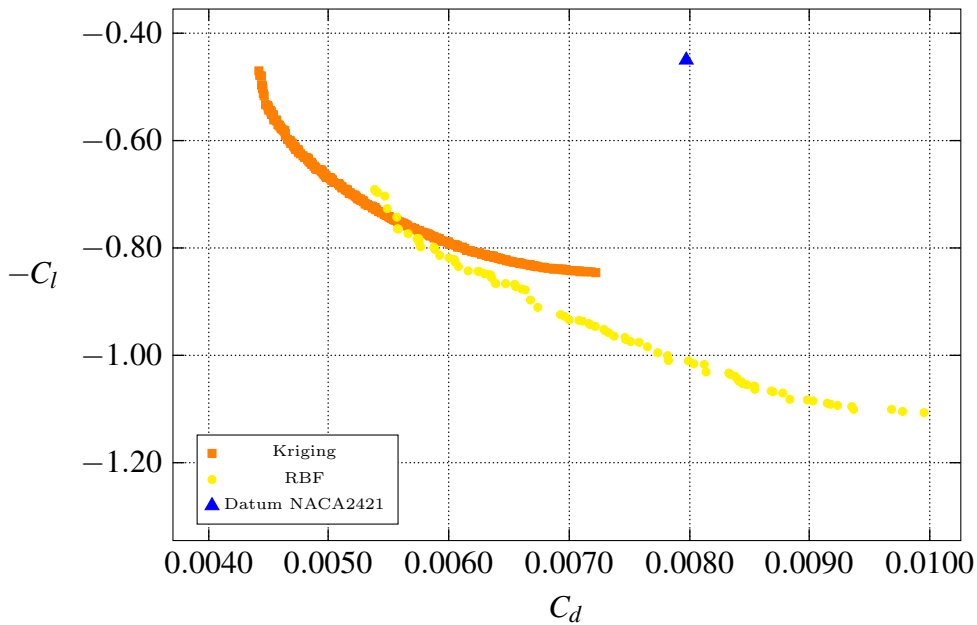


Figure 4.6: Pareto Optimal fronts of the surrogate-based optimisations for 5000 evaluations with MOTSII Algorithm

4.4 Xfoil Direct Optimisation

A direct Xfoil optimisation was necessary in order to compare the validity of the method proposed in the previous chapter. Since the resources needed by this software are low, different number of runs with different number of evaluations were tested

in order to see how the Pareto optimal front evolved. Once again, NACA2421 was selected as a datum condition from where the optimisation was launched. As stated above, only Tabu Search algorithm was finally chosen due to the higher stability in providing aerodynamically feasible designs that were able to be simulated under Xfoil without convergence problems. The results of the direct Xfoil optimisation are shown in Figure 4.7, where the 10000-evaluation optimisation (in green) was considered as the target optimum Pareto front wanted to be reached by the methodology proposed in this thesis.

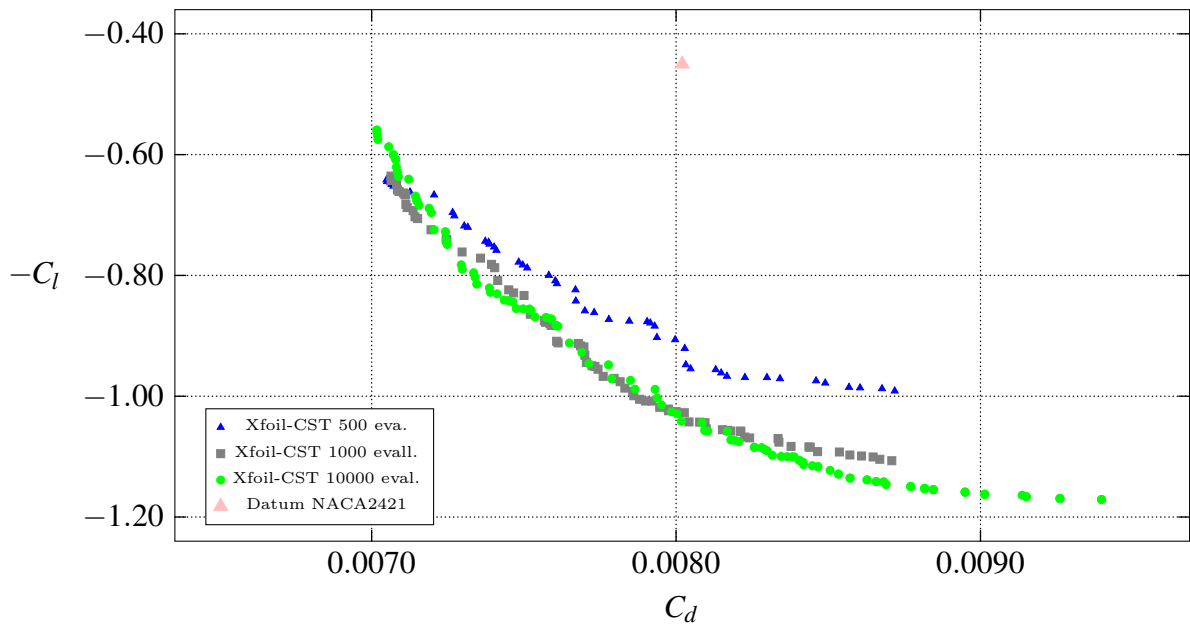


Figure 4.7: Xfoil direct optimisations

4.5 Validation of the methodology proposed

As described in previous sections, the main objective of this thesis was to develop a new design methodology of an airfoil based on experimental data which allows the whole optimisation process to save calculation time by means of introducing some guidance in the search of starting points for a direct optimisation. Thus, the steps

followed in order to prove the methodology are summarised below:

Step 1. Quick Surrogate-based optimisation, 5000 evaluations of the Kriging Surrogate.

Step 2. Selection of a 20 equally-distributed optimal design vectors sample from the Pareto optimal front of the Surrogate-based optimisation.

Step 3. Xfoil evaluation of the sample of 20 design vectors.

Step 4. Pareto optimal front identification of the 20 points re-evaluated sample.

For this case, steps 1-4 are represented below in Figure 4.8:

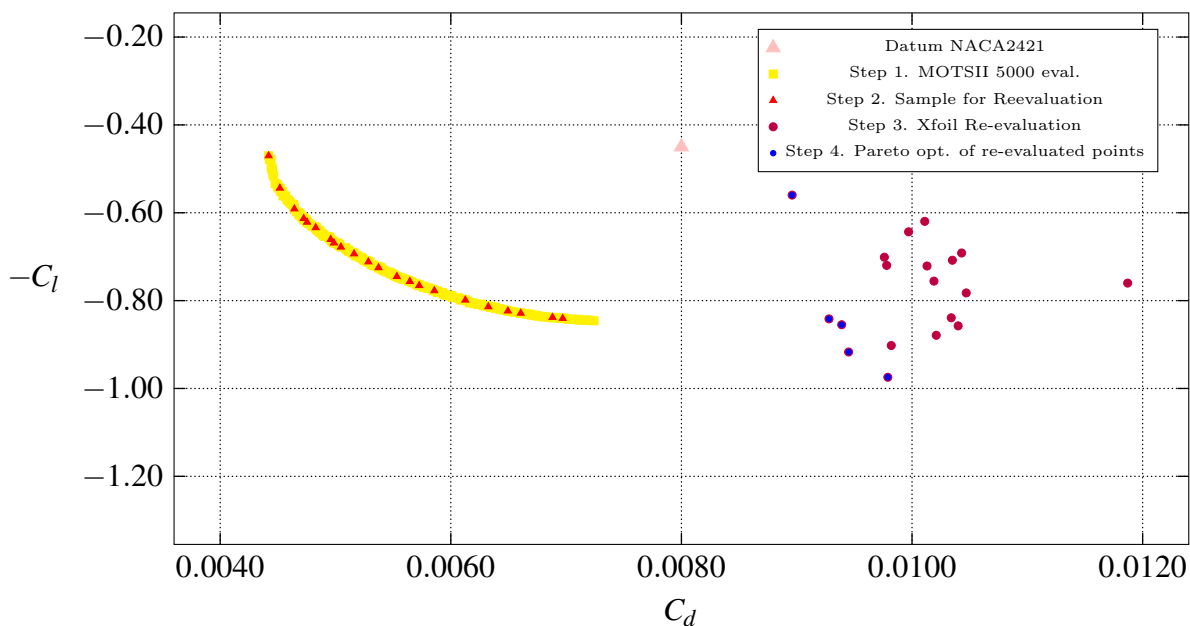


Figure 4.8: Optimisation steps of the methodology proposed

Once the Pareto optimal front of the re-evaluated points was found, the next step was to choose one of those points (there were five in total) and launch a direct Xfoil optimisation from there. As observed in the Figure 4.9, the trend of the Pareto optimal front of the re-evaluated points followed the same trend as the Xfoil direct

optimisation. Hence, 5 candidates were available for being the starting point of the direct Xfoil optimisation and, depending on which one was chosen, the results obtained varied slightly.

As can be appreciated in the Figure 4.9, if a low drag-low lift starting point was chosen (Point 1), the direct optimisation led to zones where designs with similar characteristics in a direct 1000-evaluations optimisation were found. The same applies if a medium drag- medium lift (Point 2) or high-drag-high lift (Point 3) points were selected as the starting points. Thereby, interesting findings need to be highlighted:

- Once the Pareto of the re-evaluated surrogate optimal points is found, it follows the same trend as the Pareto front from a direct optimisation.
- By running a direct, shorter full optimisation from one of those points, the optimal solutions reach the Pareto front of the 1000-evaluations direct optimisation in zones where the designs are similar to the starting point. Hence, potential optimal designs can be searched by selecting a starting point with the same characteristics. For example, the starting from Point 1, which is a point with low C_d but low C_l , led to solutions that share these features within the optimal Pareto front.
- After a direct optimisation of 500 evaluations from the Pareto optimal front of the re-evaluated solutions, the final optimum solutions reach the results that a 1000-evaluations direct optimisation is giving, even improving the solutions for the high-lift/high-drag designs (even if it is compared to a 10000-evaluations direct optimisation).
- Also, a 500-evaluations direct optimisation was carried out to check what

was the difference between this short direct optimisation from the datum NACA2421 and the 500-evaluations direct optimisation from the Pareto of re-evaluated points. Once again, the latter led to better results, especially in designs of high-lift.

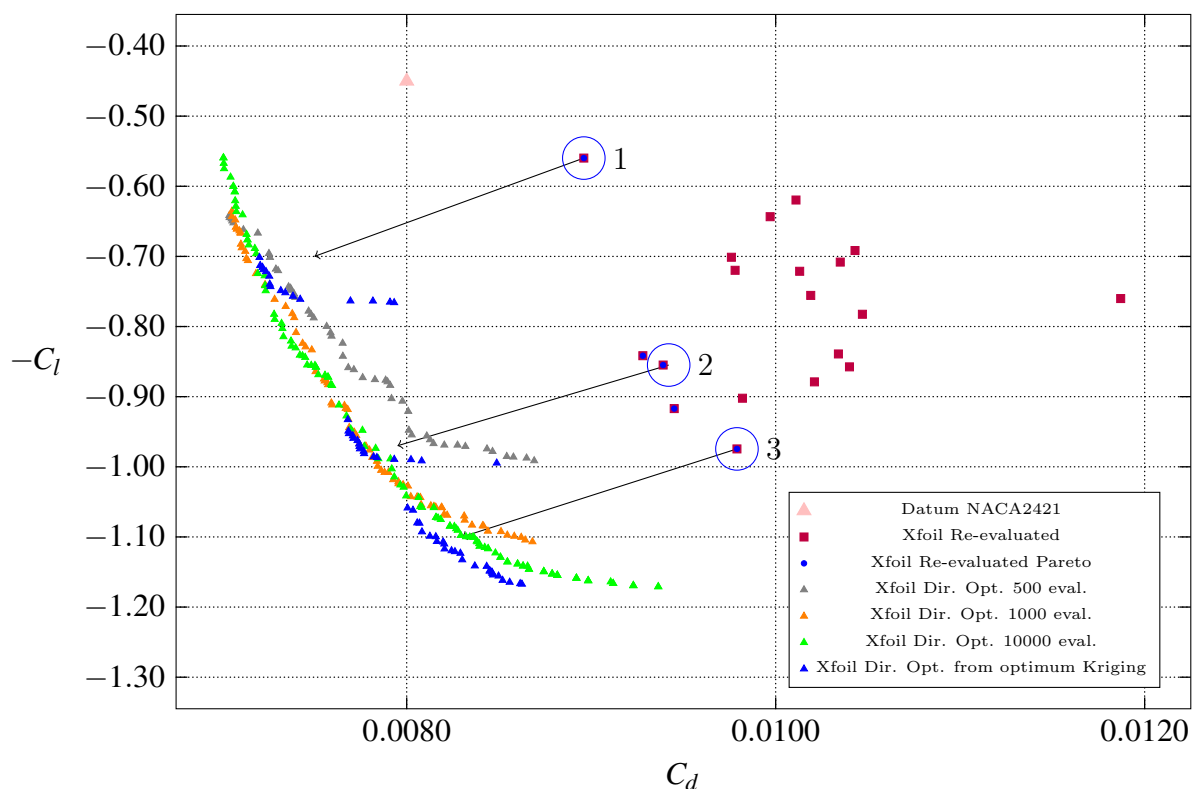


Figure 4.9: Pareto fronts of the Proposed Methodology

So, as a summary, it is shown in Table 4.3 the number of evaluations required for each method, either the direct optimisation or the optimisation based on the methodology proposed. As it can be read, if a direct optimisation of 1000 evaluations is done, only 500+20 evaluations plus the flexibility to find designs with a particular trend are needed following the methodology proposed.

Table 4.3: Benefits in the use of the method developed

Method	number of evaluations
Xfoil-CST direct optimisation	1000
Total	1000
Kriging's Optimal Pareto Re-evaluations	20
Xfoil-CST direct optimisation from optimum	500
Total	520

4.6 Optimum shapes and pressure distributions

After the optimisation of the NACA 2421, that featured a $C_l = 0.45$ and a $C_d = 0.0080$ for the considered flow conditions of $\alpha = 3^\circ$, $Re = 3 \cdot 10^6$ and $Mach = 0.16$, three optimal re-evaluated starting points (Point 1, 2 and 3 in Figure 4.9) were simulated in Xfoil and the pressure distributions are plotted in Figures 4.10a, 4.10c and 4.10e. As it can be appreciated, the optimal designs given by Kriging optimisation are different depending on their position within the Pareto optimal front. For example, low-drag optimal solution (Figure 4.10a from Point 1) features a lower curvature and more symmetry than a high-drag/high-lift solution.

Since the optimisation was carried out for low angle of attack, the drag was already very low and hence the major improvements are observed in the lift coefficient. When the 500-evaluation direct optimisations starting in those designs were done, all the final geometries featured an increase in the airfoil curvature near to the Leading Edge. This means that with Xfoil optimisation, the low pressure peak is displaced to the front of the airfoils whereas with the Kriging optimisation this

peak is kept in the half chord. The Figures 4.10b, 4.10d and 4.10f show examples of solutions that were obtained when the 500-evaluation direct optimisations from Point 1-3 were run.

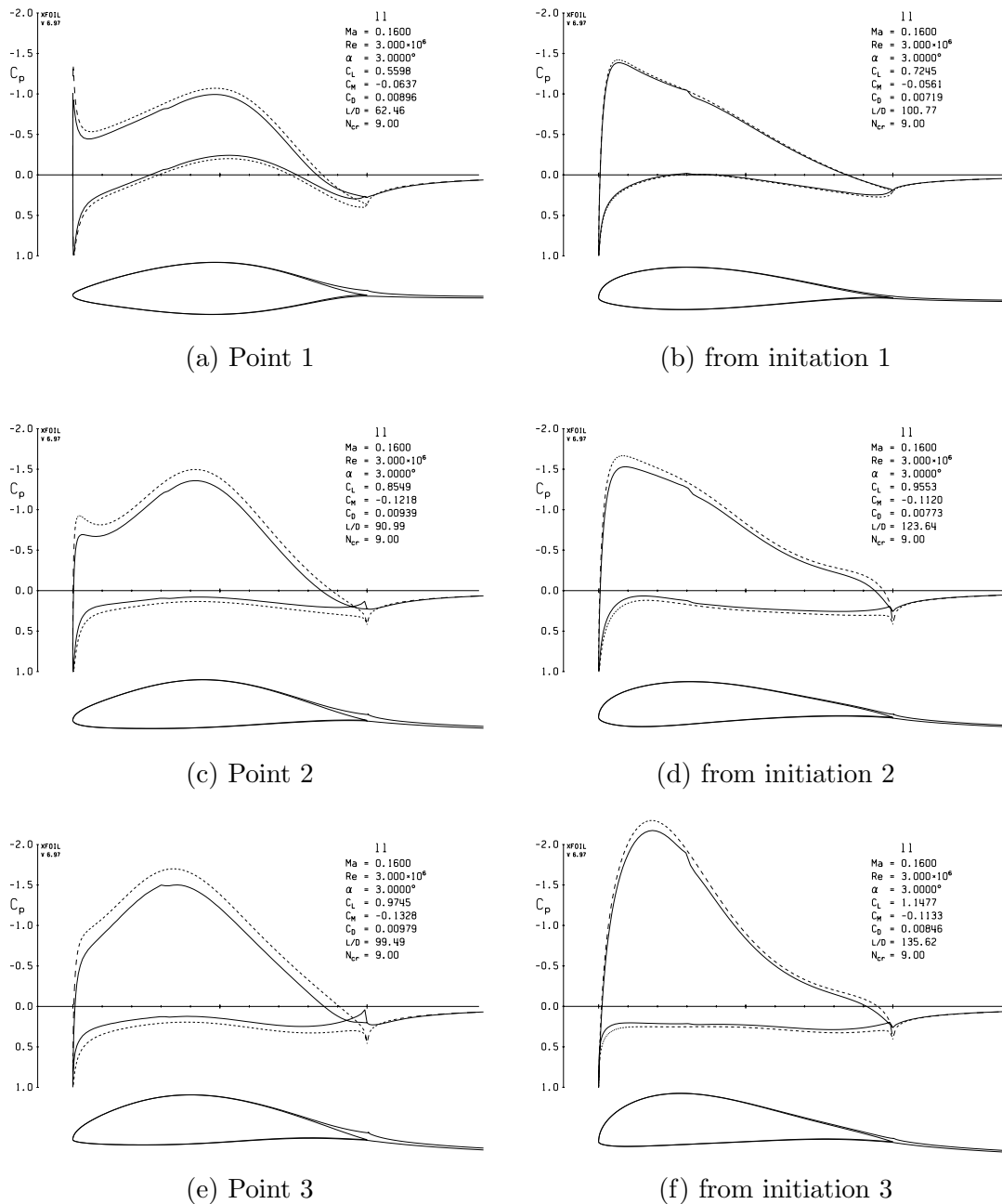


Figure 4.10: Xfoil optimal solutions

4.7 Analysis of results with Parallel Coordinates

In this section, the physical meaning of some components of the design vector, $\mathbf{b} = [b_{1U}, b_{2L}, b_{3U}, b_{4L}, b_{5U}, b_{6L}, b_{7U}, b_{8L}, b_{9U}, b_{10L}, b_{11U}, b_{12L}]$ is inferred by the analysis of the optimum solutions in Parallel Coordinates (i.e. Kriging and Xfoil from initiation in points 1-3). Due to the high number of variables involved (see Appendix K for the whole plots), it has been complicated to identify these relationships, particularly because to identify the patterns, the order of collocation of the variables (which is unfortunately unknown) plays an important role. Nevertheless, some conclusions are presented below:

