UNIVERSITY OF SOUTHAMPTON

FACULTY OF ENGINEERING AND THE ENVIRONMENT

Computational Engineering and Design

Multidisciplinary Aircraft Design Optimisation Using an Improved Blackboard Framework

by

Nickolay D. Jelev

Thesis for the degree of Doctor of Philosophy

September 2018

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND THE ENVIRONMENT
Computational Engineering and Design

Doctor of Philosophy

MULTIDISCIPLINARY AIRCRAFT DESIGN OPTIMISATION USING AN IMPROVED BLACKBOARD FRAMEWORK

by Nickolay D. Jelev

The commercial aircraft design process is controlled by chief engineers that meet at regular intervals to make key decisions. This has remained largely unchanged since the early days of aircraft design and has prompted researchers and industry practitioners to explore various communication architectures under the topic of Multidisciplinary Design Optimisation. Although many have been widely studied, they are rarely used in industrial design primarily because they fail to integrate well within the existing organisational structure of aircraft companies.

A legacy blackboard framework for Multidisciplinary Design Optimisation has been the subject of this study. Blackboard frameworks promote concurrent engineering practices using a database, some form of system level controller and a flexible arrangement of the knowledge sources that make up the design process. The specific framework considered in this thesis uses an automatic rule base to iteratively change the bounds on the shared design variables until they converge to what is deemed to be a single optimal design. The thesis covers the development and testing of a novel rule set, which has been given the name "Multidisciplinary Pattern Search" by the author, to reflect that its logic combines ideas from several well established heuristic optimisers.

Two aircraft design test cases demonstrate the merit of the Multidisciplinary Pattern Search, as well as the work done on the database and visualisation modules. The results indicate that the revised blackboard performs better than the distributed Collaborative Optimisation approach, albeit sometimes worse than the monolithic Simultaneous Analysis and Design method that tends to be very organisationally disruptive to implement. An additional 25% reduction in the convergence rate was achieved simply by reusing the available data in the database.

Finally a team based application investigated the ease of use of the revised blackboard method. The feedback highlighted that the process was intuitive and largely easy to use, but further work is needed on a better human process interface.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, Nickolay D. Jelev , declare that the thesis entitled *Multidisciplinary Aircraft Design Optimisation Using an Improved Blackboard Framework* and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as: literature in [63] and [64]

Signed:.....

Date:.....

Acknowledgements

I am extremely fortunate to have been advised by Andy Keane. He has provided immeasurable inspiration and a wealth of advice. Our weekly discussions shaped my attitude to research and the work that follows in this thesis. I am truly grateful for all his efforts and motivational style of supervision. I would like to thank Carren Holden. She has always been passionate about my work and has shared invaluable insight into the industrial design process at Airbus UK. Her comments on the rule base and visualisation modules have been very much appreciated. I am also grateful to András Sóbester for his help and support throughout the entire project.

During these three years, I have had the wonderful pleasure to get to know so many warm and welcoming people at the University of Southampton. I am happy to call all of them my friends as they have made the last three years here an absolute joy.

I would like to express my gratitude to my family. My mom instilled the importance of higher eduction. As an academic herself, she has inspired, advised, helped and financially supported me throughout my academic studies. My dad introduced me to aviation and to this day shares his passion about the subject with me.

Finally, I would like to thank my beautiful and lovely wife Amy. She has been overwhelmingly supportive and I am incredibility lucky to have her as my rock in life. I dedicate this thesis to her and look forward to spending the rest of our lives together.