- There is a direct relationship of b_{1U} coefficient with the drag and lift, because there is a relation between this coefficient and the curvature of the airfoil in the proximity of the Leading Edge for the upper surface. This trend is not observed for b_{1L} , which means that it does not have such a big influence. (Figure 4.11)

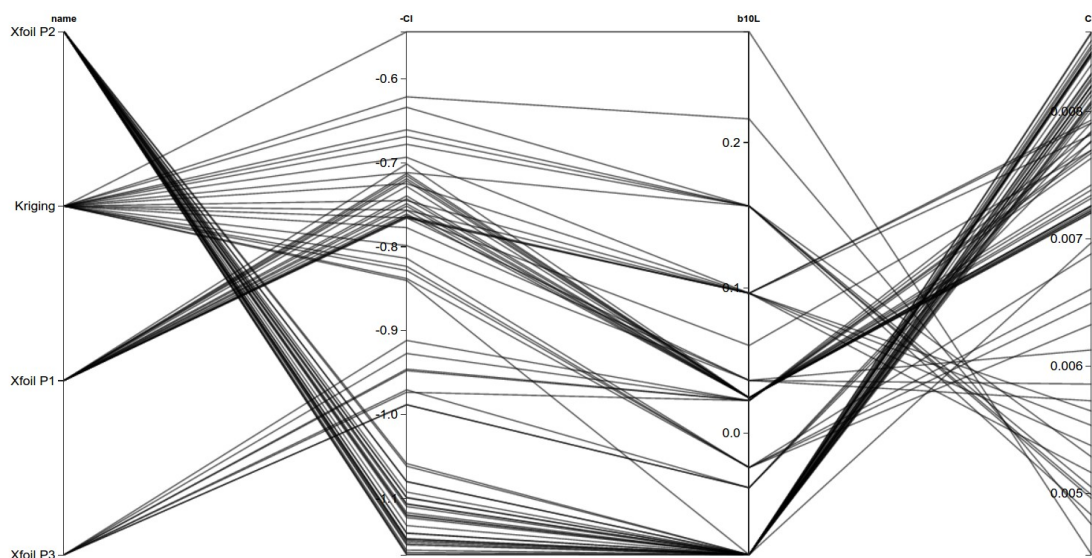


Figure 4.11: b_{1U} effect in Parallel Coordinates

- The component b_{10L} , which is the ante penultimate term of the design vector and refers to the lower surface, is proportionally related to the drag generated by the airfoil. A lower value means a lower C_d of the airfoil but also a lower lift generation. It has to do with the peak of the Trailing Edge: The higher this value, the more negative curvature has this zone hence more lift is generated but at the same time the drag increases due to the wake that is being produced. (Figure 4.12)

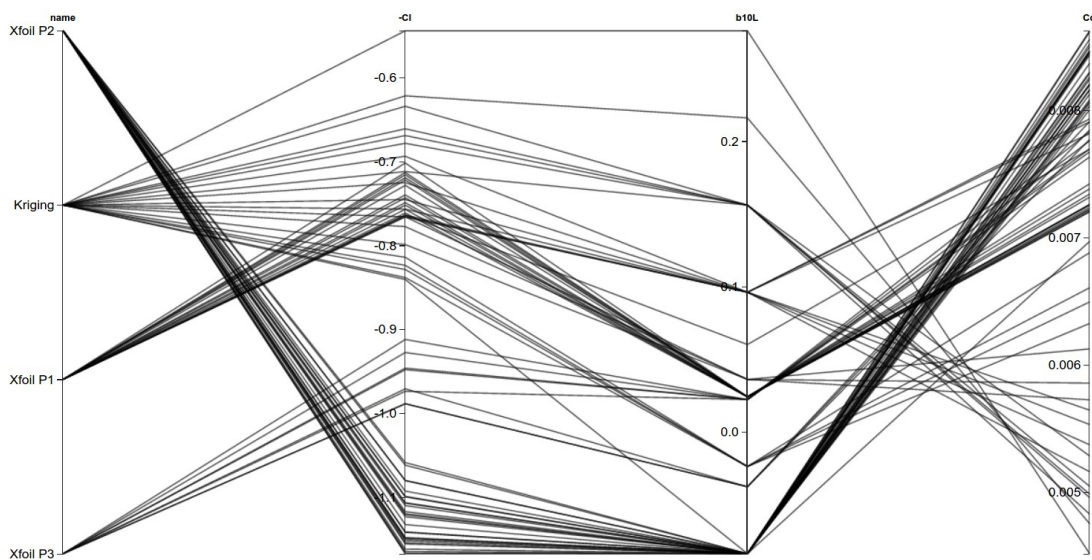


Figure 4.12: b_{10L} effect in Parallel Coordinates

- The term b_{9U} has also a great influence in the drag (Figure 4.13). The relationship is inverse, so a higher value the lower drag. This term is associated with the last part of the upper surface of the airfoil. If the value is high, it means that after the maximum chamber, the transition to the Leading edge is smooth, so there is not a big deceleration of the flow, as occurs in Figures 4.10b and 4.10d. On the other hand, if the value is small, the diffusion of the flow after the pressure peak is high to that a separation can occur and drag is more likely to be generated (Figure 4.10f).

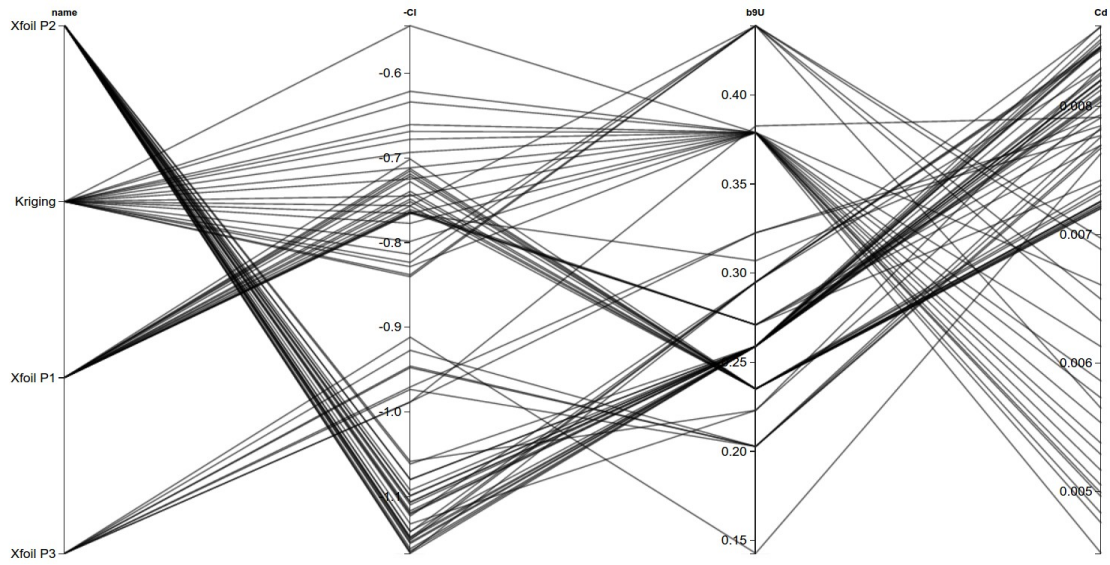


Figure 4.13: b_{9U} effect in Parallel Coordinates

Chapter 5

Conclusions and Further Work

Directions

5.1 Conclusions

The development of a new methodology initiated in previous work to map experimental data into the design space in order to build a surrogate model representing the aerodynamic performance of an airfoil has been continued and improved in this research project. Hence, the response surface model of the performance of a 2D airfoil for a specific flight condition was constructed based on reliable experimental data available in the state of the art. In addition, a new parametrisation technique, Class-Shape Transformations (CST), was implemented in order to map and include in the surrogate model the airfoils from which the experimental data was collected. Regarding the characteristics of the surrogate, two different methods were used (i.e. Radial Basis Functions and ordinary Kriging). Once the surrogates were conformed, a multiobjective surrogate-based optimisation was performed (i.e. minimising and maximising the drag and lift coefficients, respectively), having as starting point a

given datum airfoil for the flight condition considered.

In a similar fashion, a direct optimisation of the same datum airfoil was undertaken using Xfoil in order to compare the results with the surrogate-based approach. On the other hand, a small sample of optimum geometries was selected from the Pareto front of the surrogate-based optimal solutions and re-evaluated with Xfoil. After the Xfoil re-evaluation of those designs, a new smaller Pareto front of optimal solutions was found and some of them were used as starting points for a direct Xfoil optimisation, leading to a smaller number of evaluations needed to reach the solution provided by the full direct optimisation from the datum airfoil and validating thus the methodology proposed.

Thus, the main conclusions of this project are listed below:

- The optimisation methodology based on surrogate model and experimental data, alternative to a direct optimisation, proposed in this work has resulted to be effective in saving number of evaluations to reach the same type of solutions. The Pareto front of optimal solutions of the surrogate re-evaluated points have shown that follows the same trend as the Pareto front given by the direct optimisation. Furthermore, this process enables the search of potential optimal solutions obeying to particular C_l , C_d ranges by selecting a similar starting point from the above mentioned re-evaluated Pareto. Therefore, the optimisation process is faster and can be guided into the desired objective functions range values.
- The direct optimisation process have shown that achieving designs that reduce the drag coefficient is difficult for the given flight condition of low angle of attack and Mach number since the drag generated is already very low. Hence, the major improvement has been in lift coefficient, thing that was more feasible

to optimise. If the optimisations had been done for high angles of attack, there would have been an improvement in drag performance since for these flight conditions the drag is higher although the use of Xfoil in that case would not have been very precise. Conversely, the surrogate-based optimisation managed to produce improvements in the lift and drag behaviour for the flight condition but it has to be bore in mind that the surrogate is not representing the true physics of the problem, it just generates a response based on some other data and gives an estimation of possible starting points so that a direct optimisation with a more sophisticated program can be done.

- The surrogate-based optimisation have been showed to not be particularly influenced by the starting point. This independence of the datum condition can be harnessed in optimisation processes where the design space in which looking for optimal solutions is unknown. Thus, a random starting point can be chosen and the surrogate will still produce a response although the design is infeasible, leading after a given number of evaluations to an almost unique Pareto front of optimal solutions that would serve as a basement for later direct optimisations. In this sense, the surrogate model optimisation based on experimental data acts as a tool to steer the optimisation process if information about the starting conditions is unknown.
- Regarding to the airfoil mapping process, CST parametrisation technique has been shown to be really effective, accurate and fast. The L_2 norms between mapped and target airfoils have been very low for the order of parametrisation considered and only thicker airfoils of some families presented a higher (but still within acceptable levels) deviation. The NACA 6-series has been the family that has presented less error in the parametrisation, which means that

the CST parametrisation has been particularly effective in this kind of airfoils which represent the 80% of total mapped airfoils.

- The method considered for including the mapped airfoils in the surrogate has considered the deviation between the experimental and the Xfoil simulated mapped geometries coefficients, and the deviation between the Xfoil simulated raw and mapped geometries coefficients. This process has led to a better compromise solution when it came to take the decision than if only the deviation between the experimental and the mapped-simulated comparisons had been done because, in some cases, the experimental data seemed to be quite unrealistic. Moreover, although the method followed might seem a bit arbitrary, very often in real engineering processes, the experimental data that is available at hand does not take the form of a beautiful formatted table but it is in really old scanned charts or some part is even missing. Therefore, the approach followed in this thesis consisting of reading experimental data directly from charts is not that far from real life engineering.

5.2 Further Work suggestions

Even though the proposed methodology has been proved to be effective, still too much work is needed in order to improve the current research. Some guidelines and recommendations about future work in this field are given below:

- It is recommended to introduce uncertainty analysis in the process of reading the experimental data in order to have some statistical error distribution and propagation throughout the whole process so as to lead to a more robust surrogate model.

- It is suggested to introduce more values of angles of attack α in the surrogate model in order to widen and make the optimisation more robust under other flight conditions. In this work, the surrogate has been built for an unique flight mode, however, it should be extended to α near to stall conditions combining the available experimental data with the Xfoil results for this region. By doing this, a multi-fidelity surrogate can be built, being able to dynamically chose the proper data depending on the simulated flight conditions. For instance, under high angles of attack near to stall, the experimental data would prevail over Xfoil data.
- It is suggested that, apart from minimising/maximising the drag/lift coefficients, (C_d , C_l respectively), it would make a more realistic approach to also include the moment coefficient, C_m , as a constraint that has to remain as constant as possible during the optimisation process.
- It is recommended to improve the CST parametrisation with the CSRT refinement procedures [34] that would allow to include physical constraints in the optimisation process such as “boxes” inside the airfoil for structural or fuel purposes, or for instance including thickness specifications in the edge to be more realistic in terms of manufacturing tolerances.
- Although Xfoil performs very efficiently for incompressible flows and low α , is suggested to simulate the 2D optimal set of airfoils in a more sophisticated CFD software that includes turbulent models in order to analyse their behaviour at a higher angles of attack and Mach numbers.
- Due to the boom of 3D printer and Rapid Prototyping technologies in the last few years, it would be highly desirable to small-scale manufacture a selection

of optimal airfoils geometries from the Pareto front of optimal designs and test them in a wind tunnel in order to compare their performance with the surrogate/CFD predicted coefficients and include the measured data into the surrogate model.

- It is encouraged to research into other multivariable optimisation post-analysis techniques such as the use of Self-Organising Maps (SOM).

Bibliography

- [1] Ira H Abbott, Albert E von Doenhoff, and Louis S Stivers. Summary of airfoil data. Technical Report 824, 1945.
- [2] Andoni Agirre-Mentxaka and Timoleon Kipouros. *An aerospace application of a data directed design analysis system. [electronic resource]*. Theses 2015. 2015.
- [3] A.Korobenko, M.Pigazzini, V.Singh, H.Kim, D. Allaire, K.Willcox, A.L. Marsden, and Y.Bazilevs. Dynamic-data-driven damage prediction in aerospace composite structures. *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Washington, D.C.*, 2016.
- [4] Y. Bazilevs, M.-C. Hsu, , and M. T. Bement. Adjoint-based control of fluid-structure interaction for computational steering applications. *Procedia Computer Science*, 18:1989–1998, 2013.
- [5] Y. Bazilevs, A.L. Marsden, F. Lanza di Scalea, A. Majumdar, and M. Tatineni. Toward a computational steering framework for large-scale composite structures based on continually and dynamically injected sensor data. *Procedia Computer Science*, 9:1149–1158, 2012.
- [6] J Bollinger, J Britton, N Gisin, P Knight, P Kwiat, and I Percival. Autonomic healing of polymer composites. 409(February):1804–1807, 2001.

- [7] Marco Ceze, Marcelo Hayashi, and Ernani Volpe. A Study of the CST Parameterization Characteristics. *27th AIAA Applied Aerodynamics Conference*, (June), 2009.
- [8] I. Clark. *Practical Geostatistics*. Applied Science Pub., 1979.
- [9] MISTI Cooperative Autonomous Observing Systems (CAOS). Research supported in part by AFOSR DDDAS Program, Lincoln Laboratory and NSF. <http://caos.mit.edu/>, 2014.
- [10] Frederica Darema. Dynamic Data Driven Applications Systems: A New Paradigm for Application Simulations and Measurements. *Computational Science - ICCS 2004: 4th International Conference*, 3038:662–669, 2004.
- [11] K Deb, S Pratab, S Agarwal, and T Meyarivan. A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computing*, 6(2):182–197, 2002.
- [12] C. C. Douglas. <http://www.1dddas.org>. 2000–2015.
- [13] C. C. Douglas, V. Calo, D. C. Cerwinsky, L. Deng, , and Y. Efendiev. Using shape memory alloys: a dynamic data driven approach. *Procedia Computer Science*, 18:1844–1850, 2013.
- [14] Craig C. Douglas, Yalchin Efendiev, Peter Popov, and Victor M. Calo. An introduction to a porous shape memory alloy dynamic data driven application system. *Procedia Computer Science*, 9:1081–1089, 2012.
- [15] M. Drela. <http://web.mit.edu/drela/public/web/xfoil>, 2000.

- [16] DDDAS Workshop Dynamic Data Driven Applications Systems (DDDAS) Website. National Science Foundation. <http://www.dddas.org/nsfworkshop2000.html>, 2000.
- [17] C. Farhat, J.G. Michopoulos, F.K. Chang, L.J. Guibas, and A.J. Lew. Towards a dynamic data driven system for structural and material health monitoring. In V.N. Alexandrov, G.D. van Albada, P.M.A. Slood, and J.J. Dongarra, editors, *Computational Science - ICCS 2006: 6th International Conference, Reading, UK, May 28-31, 2006, Proceedings, Part III*, volume 3993 of *Lecture Notes in Computer Science*, pages 456–464, Heidelberg, 2006. Springer-Verlag.
- [18] Julian Heinrich. <http://www.parallelcoordinates.de/>, 2016.
- [19] Robert Hooke and T. A. Jeeves. "Direct Search" Solution of Numerical and Statistical Problems. *Journal of the ACM*, 8(2):212–229, 1961.
- [20] Alfred Inselberg. Parallel coordinates: Visual, multidimensional geometry and its applications. *Springer*, 2009.
- [21] Alfred Inselberg. Lecture Notes PARALLEL COORDINATES. *Senior Fellow San Diego SuperComputing Center and School of Mathematical Sciences Tel Aviv University, Israel*, 1999.
- [22] D M Jaeggi, G T Parks, T Kipouros, and P J Clarkson. The development of a multi-objective Tabu Search algorithm for continuous optimisation problems. *European Journal of Operational Research*, 185(3):1192–1212, 2008.
- [23] D Knight, Q Ma, T Rossman, and Y Jaluria. Evaluation of fluid-thermal systems by dynamic data driven application systems - Part II. *Computational Science - ICCS 2007, Pt 1, Proceedings*, 4487:1189–1196, 2007.

- [24] Doyle D Knight. Data Driven Design Optimization Methodology: A Dynamic Data Driven Application System. *International Conference on Computational Science*, 2660:329–336, 2003.
- [25] Brenda M Kulfan, John E Bussoletti, Boeing Commercial, and Airplane Group. Fundamental Parametric Geometry Representations for Aircraft Component Shapes. 2006.
- [26] Brenda M Kulfan and Reno Hilton. A Universal Parametric Geometry Representation Method – CST – A Universal Parametric Geometry Representation Method – CST –. *Aiaa-07-0062*, pages 1–36, 2007.
- [27] Gregory R. Madey, M. Brian Blake, Christian Poellabauer, Hongsheng Lu, R. Ryan McCune, and Yi Wei. Applying DDDAS principles to command control and mission planning for UAV swarms. *Procedia Computer Science*, 9:1177–1186, 2012.
- [28] Robert McCune and Greg Madey. Control of artificial swarms with DDDAS. *Procedia Computer Science*, 29:1171–1181, 2014.
- [29] Ying Ouyang, Jia En Zhang, and Shi Ming Luo. Dynamic data driven application system: Recent development and future perspective. 4:1–8, 2006.
- [30] E.E. Prudencio, P. T. Bauman, D. Faghihi, , J. T. Oden, K. Ravi-Chandar, , and S. V. Williams. A dynamic data driven application system for real-time monitoring of stochastic damage. *Procedia Computer Science*, 18:2056–2065, 2013.
- [31] Jonathon Shlens, Mountain View, and I Introduction. A Tutorial on Principal Component Analysis. 2014.








- [32] UIUC Airfoil Data Site. <http://airfoiltools.com/>, 2016.
- [33] Vaclav Skala. Radial Basis Functions for High-Dimensional Visualization. (c):193–198, 2012.
- [34] Michiel H Straathof. *Shape Parameterization in Aircraft Design: A Novel Method, Based on B-Splines*.
- [35] T. Kipouros T. C. Peachey and Monash University. Nimrod/O Users’ Guide. 2012.
- [36] REN R 440/ RENR 501 – Disturbance Ecology Course University of Alberta. <https://sites.ualberta.ca/~haitao2/rejr501/>, 2010.
- [37] Y. Wei, G. Madey, , and M. B. Blake. An operation-time simulation framework for uav swarm configuration and mission planning. *Procedia Computer Science*, 18:1949–1958, 2013.
- [38] Karen Willcox. Afri-osr-va-tr-2015-0127 dynamic data driven methods for self-aware aerospace vehicles. 2015.
- [39] H. Zhao, T. Icoz, Y. Jaluria, and D. Knight. Application of data-driven design optimization methodology to a multi-objective design optimization problem. *Journal of Engineering Design*, 18(4):343–359, 2007.
- [40] H Zhao, D Knight, E Taskinoglu, and V Jovanovic. Data Driven Design Optimization Methodology Development and Application. pages 748–755, 2004.

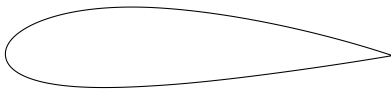
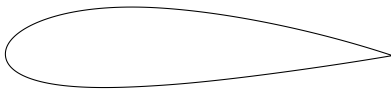


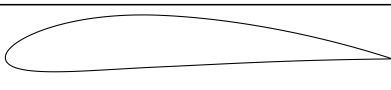




Appendix A

Mapped Airfoils



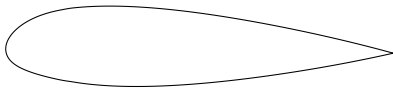

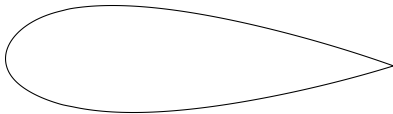
A.1 Available Data Airfoils

A.1.1 NACA 4-Digits




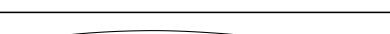





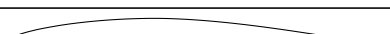
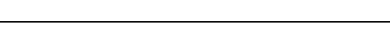
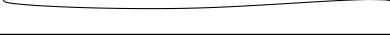




Experimental Data			
Airfoil	C_l	C_d	
NACA0006	0.30	0.0055	
NACA0009	0.35	0.0060	
NACA1408	0.65	0.0075	
NACA1410	0.45	0.0060	
NACA1412	0.45	0.0065	
NACA2412	0.55	0.00650	
NACA2415	0.50	0.00675	

















Airfoil	Experimental Data		
	C_l	C_d	
NACA2418	0.55	0.00750	
NACA2421	0.45	0.0080	
NACA2424	0.45	0.0090	
NACA4412	0.75	0.0070	
NACA4415	0.75	0.0075	
NACA4418	0.70	0.0080	
NACA4421	0.65	0.0090	
NACA4424	0.70	0.0100	

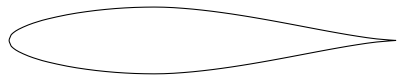
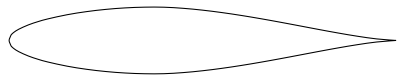


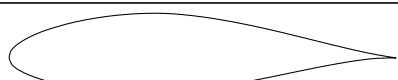

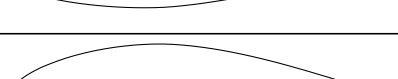
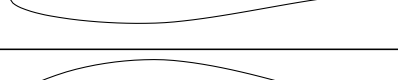
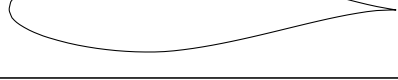

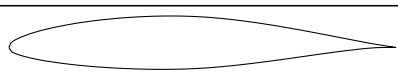





A.1.2 NACA 5-Digits

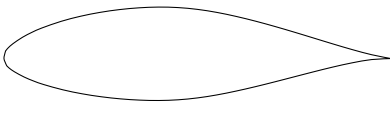
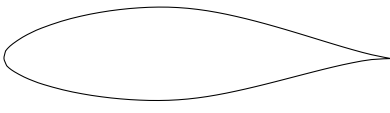
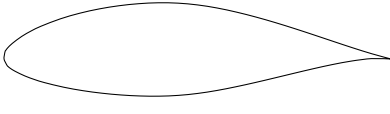
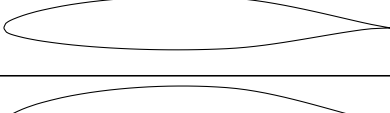
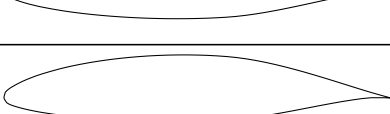
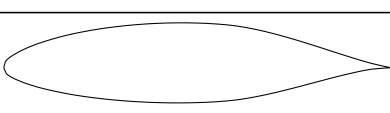
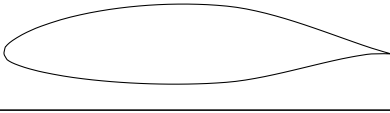
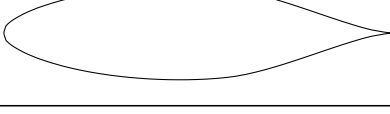
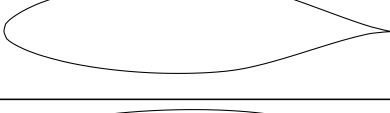
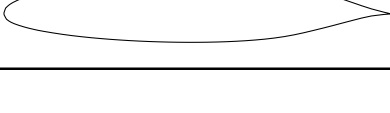

Experimental Data			
Airfoil	C_l	C_d	
NACA23012	0.45	0.0065	
NACA23015	0.45	0.0075	
NACA23018	0.45	0.0075	
NACA23021	0.45	0.0085	
NACA23024	0.40	0.0095	







A.1.3 NACA 6-Series

Experimental Data			
Airfoil	C_l	C_d	
NACA63206	0.50	0.0065	
NACA63209	0.50	0.00675	
NACA63210	0.50	0.0060	
NACA64108	0.35	0.0065	
NACA64110	0.45	0.0070	
NACA64206	0.45	0.00675	
NACA64208	0.50	0.0045	
NACA64209	0.50	0.00675	
NACA64210	0.50	0.0070	
NACA65206	0.475	0.0065	
NACA65209	0.475	0.0075	
NACA65210	0.50	0.0070	
NACA65410	0.60	0.0045	
NACA66206	0.50	0.0070	
NACA66209	0.45	0.0070	
NACA66210	0.425	0.0060	



Airfoil	Experimental Data		
	C_l	C_d	
NACA63(1)212	0.550	0.0055	
NACA63(1)412	0.675	0.0055	
NACA63(2)015	0.350	0.0055	
NACA63(2)215	0.500	0.0060	
NACA63(2)415	0.700	0.0060	
NACA63(2)615	0.800	0.00625	
NACA63(3)218	0.550	0.0060	
NACA63(3)418	0.675	0.00675	
NACA63(3)618	0.825	0.00675	
NACA63(4)221	0.525	0.00675	
NACA63(4)421	0.700	0.0070	
NACA64(1)012	0.350	0.0070	
NACA64(1)112	0.450	0.00625	
NACA64(1)212	0.475	0.00525	
NACA64(1)412	0.650	0.0055	

Airfoil	Experimental Data		
	C_l	C_d	
NACA64(2)015	0.350	0.0060	
NACA64(2)215	0.500	0.00525	
NACA64(2)415	0.700	0.00575	
NACA64(3)218	0.500	0.00575	
NACA64(3)418	0.700	0.0065	
NACA64(3)618	0.775	0.0060	
NACA64(4)221	0.525	0.0065	
NACA64(4)421	0.675	0.00675	
NACA65(1)212	0.475	0.0060	
NACA65(1)412	0.675	0.0050	
NACA65(2)215	0.500	0.0050	
NACA65(2)415	0.650	0.00525	
NACA65(3)218	0.450	0.0055	
NACA65(3)418	0.625	0.0055	
NACA65(3)618	0.850	0.00675	

Airfoil	Experimental Data		
	C_l	C_d	
NACA65(4)221	0.500	0.00575	
NACA65(4)421	0.700	0.0065	
NACA66(1)212	0.450	0.0060	
NACA66(2)215	0.470	0.0045	
NACA66(2)415	0.550	0.0045	
NACA66(3)218	0.500	0.00475	
NACA66(3)418	0.600	0.0050	
NACA66(4)021	0.275	0.00525	
NACA66(4)221	0.475	0.00525	
NACA67(1)215	0.450	0.0055	

Airfoil	Experimental Data		
	C_l	C_d	
NACA65(1)212(a06)	0.45	0.0045	
NACA65(2)415(a05)	0.60	0.0055	
NACA65(3)418(a05)	0.65	0.0055	
NACA65(3)618(a05)	0.75	0.0060	
NACA65(4)421(a05)	0.65	0.0060	

A.1.4 NACA 7-Series

Experimental Data			
Airfoil	C_l	C_d	
NACA747A315	0.475	0.00525	
NACA747A415	0.550	0.0055	

Appendix B

Vector \mathbf{b} - Mapping CST

MATLAB function

```

function [ B_upper, B_lower,aa ] = cst( A , B_orders )
%cst function: input arguments
%A: airfoil data, name and points
%B_orders: Bernstein Polynomials orders (for upper and lower surface)
%Returns B_upper and B_lower, coefficients to built the airfoil
% Detailed explanation goes here

%vectors where profile data is going to be stored, for upper and lower
%surfaces
target_upper = [];
target_lower = [];

    flag = 0; %to switch to lower surface points

    temporal = 0;
    aa = 0;
for i=1:length(A.data)
    if flag == 1
        target_lower = [target_lower; A.data(i,:)];
    else
        target_upper = [target_upper; A.data(i,:)];
        if A.data(i,1) < 0 && A.data(i,1)<temporal
            aa = A.data(i,1);
        end
    end
    if A.data(i,1)==0 && A.data(i,2)==0
        flag = 1;
        target_lower = [target_lower; A.data(i,:)];
    end
end

%plot(target_lower(:,1),target_lower(:,2),'k',target_upper(:,1),target_upper(:,2),'k')
xlabel('z/c');
ylabel('x/c');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
L2_norms_upper = [];
L2_norms_lower = [];
condition_numbers_upper = [];
condition_numbers_lower = [];

orders = [];
%for ii=10:10
n_upper = B_orders(1); %ORDER OF BENSTEIN POLYNOMIALS (upper surface)
n_lower = B_orders(2); %ORDER OF BENSTEIN POLYNOMIALS (lower surface)

%PARAMETERS OF DESIGN
B_upper = ones(1,n_upper+1); %PARAMETERS OF DESIGN upper surface
B_lower = ones(1,n_lower+1); %PARAMETERS OF DESIGN lower surface

%Class function (0.5 and 1 for 2D airfoils)% --> NACA type round nose and pointed aft end
N1 = 0.5;
N2 = 1;

datum_lower = [ ];
datum_upper = [ ];
%TE thickness (dz/c)
dzc_lower = 0;
dzc_upper = 0;

    M_upper = [ ];
    M_lower = [ ];

%% UPPER SURFACE %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%loop for each x/c point
for i = 1: length(target_upper)
    % x/c
    datum_upper(i,1) = target_upper(i,1);

```

```

%loop for the sum of all Bernstein Polynomials evaluated at the point
%of study
for r=0:n_upper
    M_upper(i,r+1) = real(datum_upper(i,1)^N1*(1-datum_upper(i,1))^N2) *...
        nchoosek(n_upper,r)*(1-target_upper(i,1))^(n_upper-r)*target_upper(i,1)^r;
end
end

% % x/c*dzc_upper

datum_upper(:,2) = M_upper*B_upper' + datum_upper(:,1)*dzc_upper;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% LOWER SURFACE %%%%%%%%%
%loop for each x/c point
for i = 1: length(target_lower)
    % % x/c
    datum_lower(i,1) = target_lower(i,1);
    %loop for the sum of all Bernstein Polynomials evaluated at the point
    %of study
    for r=0:n_lower
        M_lower(i,r+1) = -real(datum_lower(i,1)^N1 *(1-datum_lower(i,1))^N2)...
            * nchoosek(n_lower,r)*(1-target_lower(i,1))^(n_lower-r)*target_lower(i,1)^r;
    end
end
% % x/c*dzc_upper

datum_lower(:,2) = M_lower*B_lower' + datum_lower(:,1)*dzc_lower;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%hold on;
%plot(datum_lower(:,1),datum_lower(:,2),datum_upper(:,1),datum_upper(:,2))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% RESULTS %%%%%%%%%55
B_upper = M_upper\(target_upper(:,2)-datum_upper(:,1)*dzc_upper);
B_lower = M_lower\(target_lower(:,2)-datum_lower(:,1)*dzc_lower);

```


Appendix C

CST Point writer MATLAB function

```

function [ output_args ] = cst_point_writer(B,N1, N2, dzc_upper, dzc_lower, B_orders,num)
%this function writes the data geometry of the airfoil from the new_b.txt
%that contains the coefficients B computes the airfoil geometry
%coefficients input
%example cst_point_writer( 'new_b.txt', 0.5,1,0,0,[5,5])

%%LOOP FOR DIVIDING THE INPUT B INTO B_UPPER AND B_LOWER
upper = 1;
cont_up = 0;
cont_low = 0;
for i=1:length(B)
    if upper ==1
        cont_up = cont_up+1;
        B_upper(cont_up) = B(i);
        upper =0;
    else
        cont_low = cont_low+1;
        B_lower(cont_low) = B(i);
        upper = 1;
    end
end

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    n_lower = B_orders(2);
    n_upper = B_orders(1);
x
        xc_up = [cosinspace(0,0.05,15), cosinspace(0.05,0.95,45), cosinspace(0.95,1,15)]; %
generates cosinus distribution, more points at LE and TE
        xc_lo = [cosinspace(0 ,0.05,15), cosinspace(0.05,0.95,45), cosinspace(0.95,1,15) ]';

        save('xc_up.txt','xc_up','-ascii');
        save('xc_lo.txt','xc_lo','-ascii');
        for i = 1: length(xc_lo)

            %%% LOWER SURFACE %%%
            for r=0:n_lower
                M_lower(i,r+1) = -real(xc_lo(i,1)^N1 *(1-xc_lo(i,1))^N2)...
                    * nchoosek(n_lower,r)*(1-xc_lo(i,1))^(n_lower-r)*xc_lo(i,1)^r;
            end

        end

        for i = 1: length(xc_up)

            %%% UPPER SURFACE %%%
            for r=0:n_upper
                M_upper(i,r+1) = real(xc_up(i,1)^N1 *(1-xc_up(i,1))^N2)...
                    * nchoosek(n_upper,r)*(1-xc_up(i,1))^(n_upper-r)*xc_up(i,1)^r;
            end
        end

        zc_lower(:,1) = M_lower*B_lower' + xc_lo*dzc_lower;
        zc_upper(:,1) = M_upper*B_upper' + xc_up*dzc_upper;

%XFOIL REQUIREMENTS, start from TE to LE by upper surface and returning to
%TE by lower surface
%The points must be in (x,y) pairs, starting at the trailing edge (TE), going to the leading edge
(LE), and back to the TE. The points may go over the upper surface and back along the lower surface,
or vice versa (the code can figure that out).
        data = [flipud(xc_up          ), flipud(zc_upper          );
                xc_lo(2:end) ,          zc_lower(2:end)]; %flipud is for plotting purposes (avoiding
midline), first point removed to avoid repetitions at 0 0
        plot(data(:,1),data(:,2),'-'); hold on
        %plot(A.data(:,1),A.data(:,2),'r') %for comparison with real NACA coordinate data
        axis([0 1 -0.5 0.5])
        % filepath2 = 'profile';

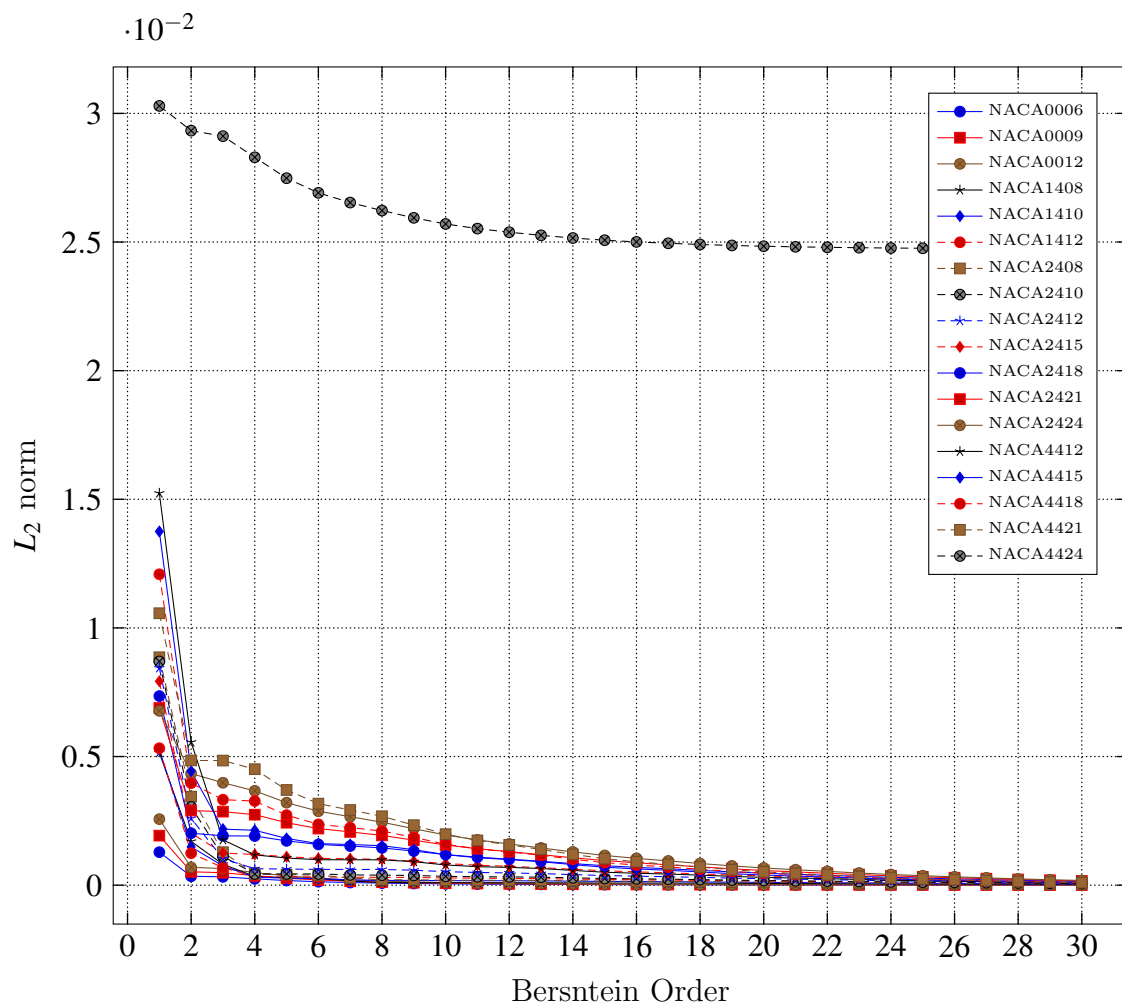
        name=['airfoil_',num2str(num),'.dat'];
        save(name,'data','-ascii');
end

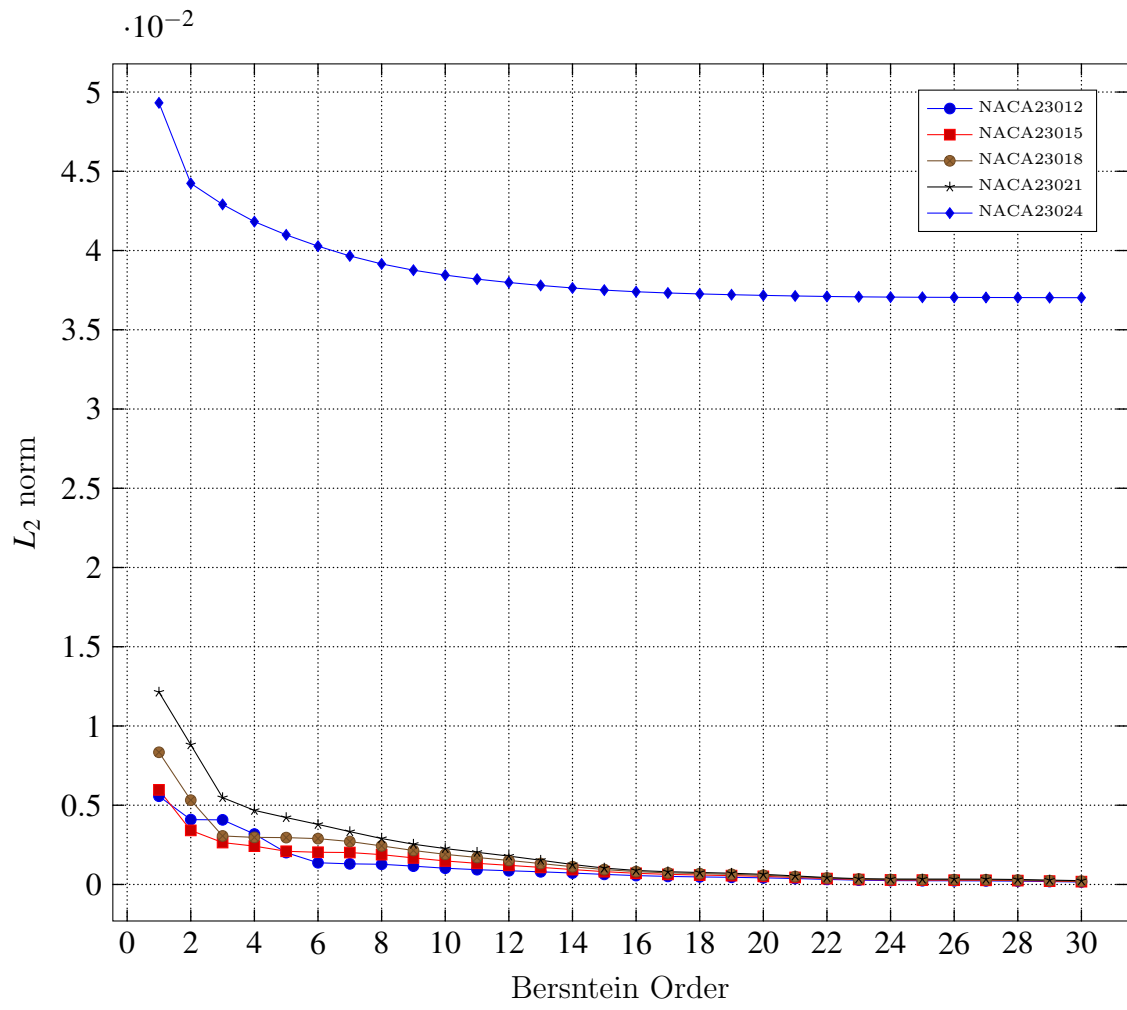
```