Nomenclature

Abbreviations

ATC	Analytical Target Cascading
CFD	Computational Fluid Dynamics
CO	Collaborative Optimisation
ECO	Enhanced Collaborative Optimisation
FEM	Finite Element Method
GUI	Graphical User Interface
IRT	Institute of Technological Research
MDO	Multidisciplinary Design Optimisation
MDF	Multidisciplinary Feasible
MDPS	Multidisciplinary Pattern Search
MOCO	Multiobjective Collaborative Optimisation
NSGA-II	Non Dominated Genetic Algorithm
SAND	Simultaneous Analysis and Design
SQP	Sequential Quadratic Programming
UAV	Unmanned Aerial Vehicle

Symbols

a_i	Weighting for local objective functions
b	Semi-span (m)
c	Wing chord (m)
$c_{i_{FAIL}}$	Scaled constraint failure
d	Number of data points
d_{spar}	Spar diameter (m)
e	Oswald's wing efficiency factor (-)
$\mathbf{d}$	Difference between new and current design iterate
f_0	Global objective function
f_i	Objective function of the i^{th} domain
$f(x), f(\mathbf{x})$	Objective function
$f'(x), f'(\mathbf{x})$	First derivative of the objective function

$f''(x), f''(\mathbf{x})$	Second derivative of the objective function
f_{tc}	Form factor coefficient (-)
g	Gravitational acceleration (m/s^2)
$g(x), \mathbf{g}(\mathbf{x})$	Inequality constraint functions
$h(x), \mathbf{h}(\mathbf{x})$	Equality constraint functions
$\tilde{l_U}$	Undercarriage bay length (m)
$lb, \mathbf{lb}$	Variable(s) lower bounds
$lb_s, \mathbf{lb_s}$	Lower bound(s) for shared variable(s)
$\mathbf{lb_{init}}$	Absolute lower bounds limit for shared variables
$\mathbf{lb_l}$	Lower bounds for local variables
$\tilde{m}$	Limit for inequality constraint
m_{aux}	Combined mass of non structural wing elements (kg)
m_{batt}	Mass of foam in the battery (kg)
m_{foam}	Mass of foam in the wing (kg)
m_{fuse}	Total mass exuding the wing (kg)
m_{sprar}	Mass of the main spar (kg)
m_{tot}	Total UAV mass (kg)
m_{wing}	UAV wing mass (kg)
$\tilde{n}$	Limit for equality constraint
n	Load factor (-)
p	Smoothness factor in Kriging kernel
$\tilde{p}$	Pitch-up constraint (-)
p_i	Penalty function
p_{cut}	Estimated cut-out fraction (-)
$\mathbf{r}$	Correlation matrix for query point $\mathbf{x}*$
q	Dynamic pressure (Pa)
$q(y)$	Span-wise loading (N/m)
$\mathbf{r}$	Vector of correlations between the observed data and the new prediction
r_1	Relaxation value for convergence factor
$\mathbf{s}$	Vector of slack functions in ECO
t	Wing thickness (m)
t_{buff}	Buffer gap between spar and skin of the wing (m)
t_{spar}	Spar thickness (m)
$t_{\%}$	Chord-wise location of maximum thickness (-)
$ub, \mathbf{ub}$	Variable(s) upper bounds
$ub_s, \mathbf{ub_s}$	Upper bound(s) for shared variable(s)
$\mathbf{ub_{init}}$	Absolute upper bounds limit for shared variables
$\mathbf{ub_l}$	Upper bounds for local variables
$\mathbf{w}$	Difference between gradients at new and current evaluation point
$\mathbf{x}, x$	Design point
$\mathbf{x_l}$	Local variables only appearing in the i^{th} domain