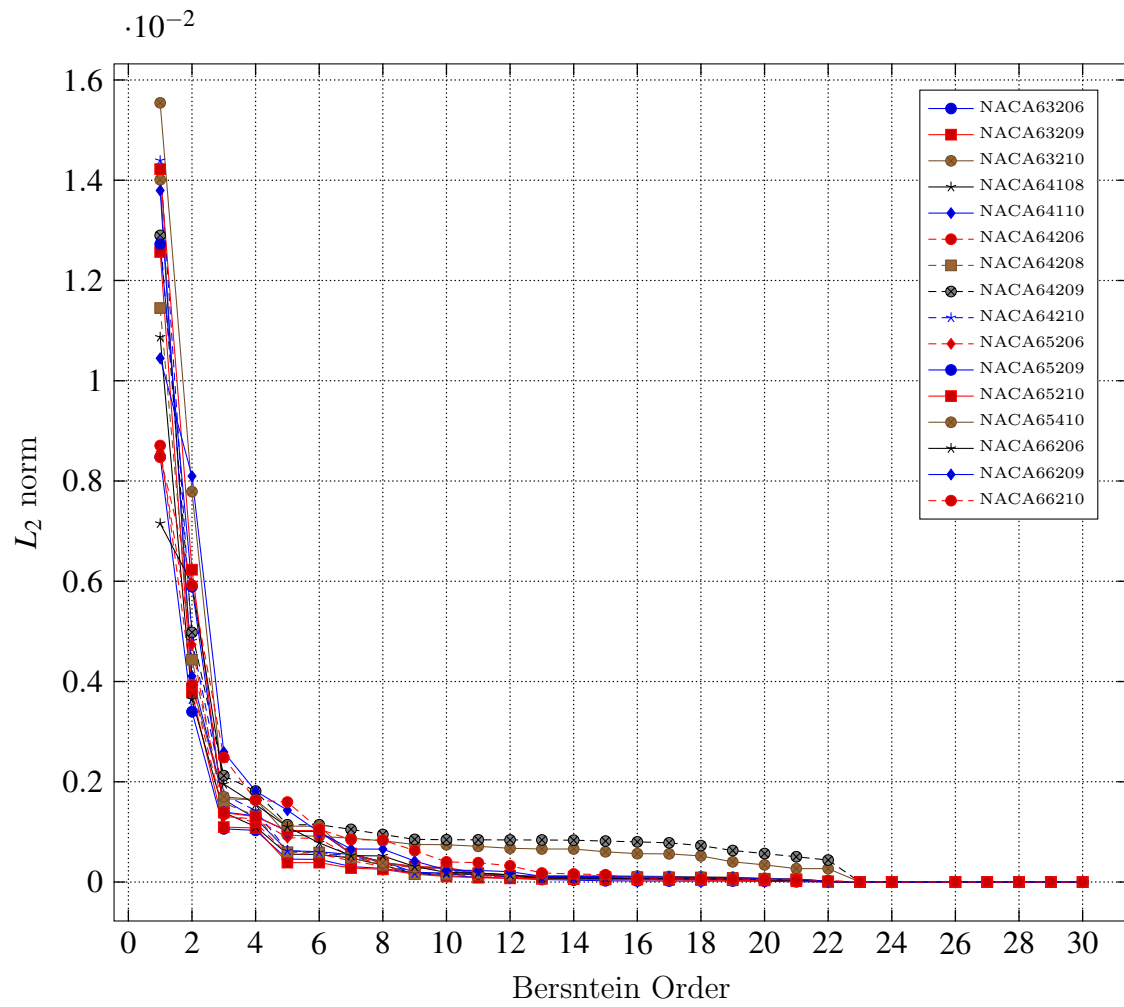

Appendix D

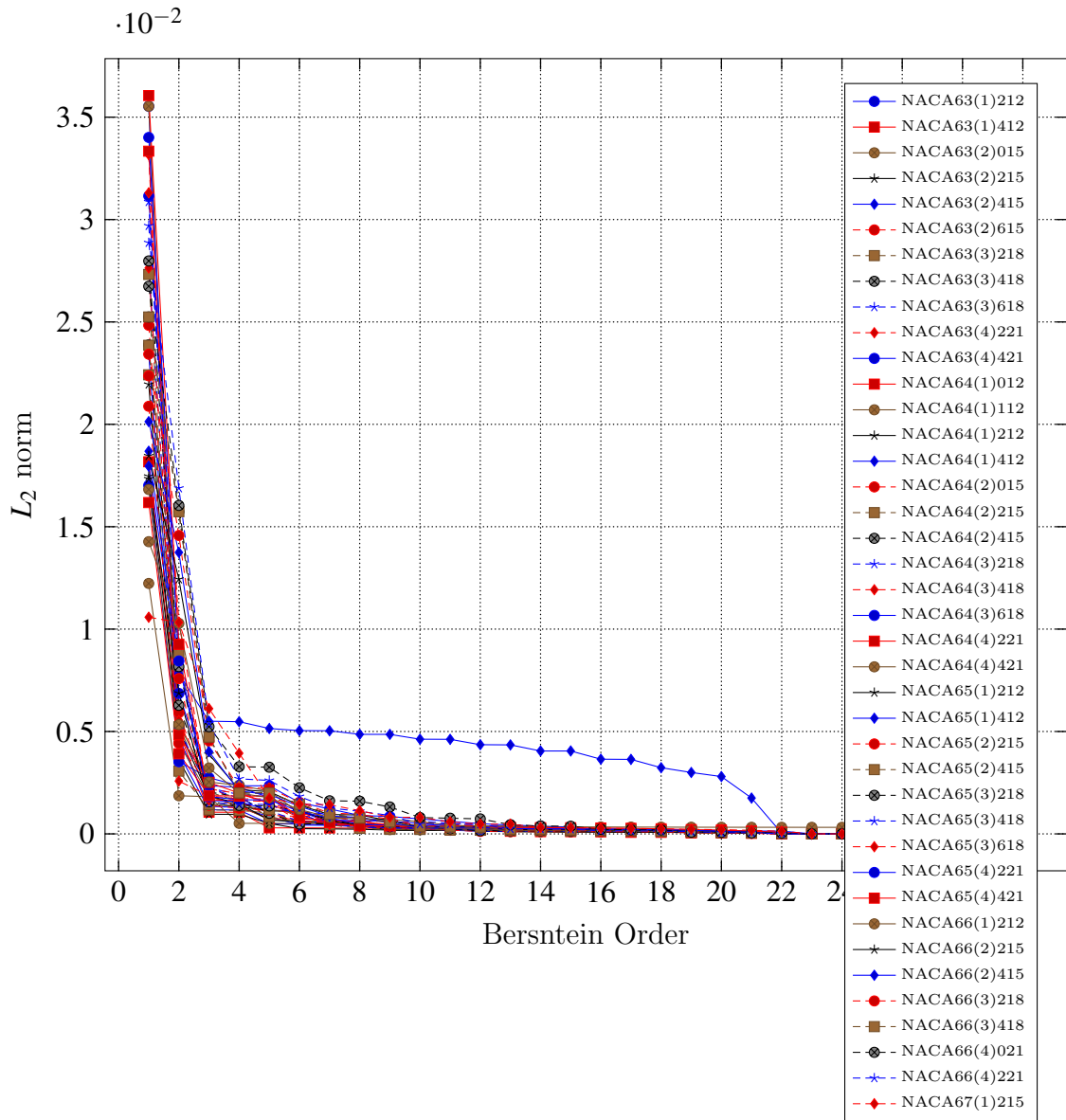
L_2 norms of Mapped Airfoils

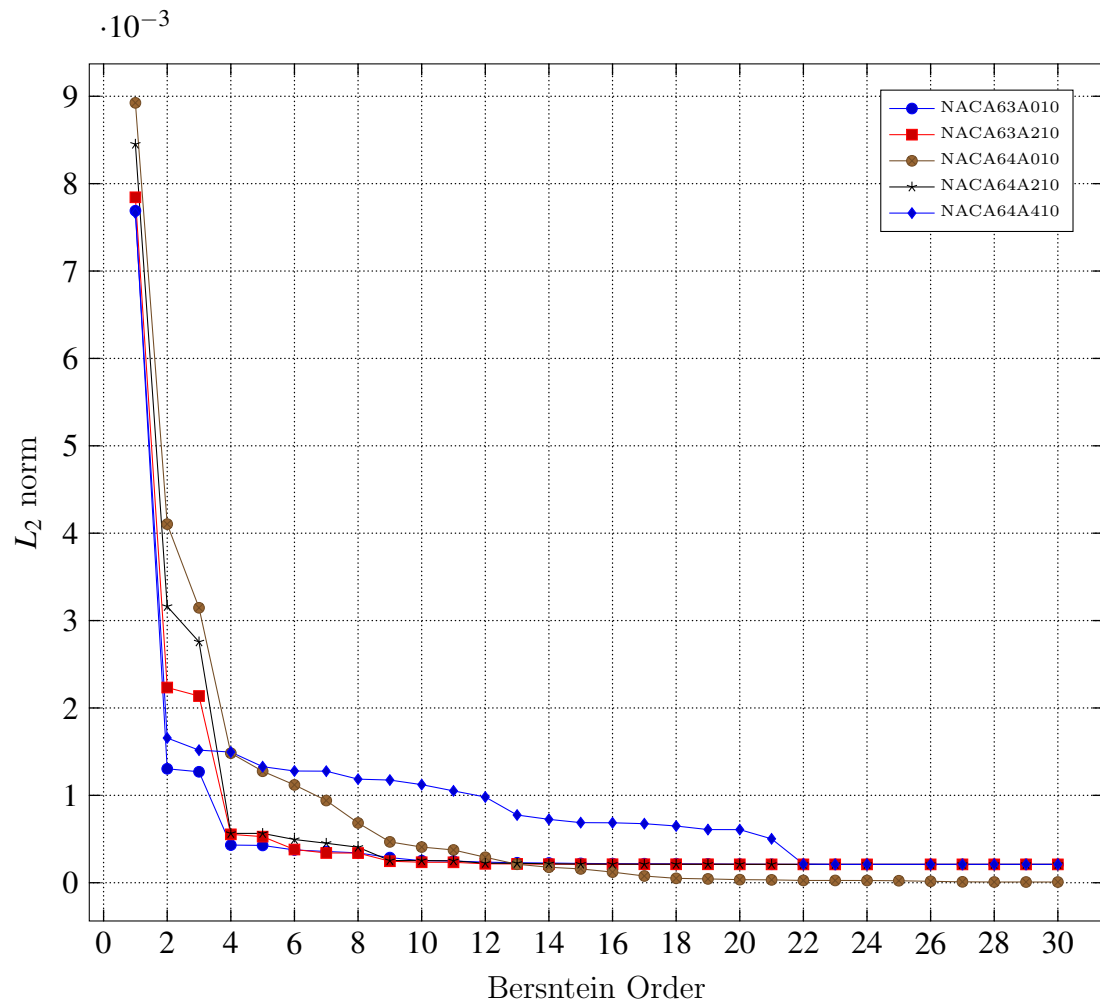
D.1 Upper Surfaces

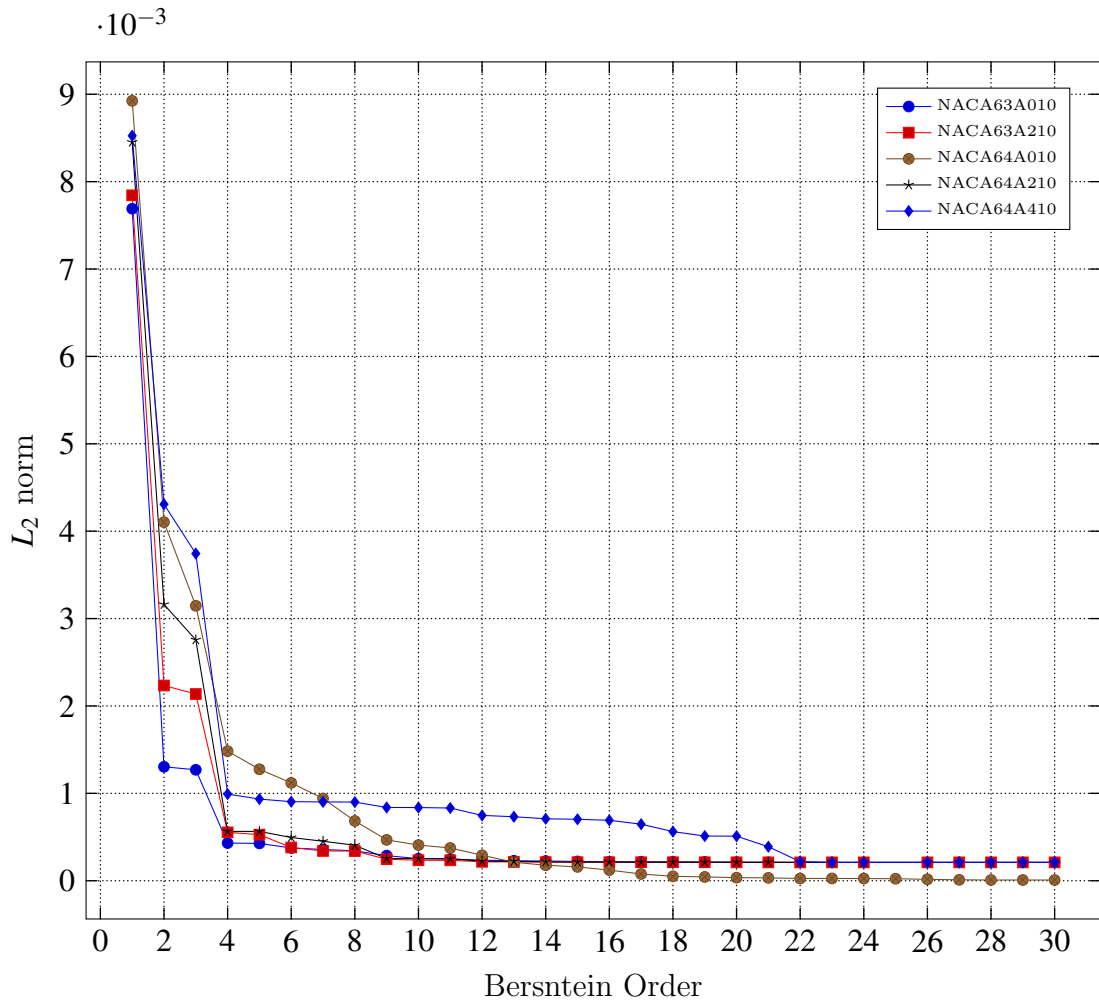


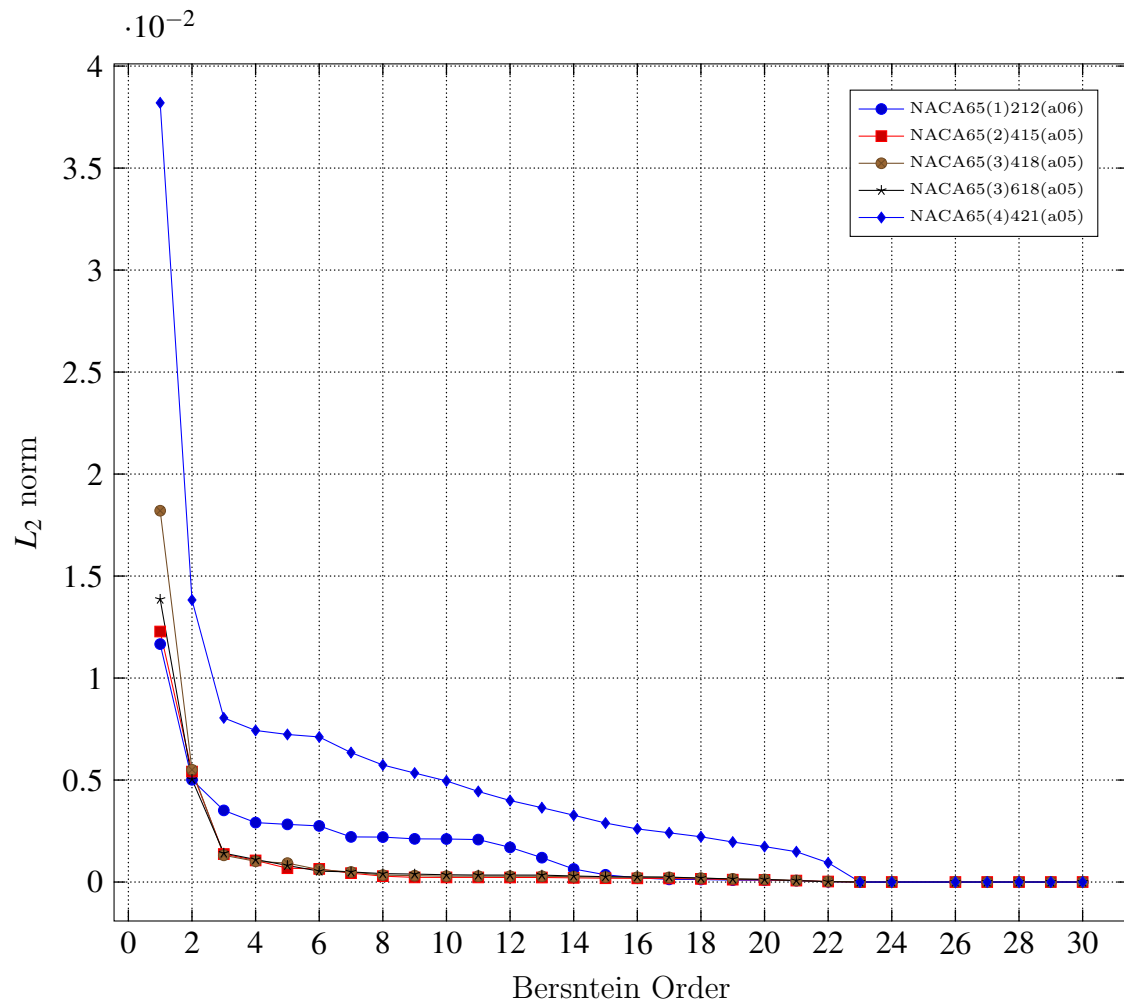


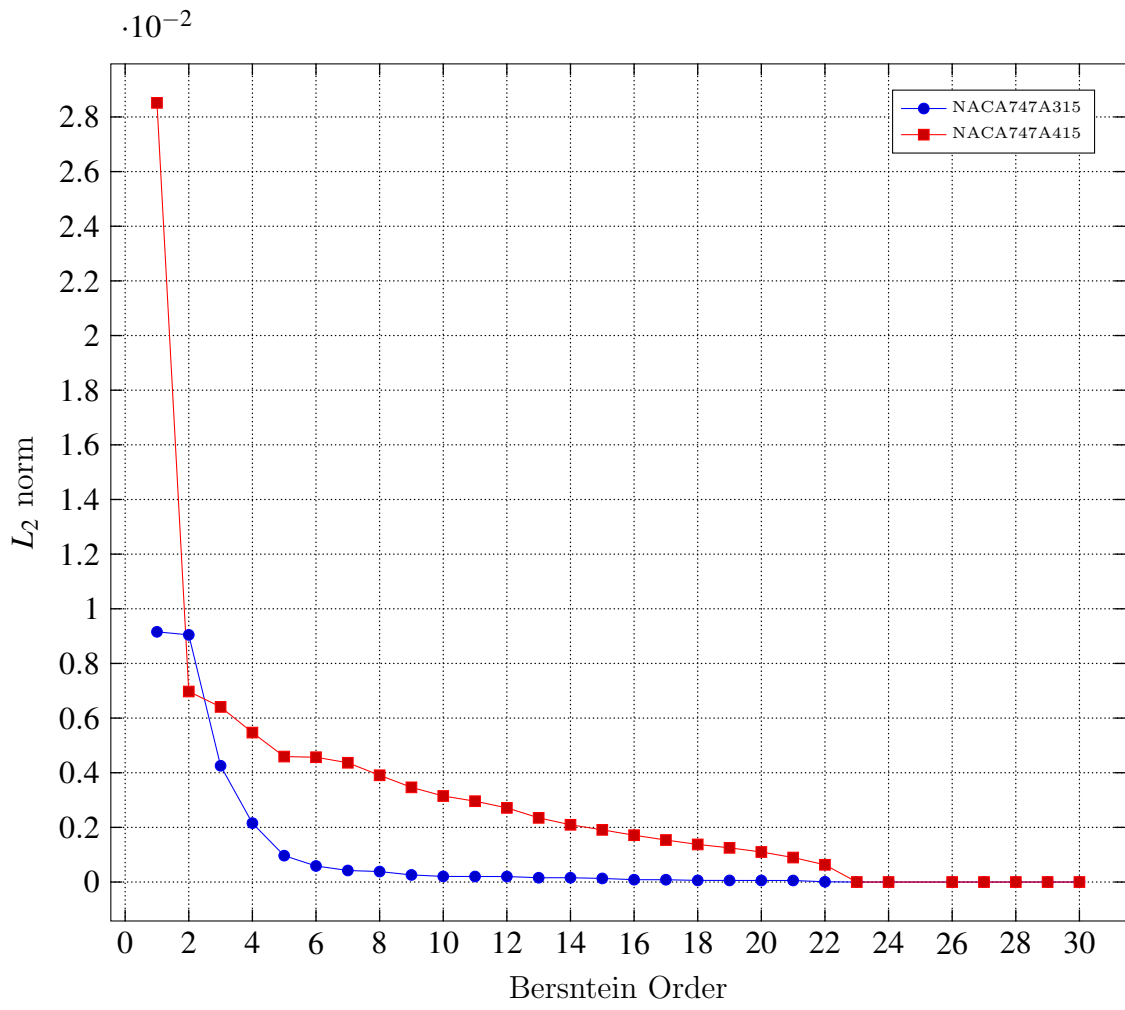




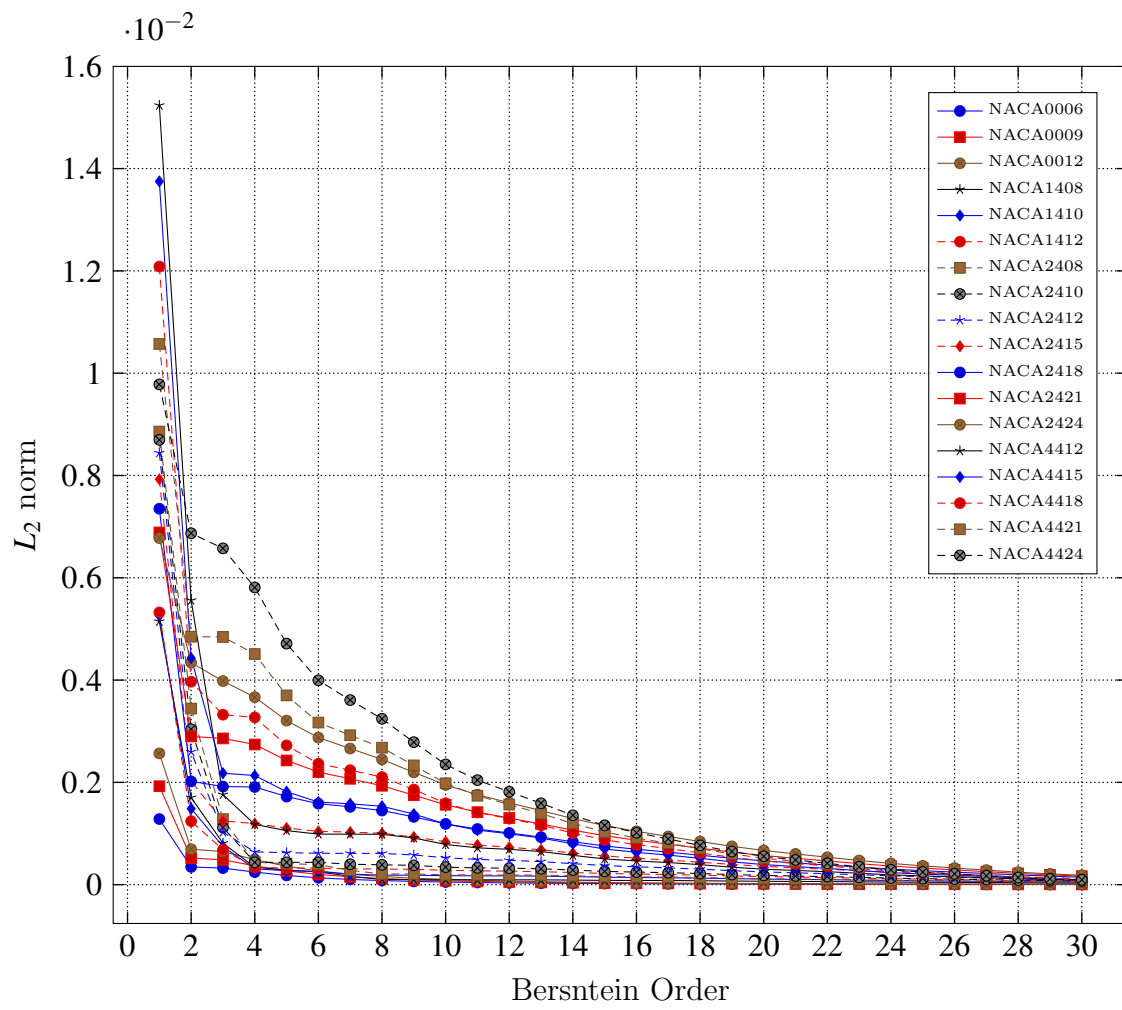


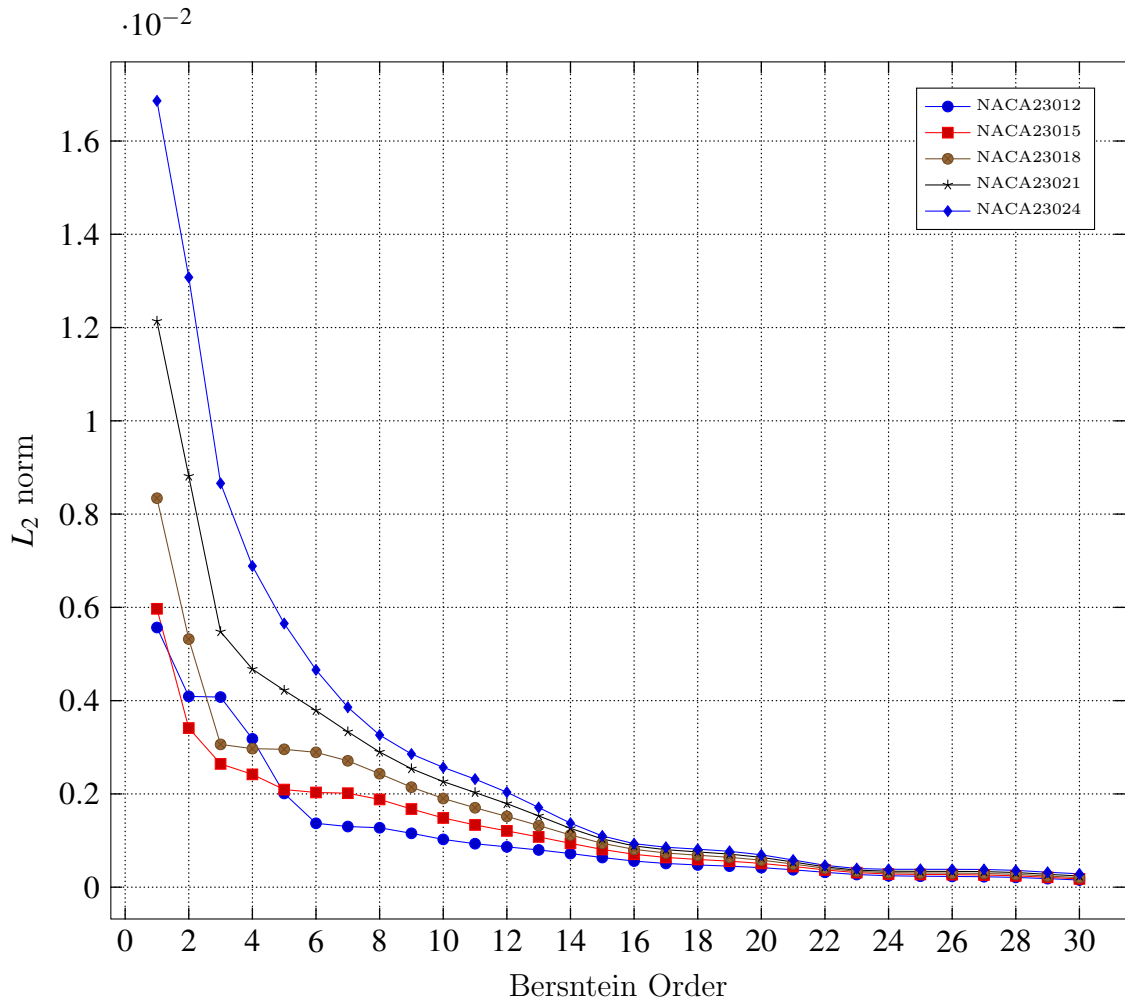


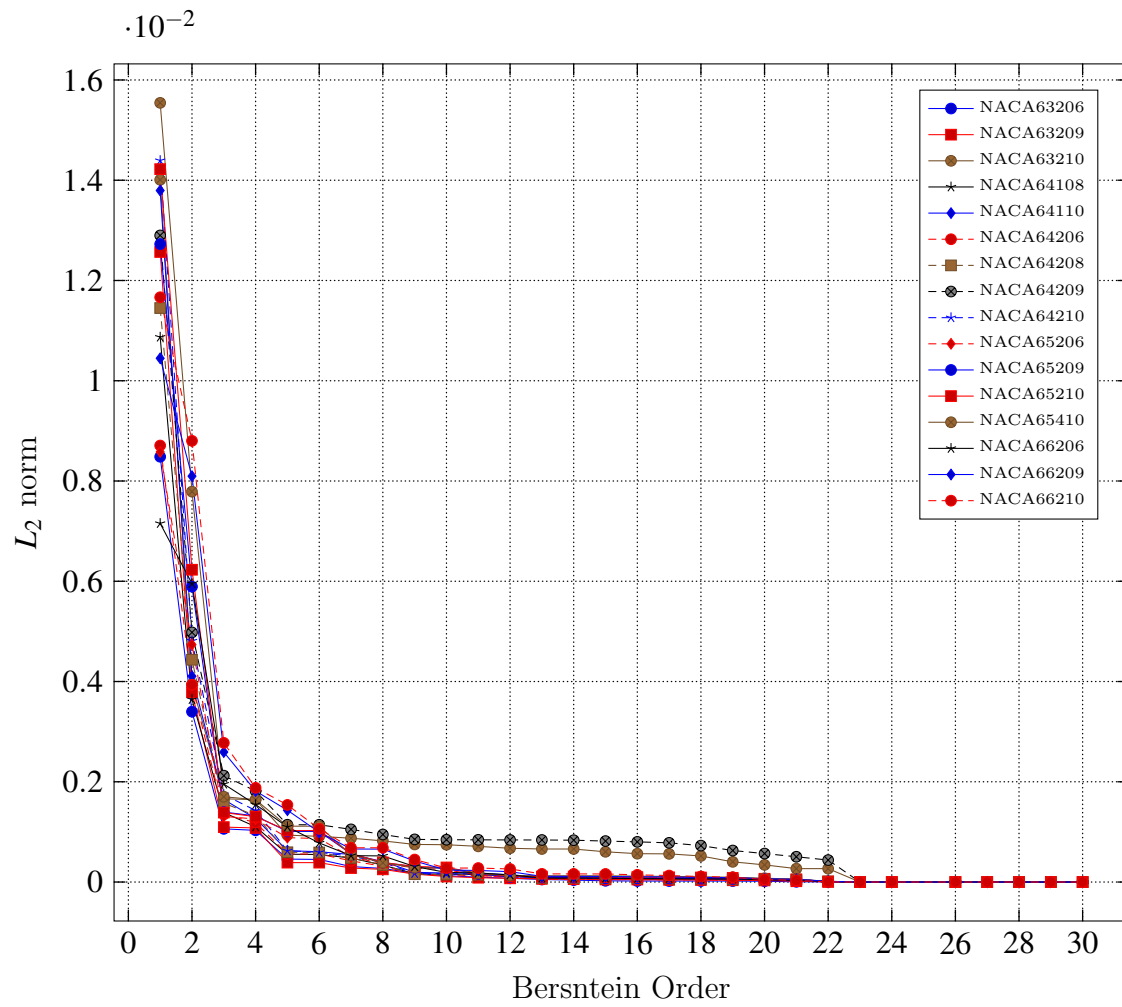


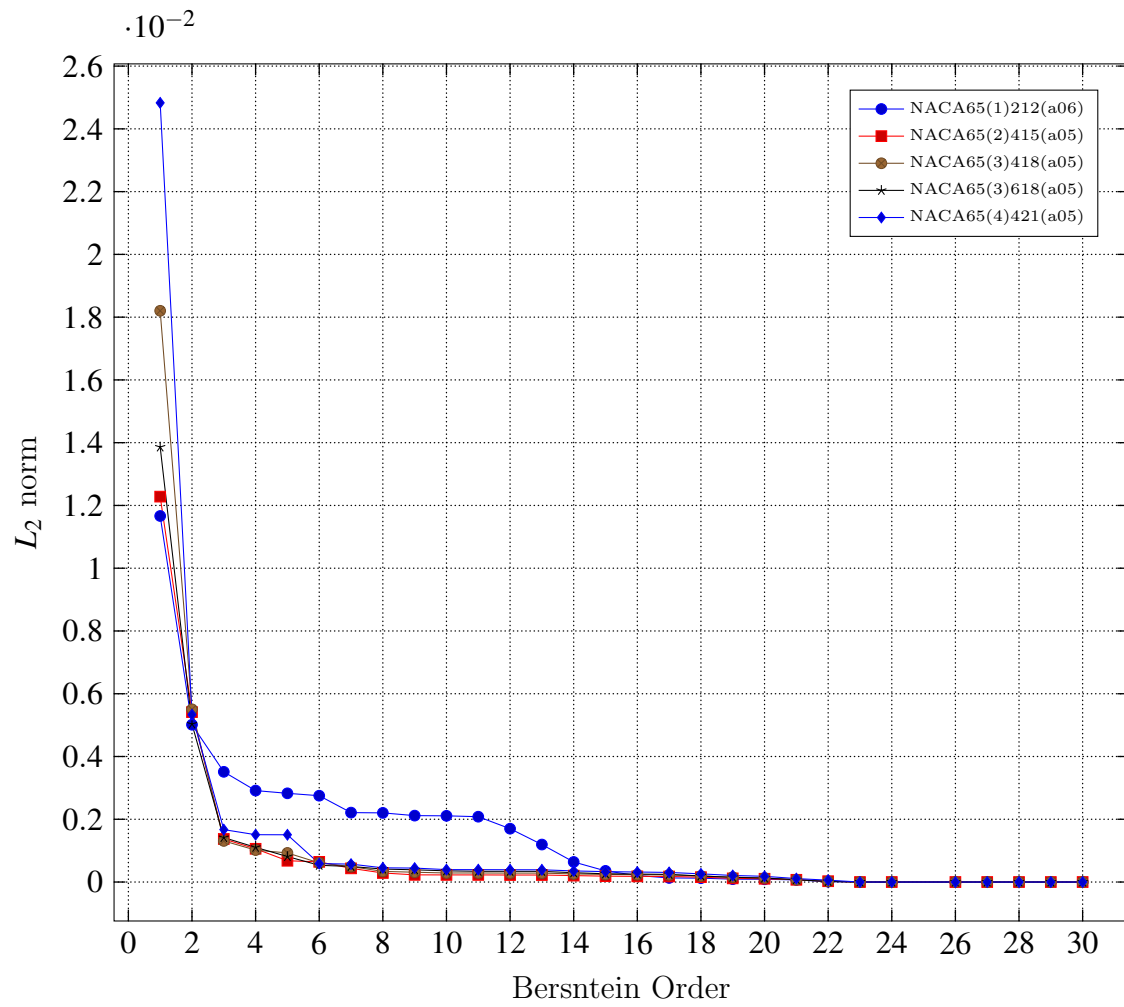


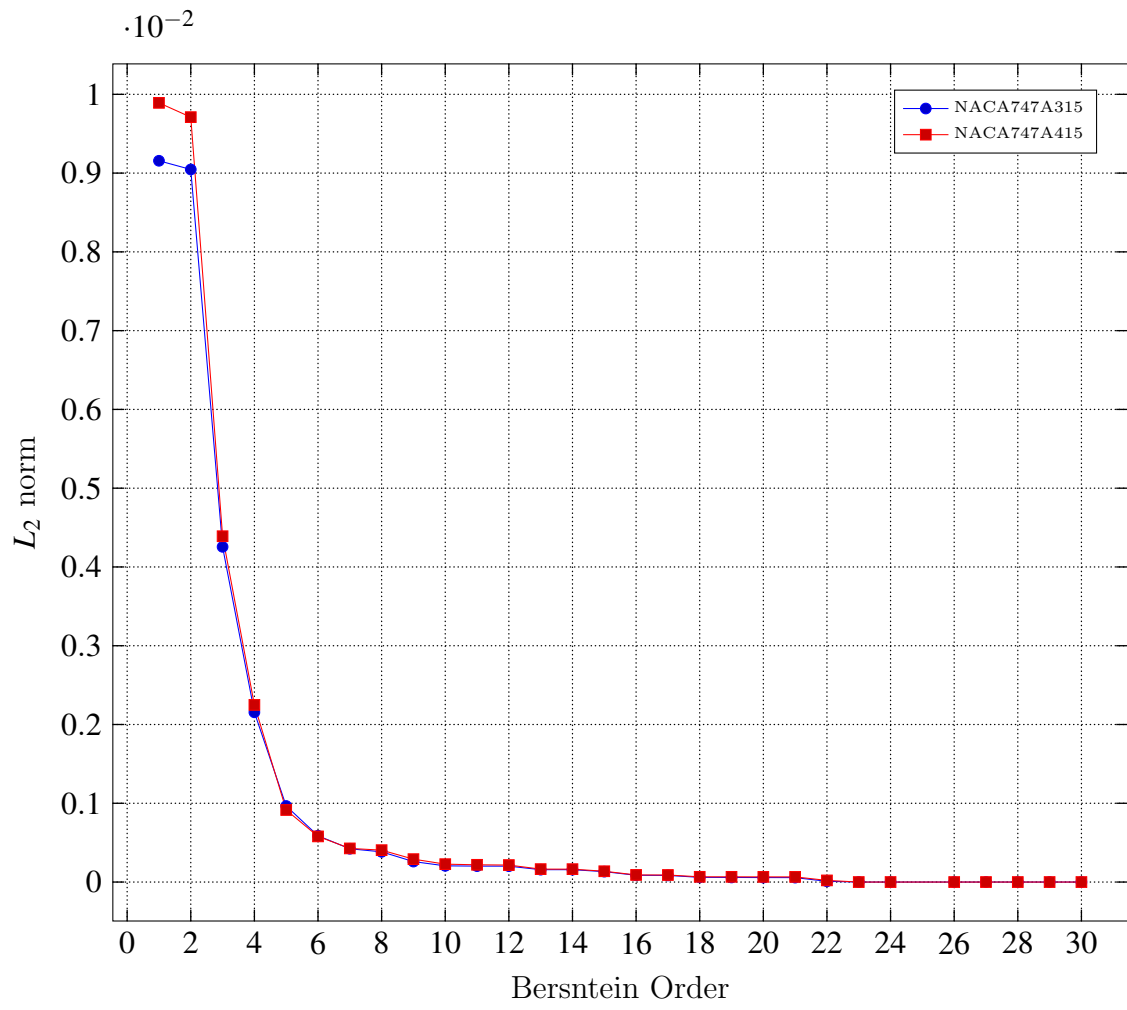
D.2 Lower Surfaces











Appendix E

Data of Airfoils for Surrogate decision

Airfoil	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
	C_l	C_d	C_l	C_d	C_l	C_d	%	%	%	%
NACA 0006	0.30	0.0055	0.332	0.00570	0.332	0.00564	0.1	1.05	10.6	2.5
NACA 0009	0.35	0.0060	0.307	0.00534	0.307	0.00533	0.1	0.19	12.3	11.2
NACA 1408	0.65	0.0075	0.445	0.00536	0.446	0.00535	0.2	0.19	31.3	28.7
NACA 1410	0.45	0.0060	0.423	0.00510	0.422	0.00512	0.1	0.39	6.1	14.7
NACA 1412	0.45	0.0065	0.423	0.00534	0.422	0.00536	0.2	0.37	6.2	17.5
NACA 2412	0.55	0.0065	0.537	0.00509	0.536	0.00514	0.2	0.98	2.5	20.9
NACA 2415	0.50	0.0068	0.526	0.00573	0.524	0.00572	0.4	0.17	4.9	15.3
NACA 2418	0.55	0.0075	0.473	0.00620	0.473	0.00616	0.1	0.65	14.1	17.9
NACA 2421	0.45	0.0080	0.396	0.00689	0.398	0.00688	0.6	0.15	11.5	14.0
NACA 2424	0.45	0.0090	0.320	0.00775	0.329	0.00770	2.7	0.65	26.9	14.4
NACA 4412	0.75	0.0070	0.807	0.00519	0.809	0.00522	0.2	0.58	7.8	25.4
NACA 4415	0.75	0.0075	0.727	0.00587	0.723	0.00582	0.6	0.85	3.7	22.4
NACA 4418	0.70	0.0080	0.648	0.00668	0.648	0.00666	0.1	0.30	7.5	16.8
NACA 4421	0.65	0.0090	0.557	0.00765	0.561	0.00762	0.7	0.39	13.8	15.3
NACA 4424	0.70	0.0100	0.467	0.00868	0.476	0.00859	2.0	1.04	31.9	14.1

Airfoil	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
	C_l	C_d	C_l	C_d	C_l	C_d	%	%	%	%
NACA 23012	0.45	0.0065	0.426	0.00571	0.429	0.00557	0.8	2.5	4.6	14.3
NACA 23015	0.45	0.0075	0.414	0.00631	0.414	0.00620	0.0	1.7	8.0	17.3
NACA 23018	0.45	0.0075	0.389	0.00690	0.383	0.00688	1.5	0.3	14.8	8.3
NACA 23021	0.45	0.0085	0.323	0.00749	0.307	0.00747	4.9	0.3	31.7	12.1
NACA 23024	0.40	0.0095	0.260	0.00834	0.241	0.00838	7.4	0.5	39.8	11.8

Airfoil	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
	C_l	C_d	C_l	C_d	C_l	C_d	%	%	%	%
NACA 63206	0.50	0.0065	0.502	0.00660	0.502	0.00634	0.0	3.9	0.4	2.5
NACA 63209	0.50	0.0068	0.513	0.00673	0.514	0.00627	0.2	6.8	2.8	7.1
NACA 63210	0.50	0.0060	0.516	0.00770	0.523	0.00608	1.3	21.0	4.6	1.3
NACA 64108	0.35	0.0065	0.421	0.00647	0.421	0.00632	0.1	2.3	20.3	2.8
NACA 64110	0.45	0.0070	0.424	0.00673	0.425	0.00637	0.2	5.3	5.5	9.0
NACA 64206	0.45	0.0068	0.500	0.00659	0.500	0.00633	0.1	3.9	11.2	6.2
NACA 64208	0.50	0.0045	0.508	0.00672	0.507	0.00648	0.1	3.6	1.5	44.0
NACA 64209	0.50	0.0068	0.511	0.00692	0.510	0.00650	0.1	6.1	2.1	3.7
NACA 64210	0.50	0.0070	0.512	0.00678	0.512	0.00640	0.1	5.6	2.5	8.6
NACA 65206	0.48	0.0065	0.497	0.00677	0.497	0.00648	0.0	4.3	4.5	0.3
NACA 65209	0.48	0.0075	0.503	0.00695	0.502	0.00658	0.1	5.3	5.7	12.3
NACA 65210	0.50	0.0070	0.503	0.00709	0.504	0.00649	0.2	8.5	0.8	7.3
NACA 65410	0.60	0.0045	0.675	0.00741	0.679	0.00635	0.7	14.3	13.2	41.1
NACA 66206	0.50	0.0070	0.493	0.00650	0.493	0.00668	0.1	2.8	1.4	4.6
NACA 66209	0.45	0.0070	0.490	0.00700	0.494	0.00664	1.0	5.1	9.9	5.1
NACA 66210	0.43	0.0060	0.486	0.00720	0.495	0.00653	1.7	9.3	16.4	8.8

Airfoil	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
	C_l	C_d	C_l	C_d	C_l	C_d	%	%	%	%
NACA 63(1)212	0.55	0.0055	0.530	0.00540	0.528	0.00539	0.4	0.2	4.1	2.0
NACA 63(1)412	0.68	0.0055	0.699	0.00539	0.700	0.00541	0.2	0.4	3.7	1.6
NACA 63(2)015	0.35	0.0055	0.347	0.00524	0.331	0.00529	4.6	1.0	5.5	3.8
NACA 63(2)215	0.50	0.0060	0.540	0.00532	0.536	0.00532	0.6	0.0	7.2	11.3
NACA 63(2)415	0.70	0.0060	0.719	0.00545	0.712	0.00551	1.0	1.1	1.8	8.2
NACA 63(2)615	0.80	0.0063	0.896	0.00565	0.886	0.00574	1.1	1.6	10.8	8.2
NACA 63(3)218	0.55	0.0060	0.523	0.00779	0.541	0.00557	3.4	28.5	1.6	7.2
NACA 63(3)418	0.68	0.0068	0.714	0.00461	0.702	0.00474	1.7	2.8	4.0	29.8
NACA 63(3)618	0.83	0.0068	0.903	0.00600	0.891	0.00598	1.2	0.3	8.0	11.4
NACA 63(4)221	0.53	0.0068	0.552	0.00590	0.547	0.00580	0.9	1.7	4.1	14.1
NACA 63(4)421	0.70	0.0070	0.730	0.00612	0.722	0.00600	1.2	2.0	3.1	14.3
NACA 64(1)012	0.35	0.0070	0.343	0.00628	0.343	0.00608	0.2	3.2	1.9	13.1
NACA 64(1)112	0.45	0.0063	0.432	0.00620	0.432	0.00591	0.0	4.7	4.0	5.4
NACA 64(1)212	0.48	0.0053	0.521	0.00612	0.522	0.00565	0.2	7.7	9.9	7.6
NACA 64(1)412	0.65	0.0055	0.699	0.00539	0.700	0.00541	0.2	0.4	7.7	1.6
NACA 64(2)015	0.35	0.0060	0.357	0.00516	0.356	0.00521	0.4	1.0	1.7	13.2
NACA 64(2)215	0.50	0.0053	0.537	0.00524	0.533	0.00526	0.8	0.4	6.6	0.2
NACA 64(2)415	0.70	0.0058	0.717	0.00538	0.709	0.00539	1.0	0.2	1.3	6.3
NACA 64(3)218	0.50	0.0058	0.544	0.00548	0.539	0.00542	0.9	1.1	7.9	5.7

Airfoil	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
	C_l	C_d	C_l	C_d	C_l	C_d	%	%	%	%
NACA 64(3)418	0.70	0.0065	0.722	0.00566	0.714	0.00562	1.0	0.7	2.1	13.5
NACA 64(3)618	0.78	0.0060	0.896	0.00596	0.885	0.00593	1.3	0.5	14.1	1.2
NACA 64(4)221	0.53	0.0065	0.548	0.00589	0.543	0.00574	0.9	2.5	3.4	11.7
NACA 64(4)421	0.68	0.0068	0.724	0.00610	0.715	0.00596	1.3	2.3	5.9	11.7
NACA 65(1)212	0.48	0.0060	0.508	0.00679	0.513	0.00582	1.0	14.3	7.9	3.0
NACA 65(1)412	0.68	0.0050	0.686	0.00660	0.694	0.00535	1.2	18.9	2.8	7.0
NACA 65(2)215	0.50	0.0050	0.532	0.00509	0.528	0.00501	0.9	1.6	5.5	0.2
NACA 65(2)415	0.65	0.0053	0.713	0.00503	0.704	0.00509	1.3	1.2	8.2	3.0
NACA 65(3)218	0.45	0.0055	0.542	0.00506	0.534	0.00508	1.5	0.4	18.6	7.6