$\mathbf{x_s}$	Shared variables
$\mathbf{x_t}$	Copies of local, shared and state variables (target variables)
$\mathbf{x_0}$	Optimiser starting point
$\mathbf{x}^*$	Kriging model query point
$\mathbf{y}$	State variables
$\tilde{y}$	Observed values
$\tilde{y*}$, $y(\mathbf{x}*)$	Predicted value from Kriging model
z	Vertical distance from neutral axis for tubular main spar (m)
A_{root}	Estimated wing profile area at the root (m^2)
A_{tip}	Estimated wing profile area at the tip (m^2)
AR	Aspect ratio (-)
$mathbf B$	Correlation function as expressed by Kriging kernel
C	Wing cost (£)
C_{block}	Cost of a black of foam (£)
C_{foam}	Foam material cost for a wing
C_{spar}	Main spar cost for a wing
C_f	Skin friction coefficient (-)
C_{Di}	Induced drag coefficient (-)
C_{Dp}	Profile drag coefficient (-)
C_L	Coefficient of lift (-)
D	Wing drag (N)
D_{fuse}	Total drag exuding the wing (N)
E	Modulus of elasticity (Pa)
E^*	Mass specific energy content of battery (Wh/kg)
FOS	Factor of safety (-)
$\mathbf{H}$	Hessian matrix
$\tilde{\mathbf{H}}$	Approximation of Hessian matrix
I_y	Second moment of area around neutral axis (m^4)
$\mathbf{J}$	Jacobian matrix
J_i	Domain level objective and system level constraint for CO and ATC
L	Wing lifting force (N)
$L(\mathbf{x}, \lambda_\mathbf{g}, \lambda_\mathbf{h})$	Lagrangian function
LF	Likelihood function
M_{root}	Maximum bending moment at root (Nm)
N	Number of domains
No_{wings}	Number of wings that can be manufactured from foam block
No_X	Number of wings that fit in X direction of foam block
No_Y	Number of wings that fit in Y direction of foam block
No_Z	Number of wings that fit in Z direction of foam block
Q_i	Target for local objective function

R	UAV still air range (km)
RR	Recycle region
$R_i, \mathbf{R_i}$	Domain level analysis function(s)
Re	Reynolds number (-)
S	Reference wing area (m^2)
S_{wet}	Wetted wing area (m^2)
V	Cruise velocity (m/s)
$\tilde{V}$	Wing fuel volume (m/s)
W	Total UAV weight (N)
$W_{excl.wing}$	UAV weight excluding the wing (N)
W_{wg}	Transonic aircraft wing weight (N)
$\mathbf{Y}$	Random variable vector
$Y(\mathbf{x})$	Random variable used to represent the output of an analysis
$\alpha_{1,2,...i}$	Objective weightings (-)
δ	Wing tip deflection (m)
$\epsilon_{\mathbf{c}}$	Convergence factor for MDPS
$\epsilon_{\mathbf{c_{init}}}$	Starting convergence factor for MDPS
$\epsilon_{\mathbf{c_{min}}}$	Final convergence factor for MDPS
$\epsilon_{\mathbf{cb}}$	Bound contraction factor for MDPS
$\epsilon_{\mathbf{mb}}$	Bound movement factor for MDPS
$\epsilon_{\mathbf{x}}$	Slack variables minimising shared variables differences in ECO
$\epsilon_{\mathbf{y}}$	Slack variables minimising state variables differences violations in ECO
η_{total}	Total efficiency of the power plant
θ	Width factor in Kriging kernel
λ	Regression constant in Kriging model
$\lambda_{\mathbf{C}}$	Vector of compatibility penalties in ECO
$\lambda_{\mathbf{F}}$	Vector of feasibility penalties in ECO
$\lambda_{\mathbf{g}}$	Lagrange multipliers for inequality constraints
$\lambda_{\mathbf{h}}$	Lagrange multipliers for equality constraints
μ	Mean
π	Circumference-to-diameter ratio (-)
ρ	Air density (kg/m^3)
ρ_{foam}	Foam density (kg/m^3)
ρ_{spar}	Spar material density (kg/m^3)
σ	Material stress (Pa)
σ_{UTS}	Ultimate tensile strength of main spar material (Pa)
σ^2	Variance
Δ	Natural logarithm of likelihood function
Λ	Taper ratio (-)

Subscripts

i	Domain index identifier
j	Current design iterate
$j+1$	Next design iterate

Chapter 1

Introduction

1.1 Background

Commercial aircraft start life as a set of specifications that are put together by the manufacturer with the aid of aircraft operators, as each is anticipating changes in their respective markets. The former sees potential to apply new technologies in the hope to stimulate demand for the new aircraft, while the latter recognises limitations in its existing fleet and aims to increase its competitiveness by modernising it [11]. Whether a manufacturer decides to develop a completely new aircraft, improve an existing design or commit to developing a family of similar aircraft, the process in each case follows a number of predetermined steps.