NACA 65(3)418	0.63	0.0055	0.718	0.00525	0.707	0.00524	1.5	0.2	13.2	4.7
NACA 65(3)618	0.85	0.0068	0.892	0.00543	0.874	0.00554	2.0	2.0	2.9	17.9
NACA 65(4)221	0.50	0.0058	0.544	0.00538	0.534	0.00536	1.8	0.4	6.8	6.8
NACA 65(4)421	0.70	0.0065	0.719	0.00553	0.702	0.00559	2.4	1.1	0.3	14.0
NACA 66(1)212	0.45	0.0060	0.477	0.00749	0.498	0.00611	4.4	18.4	10.7	1.8
NACA 66(2)215	0.47	0.0045	0.488	0.00636	0.520	0.00482	6.6	24.2	10.7	7.1
NACA 66(2)415	0.55	0.0045	0.674	0.00583	0.699	0.00473	3.8	18.9	27.2	5.1
NACA 66(3)218	0.50	0.0048	0.538	0.00448	0.530	0.00461	1.5	2.9	5.9	2.9
NACA 66(3)418	0.60	0.0050	0.714	0.00461	0.702	0.00474	1.7	2.8	17.0	5.2
NACA 66(4)021	0.28	0.0053	0.364	0.00472	0.358	0.00469	1.6	0.6	30.3	10.7
NACA 66(4)221	0.48	0.0053	0.539	0.00484	0.529	0.00479	1.9	1.0	11.3	8.8
NACA 67(1)215	0.45	0.0055	0.394	0.00577	0.412	0.00555	4.7	3.8	8.5	0.9

	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
Airfoil	C_l	C_d	C_l	C_d	C_l	C_d	%	%	%	%
NACA 65(1)212(a06)	0.45	0.0045	0.486	0.00587	0.494	0.00476	1.6	18.9	9.8	5.8
NACA 65(2)415(a05)	0.60	0.0055	0.669	0.00456	0.666	0.00467	0.5	2.4	11.0	15.1
NACA 65(3)418(a05)	0.65	0.0055	0.681	0.00506	0.674	0.00511	1.0	1.0	3.7	7.1
NACA 65(3)618(a05)	0.75	0.0060	0.837	0.00520	0.829	0.00533	1.0	2.5	10.5	11.2
NACA 65(4)421(a05)	0.65	0.0060	0.690	0.00549	0.677	0.00556	1.9	1.3	4.2	7.3

	Experimental		Raw Xfoil		CST Xfoil		Dev.Raw-CST		Dev.Exp-CST	
Airfoil	C_l	C_d	C_l	C_d	C_l	C_d	%	%	%	%
NACA 747A315	0.48	0.0053	0.499	0.00478	0.508	0.00474	1.8	0.8	7.0	9.7
NACA 747A415	0.55	0.0055	0.590	0.00492	0.598	0.00488	1.4	0.8	8.7	11.3

Appendix F

RBF Surrogate Fortran Code

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CC Program to build the surrogate model in Naimrod/0
CC 9/7/2015
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

```

PROGRAM RBF_main
INTEGER m,n,mp,np,CFD
DOUBLE PRECISION A(5000,5000),costs(5000,2),new_b(12),
& costs_pred(2), Cl,Cd,OF,Cdo,ClO,w_equality

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCC Read the matrix 'A' from a file
C Matrix A(m,n) contains the control points position (n) of each MACA airfoil(m)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
n=12
mp=5000
np=5000
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
OPEN(UNIT=8,FILE='A.txt')

```

```

DO i=1,1000000
READ(8,*,END=1010) (A(i,j),j=1,12)
m=i
ENDDO

```

```

1010 CLOSE(8)
C To check 'A' is read OK:
C WRITE(*,*) 'number of airfoils=',m
C WRITE(*,*) 'A(13,9)=' ,A(13,9)
C WRITE(*,*) 'A(28,4)=' ,A(28,4)
C WRITE(*,*) 'A(36,12)=' ,A(36,12)
C WRITE(*,*) 'A(4,5)=' ,A(4,5)
C WRITE(*,*) 'A(52,2)=' ,A(52,2)

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCC Read the matrix 'costs' from a file
C Matrix costs(m,2) contains the Cl and Cd coefficients of each MACA airfoil
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