The first key step defines the product in terms of performance requirements. Since the early 1990s, it has become common practice for aircraft operators to help define some of the requirements in exchange for contractually binding orders [62, 109, 130]. For example range, payload, internal volume, procurement cost, etc., are some of the requirements that can be defined by the airlines. The remaining specifications stem from safety, legislatorial and airworthiness legislation. These often limit the noise, pollution output, evacuation time, etc., of the airframe in the intended countries of operations. Combined these culminate in the specification of a design problem with multiple objectives and numerous constraints in engineering terms. The following stages in the process try to solve this problem using a variety of tools and methods.

The conceptual stage marks the second step in the process. Designers assess several configurations capable of meeting the earlier defined specifications using well calibrated empirical models or low fidelity physics-based analyses. Analysis tools are often grouped together in spreadsheet tools to enable numerous evaluation studies to be performed in a relatively short period of time. A final design emerges after a formal optimisation search or simply by means of manually trading the various design parameters using the available tools [73].

Towards the end of the conceptual stage, the key design variables are fixed and the process begins to change shape. Organisational teams start break up the aircraft into a number of domains, disciplines or sections. Engineers begin to analyse the various aspect of the design separately and sequentially. The aerodynamic shape is generally fixed first and the structural skeleton follows suit, as the whole aircraft moves through the various disciplines in the process. At this stage, any significant changes to the design are largely driven by the need to meet safety and airworthiness requirements [50, 106, 125], rather than the desire to improve performance. Longer design cycles justify the application of higher fidelity analysis tools, which are used to generate more accurate performance estimates and provide guarantees to potential customers [62, 73]. Only when sufficient orders have been received to indicate a return on investment, does the design proceed to the final stage.

The majority of development costs ($10 - $15 billion) [44, 89] are committed by the manufacturer at the start of the detailed design stage. Individual components are conceived and refined asynchronously with the necessary manufacturing processes needed to fabricate the aircraft. This is the most critical point in the development, where most delays and budget cost increases occur [40]. These failures can bankrupt an airframer, which was nearly the case the recent Bombardier C-Series aircraft [118]. While the problems in this stage are not the focus of this thesis, many often originate from the preliminary design stage where the proposed body of research lies. As a result, any improvements in the preliminary design can have a large positive impact on the process. Figure 1.1 represents the main design stages and highlights the scope of the current body of work. Also, the word aircraft is defined to mean fixed wing civil airliners in this thesis, in order to avoid any confusion and ambiguity henceforth.