```

OPEN(UNIT=9,FILE='costs.txt')
DO i=1,1000
READ(9,*,END=2121) (costs(i,j),j=1,2)
ENDDO

```

```

2121 CLOSE(9)
C To check 'costs' is read OK:
C WRITE(*,*) 'costs(28,1)=' ,costs(28,1)
C WRITE(*,*) 'costs(42,2)=' ,costs(42,2)

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCC Read value of design variables(position of control points) from the screen

```

```

C READ(*,*) new_b(1),new_b(2),new_b(3),new_b(4),new_b(5),new_b(6),
& new_b(7),new_b(8),new_b(9),new_b(10),new_b(11),new_b(12)
OPEN(UNIT=10,FILE='new_b.txt')
DO i=1,1000
READ(10,*,END=5121) new_b(i)
ENDDO
C 5121 CLOSE(10)

```

```

READ(10,*) new_b(1),new_b(2),new_b(3),new_b(4),new_b(5),new_b(6),
& new_b(7),new_b(8),new_b(9),new_b(10),new_b(11),new_b(12)

```

```

5121 CLOSE(10)
C new_b(1) = 0.101081
C new_b(2) = 0.078757
C new_b(3) = 0.121204
C new_b(4) = 0.057805
C new_b(5) = 0.144904
C new_b(6) = 0.113593
C new_b(7) = 0.165153
C new_b(8) = 0.068141
C new_b(9) = 0.229615
C new_b(10) = 0.184035
C new_b(11) = 0.112204
C new_b(12) = -0.037723

```

```

C print *, new_b(1)
C print *, new_b(12)
C print *, A(5,1)
C print *, A(5,12)
C print *, costs(1,1)
C print *, costs(1,2)
C print *, costs(5,1)
C print *, costs(5,2)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCC Call to the subroutine RBF
CALL RBF(A,costs,new_b,costs_pred,m,n,mp,np)

```

```

OPEN(UNIT=14,FILE='Cl_Cd')
CL=costs_pred(1)
CD=costs_pred(2)
WRITE(14,708) CL,CD

```

```

708 FORMAT(F10.6,1X,F10.6)
CLOSE(14)

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCC Write the objective function(s) in the screen
CCC ClO and Cdo: my reference is the MACA 0010
CCC w_equality: weight for cl the equality constraint

```

```

C CL=costs_pred(1)
C Cd=costs_pred(2)
C Cdo=0.0055
C ClO=0.33
C w_equality=0.4
C OF=(Cd/Cdo)+w_equality*(1-(Cl/ClO))**2
C WRITE(*,*) Cl, ' ',Cd

```

```

STOP
END

```

```

SUBROUTINE RBF(A,costs,new_b,costs_pred,m,n,mp,np)
C SUBROUTINE RBF(A,costs,new_b,m,n,mp,np,CFD)
C integer FUNCTION rb_f_flag(new_b,m,n,mp,np)
C integer FUNCTION rb_f_flag(A,costs,new_b,costs_pred,m,n,mp,np)

```

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

```

SUBROUTINE RBF(A,costs,new_b,costs_pred,m,n,mp,np)
C SUBROUTINE RBF(A,costs,new_b,m,n,mp,np,CFD)
C integer FUNCTION rb_f_flag(new_b,m,n,mp,np)
C integer FUNCTION rb_f_flag(A,costs,new_b,costs_pred,m,n,mp,np)

```



```

37  position=j
    endif
    temp=dist(i)
    dist(i)=dist(position)
    dist(position)=temp
    temp=index(1)
    index(i)=index(position)
    index(position)=temp
38  continue

C The first (e.g., 80) rows of the vector "dist" will now contain the
C distances of the first 80 closest configurations to new_blade.
C Vector "index" will contain the config. # relative to the distance
C of the corresponding row.
C INSONMA: "dist" contiene le distanze di tutte le 1323 configurazioni
C dalla conf. 1324 (new_blade), ordinate per distanza. Ad es, dist(800)
C contiene NON la distanza della conf. 800 da new_blade, ma la distanza
C dell'800-esima configurazione piu' vicina alla new_blade da new_blade.
C Per sapere QUALE e' questa configurazione, devo controllare la matrice
C index: index(800) mi dice infatti la tag number della configurazione
C la cui distanza da new_blade e' dist(800).

C Predict the costs of new blade using the cluster of (e.g., 80) closest
C points. Build 'near', the matrix containing the closest points to new_b:
C cluster=50
C cluster=300
  do 40 i=1,cluster
    do 41 j=1,new_N
      near(i,j) = Aproj(index(i),j)
    continue
  enddo
42  continue

C Calculate the distances of each point from all the others. Note that the
C matrix is symmetric with zero diagonal elements, hence we will only
C calculate the upper triangles. Initialise distance matrix, i.e. matrix FI:
  do 44 i=1,cluster
    do 45 j=i,cluster
      local_dist(i,j)=0.0
    continue
  enddo
46  continue

C Calculate the distance of the i'th point:
  do 47 j=i+1,cluster
    do 48 k=1,new_N
      !sum over the new_N PCs;
      local_dist(i,j) =local_dist(i,j)+(near(i,k)-near(j,k))^2
    continue
    local_dist(i,j) = sqrt(local_dist(i,j))
    local_dist(i,j) = local_dist(i,j) !linear model
  enddo
47  continue
48  continue

C To create the entire local distance matrix, simply copy the upper triangle
C into the lower triangle:
  do 50 i=2,cluster
    do 51 j=1,i-1
      local_dist(i,j) = local_dist(j,i)
    continue
  enddo
52  continue

C Build the matrices for prediction of cost1 and cost2:
  call pinv(local_dist,pinv_local_dist,cluster,cluster,cp,cp)
  do 54 i=1,cluster
    local_cost1(i) = costs(index(i),1)
    local_cost2(i) = costs(index(i),2)
  continue
  call productmatrix(pinv_local_dist,local_cost1,omega1,
  & cluster,cluster,1,cp,cp,1)
  call productmatrix(pinv_local_dist,local_cost2,omega2,
  & cluster,cluster,1,cp,cp,1)
54  continue

C The distances of new_blade from its closest points are obviously the

```

```

C first "cluster" (e.g., 80) values of vector "dist". Using these values,
C calculate the distances (which I call dist_RBF):
  do 60 i=1,cluster
    dist_RBF(i) = dist(i) ! linear model
  enddo
61  continue

C Finally predict the costs of new_blade:
  costs_pred(1)=0
  costs_pred(2)=0
  do 62 i=1,cluster
    costs_pred(1) = costs_pred(1)+omega1(i)*dist_RBF(i)
    costs_pred(2) = costs_pred(2)+omega2(i)*dist_RBF(i)
  continue
  do 63 i=1,cluster
    print*, '1) costs_pred(1) = ', costs_pred(1)
    print*, '2) costs_pred(2) = ', costs_pred(2)
  enddo
  costs_pred(1) = costs_pred(1)/cluster
  costs_pred(2) = costs_pred(2)/cluster
  do 64 i=1,cluster
    if ((costs_pred(1).GT.unfeas1).OR.(costs_pred(1).LT.0).OR.
  & (costs_pred(2).GT.unfeas2).OR.(costs_pred(2).LT.0)) then
      costs_pred(1)=unfeas1
      costs_pred(2)=unfeas2
    endif
  enddo
  do 65 i=1,cluster
    print*, '3) costs_pred(1) = ', costs_pred(1)
    print*, '3) costs_pred(2) = ', costs_pred(2)
  enddo
  costs_pred(1) = costs_pred(1)/cluster
  costs_pred(2) = costs_pred(2)/cluster

C Test for reliability of the prediction. If the first 3 config. are too
C different, e.g. if one is feasible and the others are not, then the model
C cannot predict whether new_blade will be feasible or not. This may be
C because the prediction is very sensitive to the costs of the closest
C points, although we then use 80 to make the actual prediction. I also
C add the test suggested by Iimos: If the predicted cost is very <- from
C those of the 3 closest points, then the prediction is not considered
C reliable and I use CFD. The test has to be done for each of the 2
C objective functions the prediction will be considered unreliable even if
C the test fails for only one of the 2 objective functions. Hence we build
C a vector 'test' which will contain the results of the test for both
C objective functions.
  nearest_points=3
  do 65 tag=1,2
    !tag=1: entropy generation
    !tag=2: blockage.
    min=costs(index(1),tag)
    max=costs(index(1),tag)
    do 66 i=2,nearest_points
      if (costs(index(i),tag).LT.min) min=costs(index(i),tag)
      if (costs(index(i),tag).GT.max) max=costs(index(i),tag)
    continue
    if (costs_pred(tag).LT.min) min=costs_pred(tag)
    if (costs_pred(tag).GT.max) max=costs_pred(tag)
    test(tag) = max-min
  enddo
  do 67 i=1,nearest_points
    if (costs(index(i),tag).GT.min) min=costs(index(i),tag)
    if (costs(index(i),tag).LT.max) max=costs(index(i),tag)
  enddo
  test(1) = test(1)/nearest_points
  test(2) = test(2)/nearest_points
  do 68 i=1,nearest_points
    if (test(i).GT.0.15).OR.(test(i).GT.0.15)) then
      print*, 'test(1) = ', test(1), 'test(2) = ', test(2)
      CFD=1 !i.e., use CFD instead of the RBF model
    endif
  enddo
  do 69 i=1,nearest_points
    print*, 'costs_pred(1) = ', costs_pred(1)
    print*, 'costs_pred(2) = ', costs_pred(2)
  enddo

```

```

c print*, 'rbf arguments = ', m, ' ', n, ' ', mp, ' ', mp, ' ', mp
cc do i=1,n
c print*, 'new_b( ', i, ' ) = ', new_b(i)
cc end do

c do i=1,m
c do j=1,n
c print*, 'A( ', i, ' ) ( ', j, ' ) = ', A(i,j)
c end do
c end do
cc rbf_flag = CFD

c print*, 'rbf_flag_from_model = ', rbf_flag, ' ', ' ', CFD
return
END

SUBROUTINE productmatrix(A,B,C,dim1,dim2,dim3,mp,np,op)
SUBROUTINE productmatrix(B,C,dim1,dim2,dim3,mp,np,op)
Calculates A(dim1xdim2) * B(dim2xdim3) = C(dim1xdim3)
INTEGER dim1,dim2,dim3,mp,np,op
double precision A(mp,np),B(np,op),C(mp,op)
real A(mp,np),B(np,op),C(mp,op)
common /rbf_common/ A, costs, costs_pred
INTEGER i,j,k
do 30 i=1,dim1
do 20 j=1,dim2
C(i,j)=0.d0
do 10 k=1,dim3
C(i,j) = C(i,j) + A(i,k)*B(k,j)
continue
continue
continue
END

10
20
30

Subroutine pinv(A,pinvA,m,n,mp,np)
Subroutine pinv(pinvA,m,n,mp,np)
-----
Inverts a matrix A[m,n] using SVD
-----
double precision A(mp,np),pinvA(np,mp),u(mp,np),v(np,np),w(np)
real A(mp,np),pinvA(np,mp),u(mp,np),v(np,np),w(np)
common /rbf_common/ A, costs, costs_pred
integer mp,np,m,n

C Decompose A with SVD --> A[m,n]=U[m,n].W[n,n].Vtrns[n,n]:
do 8 i=1,m
do 9 j=1,n
u(i,j)=A(i,j) !this is not to lose A
continue
continue
call svdcmp(u,m,n,mp,np,w,v)

C pinvA[n,m]=v[n,n].diag(1/wi)[n,n].u' [n,m]:
wmax=0d0
do 30 j=1,n
if(w(j).GT.wmax) wmax=w(j)
continue
wmi=n*wmax*1.0e-12
do 31 j=1,n
if(w(j).LT.wmin) w(j)=0d0
continue
do 32 j=1,n
do 33 i=1,m
if(w(j).ne.0d0) then
u(i,j)=u(i,j)/w(j)

```

```

else
u(i,j)=0
endif
continue
33 continue
32 do 34 i=1,n
do 35 j=1,m
pinvA(i,j)=0.d0
do 36 k=1,n
pinvA(i,j) = pinvA(i,j) + v(i,k)*u(j,k)
continue
36 continue
35 continue
34 continue

END

SUBROUTINE svdcmp(a,m,n,mp,np,w,v)
SUBROUTINE svdcmp(m,n,mp,np,w,v)
INTEGER m,mp,n,np,NMAX,nothing
double precision a(mp,np),v(np,np),w(np)
real a(mp,np),v(np,np),w(np)
common /rbf_common/ A, costs, costs_pred
PARAMETER (NMAX=1500)
C Given a matrix a[m,n], with physical dimensions [mp,np],
C this routine computes its singular value decomposition,
C A = U * W * V'. The matrix U replaces a on output. The
C diagonal matrix of singular values w is output as a vector
C w[n,1]. The matrix v (NOT the transpose v') is output as v[n,n].
INTEGER i,its,j,k,l,nn
double precision anorm,c,f,g,h,s, scale,x,y,z,rvi(NMAX),pythag
real anorm,c,f,g,h,s, scale,x,y,z,rvi(NMAX),pythag
g=0.0d0
scale=0.0d0
anorm=0.0d0
do 25 i=1,n
l=i+1
rvi(i)=scale*g
g=0.0d0
s=0.0d0
scale=0.0d0
if(i.le.m) then
do 11 k=i,m
scale=scale+abs(a(k,i))
continue
if(scale.ne.0.0d0) then
do 12 k=i,m
a(k,i)=a(k,i)/scale
s=s+a(k,i)*a(k,i)
continue
f=a(i,i)
g=-sign(sqrt(s),f)
h=f*g-s
a(i,i)=f-g
do 15 j=l,n
s=0.0d0
do 13 k=i,m
s=s+a(k,i)*a(k,j)
f=s/h
continue
do 14 k=i,m
a(k,j)=a(k,j)+f*a(k,i)
continue
do 16 k=i,m
a(k,i)=scale*a(k,i)
continue
endif
endif
w(i)=scale *g
g=0.0d0

```

```

s=0.0d0
scale=0.0d0
if(i.le.m).and(.i.ne.n)then
do 17 k=l,n
scale=scale+abs(a(i,k))
continue
if(scale.ne.0.0d0)then
do 18 k=l,n
a(i,k)=a(i,k)/scale
s=s+a(i,k)*a(i,k)
f=a(i,l)
g=-sign(sqrt(s),f)
h=f*g-s
a(i,l)=f-g
do 19 k=l,n
rvl(k)=a(i,k)/h
continue
do 23 j=l,m
s=0.0d0
do 21 k=l,n
s=s+a(j,k)*a(i,k)
continue
do 22 k=l,n
a(j,k)=a(j,k)+s*rvl(k)
continue
do 24 k=l,n
a(i,k)=scale*a(i,k)
continue
endif
endif
anomr=max(anorm,(abs(w(i))+abs(rv1(i))))
do 32 i=n,l,-1
if(i.lt.n)then
if(g.ne.0.0d0)then
do 26 j=l,n
v(j,i)=(a(i,j)/a(i,l))/g
continue
do 29 j=l,n
s=0.0d0
do 27 k=l,n
s=s+a(i,k)*v(k,j)
continue
do 28 k=l,n
v(k,j)=v(k,j)+s*v(k,i)
continue
endif
do 31 j=l,n
v(i,j)=0.0d0
v(j,i)=0.0d0
continue
endif
v(i,i)=1.0d0
g=rv1(i)
l=i+1
do 39 i=min(m,n),l,-1
l=i+1
g=w(i)
do 33 j=l,n
a(i,j)=0.0d0
continue
if(g.ne.0.0d0)then
g=1.0d0/g
do 36 j=l,n
s=0.0d0
do 34 k=l,m
s=s+a(k,i)*a(k,j)
continue

```

```

f=(s/a(i,i))*g
do 35 k=l,m
a(k,j)=a(k,j)+f*a(k,i)
continue
do 37 j=i,m
a(j,i)=a(j,i)*g
continue
else
do 38 j=i,m
a(j,i)=0.0d0
continue
endif
a(i,i)=a(i,i)+1.0d0
do 49 k=n,l,-1
do 48 lts=1,30
do 41 l=k,l,-1
nm=l-1
if((abs(rv1(l))+anomr).eq.anorm) goto 2
if((abs(w(nm))+anomr).eq.anorm) goto 1
c=0.0d0
s=1.0d0
do 43 i=l,k
f=s*rv1(i)
rv1(i)=c*rv1(i)
if((abs(f)+anomr).eq.anorm) goto 2
g=w(i)
h=pythag(f,g)
w(i)=h
h=1.0d0/h
c=(g*h)
s=- (f*h)
do 42 j=l,m
y=a(j,i)
z=a(j,i)
a(j,im)=(y*c)+(z*s)
a(j,i)=-(y*s)+(z*c)
continue
z=w(k)
if(l.eq.k)then
if(z.lt.0.0d0)then
w(k)=-z
do 44 j=l,n
v(j,k)=-v(j,k)
continue
endif
goto 3
endif
if(lts.eq.30) then
WRITE(*,*) 'no convergence in svdcmp'
WRITE(*,*) 'paused, type [enter] to continue.'
READ(*,*)
GOTO 111
endif
x=w(l)
nm=k-1
y=w(nm)
g=rv1(nm)
h=rv1(k)
f=((y-z)*(y+z)+(g-h)*(g+h))/(2.0*h*y)
g=pythag(f,1.0d0)
g=pythag(f,1.0)
f=((x-z)*(x+z)+h*h*((y/(f+sign(g,f)))-h))/x
c=1.0d0
s=1.0d0
do 47 j=l,nm
i=j+1
g=rv1(i)
y=w(i)

```

```

h=s*g
g=c*g
z=pythag(f,h)
rv1(j)=z
c=f/z
s=h/z
f= (x*c)+(g*s)
g=- (x*s)+(g*c)
h=y*s
y=y*c
do 45 jj=1,n
  x=v(jj,j)
  z=v(jj,i)
  v(jj,j)=(x*c)+(z*s)
  v(jj,i)=-(x*s)+(z*c)
continue
z=pythag(f,h)
w(j)=z
if(z.ne.0.0d0)then
  z=1.0d0/z
  c=f*z
  s=h*z
endif
f= (c*g)+(s*y)
x=- (s*g)+(c*y)
do 46 jj=1,m
  y=a(jj,j)
  z=a(jj,i)
  a(jj,j)=(y*c)+(z*s)
  a(jj,i)=-(y*s)+(z*c)
continue
continue
rv1(l)=0.0d0
rv1(k)=f
w(k)=x
48 continue
49 continue
111 nothing=0
return
END
c
c
c
FUNCTION pythag(a,b)
double precision a,b,pythag
real a,b,pythag
double precision absa,absb
real absa,absb
absa=abs(a)
absb=abs(b)
if(absa.gt.absb)then
  pythag=absa*sqrt(1.d0+(absb/absa)**2.d0)
else
  if(absb.eq.0.d0)then
    pythag=0.
  else
    pythag=absb*sqrt(1.d0+(absa/absb)**2.d0)
endif
endif
return
END

```


Appendix G

Kriging Surrogate MATLAB Code

```

A = importdata('costs.txt');
B = importdata('A.txt');
aux = 0;

cl = A(:,1);
cd = A(:,2);

dim = size(B);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

h = linspace(0,15,100);
gamma=0;
cont = 0;

% 1) Empirical Semivariogram
Bproj = B;

gamma_max_cd = 1.2305e-06;
gamma_max_cl = 0.0155;

% 2) Model of Semivariogram calculation
%%% SEMI-VARIOGRAM MODEL
%SPHERICAL

a = 10;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 3) Matrix of Cov calculation
for i=1:dim(1)
    for j=1:dim(1)
        for k=1:dim(2)
            aux = aux + (Bproj(i,k) - Bproj(j,k))^2;
        end
        C(i,j) = sqrt(aux);
        aux = 0;
    end
end

for i=1:dim(1)
    for j=1:dim(1)
        if C(i,j) <= a
            C(i,j) = (3*C(i,j)/(2*a) - C(i,j)^3/(2*a^3)); %spherical
        else
            C(i,j) = 1;
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% PART OF MY POINT
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
new_b = importdata('new_b.txt');
K = zeros(dim(1),1);
for i=1:dim(1)
    for k=1:dim(2)
        aux = aux + ((new_b(k)) - Bproj(i,k))^2;
    end
    K(i) = sqrt(aux);
    aux = 0;
end

for j=1:dim(1)

```

```
        if K(j) <= a
            K(j) = (3*K(j)/(2*a)-K(j)^3/(2*a^3)); %spherical
        else
            K(j) = 1;
        end
    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ORDINARY KRIGING (w2)
    lambda_cl = (ones(1,dim(1))*pinv(gamma_max_cl*C)*gamma_max_cl*K-1)/(ones(1,dim(1))*pinv
(gamma_max_cl*C)*ones(dim(1),1));
    w2_cl = pinv(gamma_max_cl*C)*(gamma_max_cl*K-lambda_cl*ones(dim(1),1));
    lambda_cd = (ones(1,dim(1))*pinv(gamma_max_cd*C)*gamma_max_cd*K-1)/(ones(1,dim(1))*pinv
(gamma_max_cd*C)*ones(dim(1),1));
    w2_cd = pinv(gamma_max_cd*C)*(gamma_max_cd*K-lambda_cd*ones(dim(1),1));

    CL = 0;
    CD = 0;
% SIMPLE KRIGING (w1)
%w1 = pinv(C)*K;

    for i=1:dim(1)
        CL = CL + w2_cl(i)*cl(i);
        CD = CD + w2_cd(i)*cd(i);
    end

data = [-CL CD];

    save('CL_CD','data','-ascii');
```


Appendix H

Nimrodo/O input file

%%%%%%%%% PARAMETER SEARCH SPACE DEFINITION %%%%%%%%%%

```
parameter x1 float range from 0 to 0.5
parameter x2 float range from 0 to 0.5
parameter x3 float range from 0 to 0.5
parameter x4 float range from 0 to 0.5
parameter x5 float range from 0 to 0.5
parameter x6 float range from 0 to 0.5
parameter x7 float range from 0 to 0.5
parameter x8 float range from 0 to 0.5
parameter x9 float range from -0.1 to 0.5
parameter x10 float range from -0.1 to 0.5
parameter x11 float range from -0.1 to 0.5
parameter x12 float range from -0.1 to 0.5
```

%%%%%%%%%

%%%%%%%%% MULTIOPTIMISATION %%%%%%%%%%

results 2

%%%%%%%%% TASKS TO EVALUATE THE OBJECTIVE FUNCTIONS %%%%%%%%%%

```
%%%%%%%%% kriging.m function used, that needs matrices A and costs
%%%%%%%%% new_b is the new design variable that Nimrod/O produces
%%%%%%%%% matlab has to be loaded
```

task main

```
copy kriging.m node:kriging.m
copy A.txt node:A.txt
copy costs.txt node:costs.txt
copy new_b.txt node:new_b.txt
node:execute echo "$x1,$x2,$x3,$x4,$x5,$x6,$x7,$x8,$x9,$x10,$x11,$x12" > new_b.txt
node:execute matlab -nojvm < kriging.m > CL_CD
copy node:CL_CD output.$jobname
endtask
```

%%%%%%%%% OPTIMISATION METHOD %%%%%%%%%%

method mots_ii

```
starts 1
resume optimisation 0
number of regions 4
size of Short Term Memory 20
intensification 15
diversification 25
stepsize reduction 50
initial step size 0.04
stepsize reduction-factor 0.5
size of sample 6
number of evaluations 10000
starting method 0
pattern move mode 0
tolerance 0.000
on error ignore
endstarts
endmethod
```

Appendix I

CST Points FORTRAN function

```

C      function cst_point_writer(B,N1, N2, dzc_upper, dzc_lower, B_orders)
function CST(B,NCP,NEWDPX,NEWDPY,np,flag)
c      PROGRAM CST
c      INTEGER NCP(2), flag
c      real newdp(100,2)
c      double precision newdp(500,2)
c      double precision newdp(100,2)
c      double precision newdpx(500), newdpy(500)
c      integer upper_flag, cont_up, cont_low, B_orders(2)
c      integer num, n_points, double_n_points, aux
c      integer i, j , r , cont, np
c      double precision B(12), Ba_upper(6),B_upper(6,1), B_lower(6,1)
c      double precision Ba_lower(6), xc_lo(76), xc_up(88)
c      double precision M_lower(76,6), M_upper(88,6), points(165,2)
c      double precision zc_lower(76,1), zc_upper(88,1)
c      real dzc_upper, dzc_lower, N1, N2

C      Geometry class
      N1 = 0.5
      N2 = 1
      B_orders(1) = 5
      B_orders(2) = 5
c      Auxiliari variables such as counters
      cont = 0

C A)
CCCCCCCCC Reads the new_b.txt desgin vector from a file CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      OPEN(UNIT=10,FILE='new_b.txt')

      READ(10,*) Ba_upper(1),Ba_lower(1),Ba_upper(2),Ba_lower(2),
      &          Ba_upper(3),Ba_lower(3),Ba_upper(4),Ba_lower(4),
      &          Ba_upper(5),Ba_lower(5),Ba_upper(6),Ba_lower(6)
5121 CLOSE(10)

      do i=1,6
      B_upper(i,1) = Ba_upper(i)
      B_lower(i,1) = Ba_lower(i)
      end do

CCCCCCCCC
C B)
CCCCCCCCC Reads the new_b vector from input argument, B CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C      B_upper(1,1) = B(1);
C      B_upper(2,1) = B(3);
C      B_upper(3,1) = B(5);
C      B_upper(4,1) = B(7);
C      B_upper(5,1) = B(9);
C      B_upper(6,1) = B(11);
C      B_lower(1,1) = B(2);
C      B_lower(2,1) = B(4);
C      B_lower(3,1) = B(6);
C      B_lower(4,1) = B(8);
C      B_lower(5,1) = B(10);
C      B_lower(6,1) = B(12);
CCCCCCCCC

CCCCCCCCC Point distribution desired CCCCCCCCCCCCCC

      n_points = 75

      OPEN(UNIT=11,FILE='xc_up.txt')

      DO i=1,88
      READ(11,*,END=2121) xc_up(i)
      ENDDO

2121 CLOSE(11)

      OPEN(UNIT=12,FILE='xc_lo.txt')
```



```

DO i=1,76
  READ(12,*,END=7121) xc_lo(i)
ENDDO

7121 CLOSE(12)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
  DO i=1,88
    DO j=1,6
C      M_lower(i,j) = 0
      M_upper(i,j) = 0
    END DO
  END DO

  DO i=1,88
    DO j=1,1
C      zc_lower(i,j) = 0
      zc_upper(i,j) = 0
    END DO
  END DO
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
  DO i=1,76
    DO j=1,6
C      M_lower(i,j) = 0
      M_upper(i,j) = 0
    END DO
  END DO

  DO i=1,76
    DO j=1,1
C      zc_lower(i,j) = 0
      zc_upper(i,j) = 0
    END DO
  END DO

CCCCCCC Lower surface point calculation CCCCCCCCCC
  DO i=1,76
    DO r=0,B_orders(2)
      M_lower(i,r+1) = -(xc_lo(i)**(N1))*((1-xc_lo(i))**(N2))
      &*bin(B_orders(2),r)*((1-xc_lo(i))**(B_orders(2)-r))*xc_lo(i)**r
    END DO
  END DO
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

CCCCCCC Upper surface point calculation CCCCCCCCCC
  DO i=1,88
    DO r=0,B_orders(1)
      M_upper(i,r+1) = (xc_up(i)**(N1))*((1-xc_up(i))**(N2))
      &*bin(B_orders(1),r)*((1-xc_up(i))**(B_orders(1)-r))*xc_up(i)**r
    END DO
  END DO
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

  CALL productmat(M_lower,B_lower,zc_lower,76,6,1)
  CALL productmat(M_upper,B_upper,zc_upper,88,6,1)

  DO i=1,165
    points(i,1) = 0
    points(i,2) = 0
  END DO

  aux = n_points
  cont = 0

DO i=1,88

```



```
DO i=1,m1
  BIN=BIN*i
ENDDO
DO i=1,m2
  BIN=BIN/i
ENDDO
DO i=1,(m1-m2)
  BIN=BIN/i
ENDDO

RETURN
END
```


Appendix J

Matrix A

0.0858	0.0858	0.0768	0.0768	0.0809	0.0809	0.0677	0.0677	0.073	0.073	0.0717	0.0717
0.1287	0.1287	0.1152	0.1152	0.1213	0.1213	0.1016	0.1016	0.1096	0.1096	0.1076	0.1076
0.1521	0.1345	0.1534	0.1022	0.1661	0.1038	0.1355	0.0899	0.1569	0.0869	0.1521	0.0871
0.1825	0.1614	0.1761	0.1305	0.1968	0.1274	0.1547	0.1154	0.1833	0.1093	0.1755	0.1115
0.1941	0.1518	0.198	0.1068	0.2325	0.0939	0.1733	0.0947	0.221	0.0731	0.2075	0.0795
0.2441	0.1887	0.225	0.1544	0.2872	0.1231	0.1942	0.1385	0.2658	0.1033	0.2413	0.1169
0.2957	0.2244	0.2496	0.2037	0.3451	0.1503	0.2123	0.1841	0.3123	0.1323	0.2747	0.1545
0.3484	0.2589	0.2727	0.2547	0.4046	0.1757	0.229	0.2312	0.3597	0.1604	0.308	0.1923
0.2179	0.1344	0.2426	0.0579	0.3019	0.0284	0.2126	0.0515	0.2953	0.0019	0.2725	0.0155
0.2757	0.1665	0.2574	0.1126	0.3721	0.0493	0.2194	0.1025	0.3489	0.0275	0.3043	0.054
0.3369	0.1965	0.2664	0.1699	0.4501	0.0677	0.2189	0.1556	0.4072	0.0517	0.3347	0.0926
0.4006	0.2245	0.2714	0.2294	0.5327	0.0839	0.2147	0.2103	0.4677	0.0748	0.3648	0.1312
0.2444	0.1128	0.2041	0.0732	0.1753	0.1899	0.177	0.0544	0.1625	0.1534	0.1673	0.1105
0.3035	0.142	0.2102	0.1383	0.26	0.1938	0.1697	0.1219	0.2235	0.1699	0.1938	0.1541
0.3682	0.1683	0.2037	0.2087	0.3632	0.1909	0.1445	0.1956	0.2956	0.183	0.2159	0.1989
0.0798	0.0595	0.0994	0.0324	0.1128	0.0872	0.1165	0.0132	0.1009	0.0597	0.0896	-0.0611
0.116	0.0919	0.131	0.0706	0.1668	0.1319	0.1432	0.0494	0.1412	0.0933	0.0929	-0.0551
0.1286	0.1037	0.1409	0.0789	0.1838	0.1577	0.1575	0.0465	0.1436	0.114	0.1054	-0.058
0.1195	0.107	0.112	0.0829	0.1894	0.17	0.1272	0.0822	0.1478	0.1223	0.0657	-0.0076
0.0783	0.0583	0.0915	0.024	0.1191	0.0941	0.1165	0.0126	0.1047	0.064	0.0945	-0.0563
0.1141	0.0924	0.1175	0.0519	0.179	0.1518	0.1416	0.0401	0.1468	0.104	0.1008	-0.0495
0.1261	0.101	0.1261	0.0678	0.1995	0.1614	0.1491	0.0588	0.1608	0.1103	0.1025	-0.0445
0.0736	0.0539	0.0908	0.0227	0.1105	0.0863	0.1307	0.0261	0.1141	0.074	0.1026	-0.0486
0.1057	0.083	0.1194	0.0567	0.1599	0.1278	0.171	0.0747	0.1561	0.1098	0.1143	-0.0347
0.1163	0.0923	0.1294	0.0688	0.1759	0.1406	0.185	0.092	0.1691	0.1203	0.1179	-0.0302
0.0702	0.0507	0.0925	0.0241	0.1012	0.0773	0.1247	0.0199	0.1639	0.1239	0.1023	-0.0493
0.1011	0.0788	0.1212	0.0578	0.1449	0.1136	0.1652	0.0681	0.2296	0.184	0.1122	-0.0377
0.152	0.1235	0.1633	0.11	0.2202	0.1762	0.171	0.0859	0.1765	0.1224	0.0968	-0.0487
0.1681	0.1093	0.1862	0.0863	0.2485	0.1457	0.2064	0.057	0.2087	0.082	0.1675	-0.112
0.1666	0.1666	0.1807	0.1807	0.2198	0.2198	0.2041	0.2041	0.1489	0.1489	0.1589	0.1589
0.1874	0.1518	0.1973	0.157	0.2726	0.2096	0.1988	0.1337	0.2091	0.1406	0.1003	-0.0394
0.2074	0.1361	0.2131	0.1327	0.3107	0.1846	0.2246	0.094	0.2479	0.1115	0.1685	-0.1109
0.2289	0.1217	0.2258	0.1066	0.3537	0.1617	0.2454	0.0532	0.2901	0.0825	0.2354	-0.1828
0.2199	0.1786	0.2374	0.2062	0.3385	0.2415	0.1981	0.1824	0.2628	0.1542	0.0917	-0.0294
0.2733	0.143	0.2437	0.1635	0.4298	0.1836	0.249	0.1119	0.3356	0.0884	0.2325	-0.1696
0.2543	0.2026	0.2752	0.2612	0.3652	0.2663	0.266	0.2372	0.2565	0.1618	0.1111	-0.018
0.2844	0.1807	0.2729	0.2458	0.4288	0.2299	0.2654	0.2095	0.3143	0.1234	0.1714	-0.086
0.1351	0.1351	0.1192	0.1192	0.2139	0.2139	0.1269	0.1269	0.1571	0.1571	0.034	0.034
0.1423	0.128	0.1321	0.106	0.226	0.2024	0.1468	0.1063	0.1717	0.1431	0.0699	-0.0021
0.1497	0.1213	0.1447	0.0922	0.2384	0.1918	0.1665	0.085	0.1863	0.1295	0.106	-0.0385
0.1681	0.1093	0.1862	0.0863	0.2485	0.1457	0.2064	0.057	0.2087	0.082	0.1675	-0.112
0.1672	0.1672	0.1522	0.1522	0.2665	0.2665	0.1587	0.1587	0.1887	0.1887	0.041	0.041
0.1852	0.1507	0.1716	0.1299	0.2996	0.2375	0.1893	0.1239	0.2244	0.156	0.1105	-0.0295
0.2045	0.1354	0.1884	0.1054	0.3367	0.2121	0.2159	0.0853	0.2627	0.1257	0.179	-0.1008
0.2198	0.1782	0.1993	0.1698	0.3608	0.2813	0.2109	0.1634	0.2599	0.1786	0.1143	-0.0205
0.2437	0.1602	0.2077	0.1494	0.4103	0.2504	0.2246	0.1302	0.3074	0.1448	0.1791	-0.0911
0.2692	0.1437	0.2125	0.1263	0.4653	0.2231	0.2328	0.0943	0.3585	0.1115	0.2427	-0.1605
0.2554	0.2053	0.2256	0.2108	0.4257	0.3256	0.2279	0.2017	0.2967	0.1999	0.1165	-0.0121
0.2846	0.184	0.2239	0.1951	0.49	0.2881	0.2264	0.1757	0.355	0.1601	0.177	-0.0797
0.1376	0.1108	0.1483	0.0926	0.2099	0.1675	0.2108	0.1254	0.1955	0.1412	0.1242	-0.0219
0.1523	0.0986	0.1739	0.0628	0.2341	0.1492	0.2507	0.0797	0.2244	0.116	0.1969	-0.0954
0.1688	0.1369	0.1787	0.1317	0.2578	0.203	0.2543	0.1819	0.2285	0.1645	0.1353	-0.0072
0.1866	0.1226	0.1987	0.1051	0.2909	0.18	0.2842	0.1412	0.2646	0.1355	0.2052	-0.0794
0.2207	0.1453	0.2259	0.1479	0.3451	0.2127	0.3204	0.2005	0.3006	0.1537	0.2119	-0.0662
0.2444	0.1302	0.2314	0.1222	0.4004	0.1887	0.328	0.1612	0.3522	0.1232	0.2758	-0.1377
0.2295	0.1847	0.2393	0.2133	0.36	0.2745	0.3334	0.2935	0.2919	0.2037	0.1504	0.0169
0.2551	0.1655	0.2453	0.194	0.4133	0.2414	0.3425	0.2638	0.3432	0.1661	0.2145	-0.052
0.1314	0.1053	0.1512	0.0947	0.1858	0.1439	0.211	0.1256	0.2898	0.2355	0.1203	-0.0269
0.1612	0.1304	0.1821	0.1338	0.2244	0.17	0.2605	0.1886	0.3454	0.2811	0.1275	-0.0165
0.1889	0.1526	0.2178	0.1788	0.2537	0.1853	0.3241	0.2676	0.3879	0.3119	0.1353	-0.0049
0.2169	0.1743	0.2517	0.2231	0.2856	0.2017	0.3859	0.3466	0.4321	0.3428	0.1395	0.0037
0.1764	0.1437	0.1466	0.1035	0.2984	0.2358	0.1104	0.0463	0.5293	0.4618	0.1425	-0.0033
0.1417	0.1075	0.1587	0.0805	0.2151	0.1643	0.2295	0.1092	0.213	0.1108	0.0813	0.0447
0.2017	0.1119	0.2121	0.0842	0.324	0.1581	0.3088	0.1041	0.2822	0.1294	0.1152	0.0051
0.2396	0.1331	0.2294	0.1284	0.3931	0.1896	0.3297	0.1644	0.3306	0.1452	0.1161	0.0196
0.2725	0.1136	0.2432	0.0902	0.4613	0.158	0.3524	0.1025	0.3915	0.1149	0.1345	-0.011
0.2766	0.1516	0.2483	0.1758	0.4603	0.2179	0.3525	0.2279	0.3753	0.1555	0.1175	0.0353
0.2178	0.1188	0.182	0.096	0.4043	0.1648	0.156	0.0752	0.2771	0.3085	0.052	0.0116
0.2281	0.1121	0.1889	0.0822	0.4252	0.154	0.1667	0.054	0.2983	0.295	0.0854	-0.0251

Appendix K

Parallel Coordinates