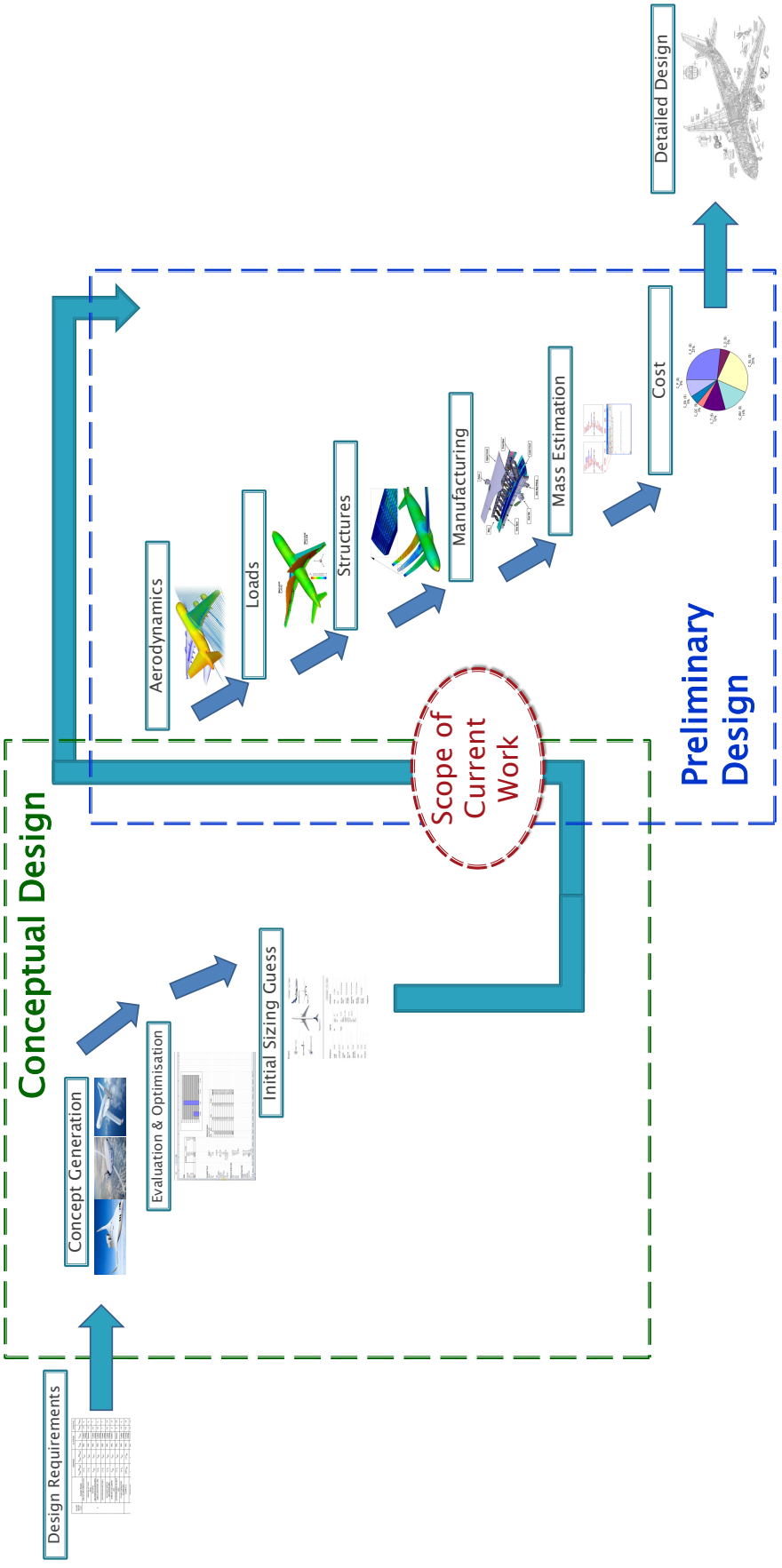

Figure 1.1: Scope of current work within the aircraft design process

1.2 Research Area

The traditional method of designing passenger aircraft sees a single configuration selected fairly early and sequentially changed until it satisfies the multiple performance objectives and regulatory constraints. To understand why this process can be inefficient, consider the simple problem of selecting a single meal option for a large group of people, which can only be reached electronically. The event organiser can propose a meal option as a suggestion to everyone. The first person to respond dislikes that option and between the two of them, they decide on an alternative that they propose to everyone again. As more replies start to come in one by one, this forces the organiser to revise the choice many times in order not to conflict with food allergies and other dietary requests. Trying to reach a consensus in this way can take a long time, particularly when trying to coordinate with a large group of busy individuals.

The traditional method of designing passenger aircraft sees a single configuration selected fairly early and sequentially changed until it satisfies the multiple performance objectives and regulatory constraints. To understand why this process can be inefficient, consider the problem of organising a meeting between a large group of people, which can only be reached electronically. An organiser starts by proposing a date and time to everyone. If the first person to respond is unable to make this option, they will propose and alternative to the organiser and between the two of them will agree on an alternative that they submit to everyone again. As more replies start to come in, this forces the organiser to revise the choice many times in order to fit within schedules of the participants. Trying to reach a consensus in this way can be inefficient, particularly when trying to coordinate with a large group of busy individuals. Using this very metaphor, Ward et al. [156] highlighted these issues in the point-based design process in their paper - *The second Toyota paradox: how delaying decision making can make better cars.* The authors of the paper discussed two strategies to shorten the search in this design model.

The first option forces everyone to meet in person to discuss when to have a meeting. In the scope of the design process, this corresponds to more frequent meetings between engineers from the different organisational departments. In practice this can help reduce the re-work by improving communication, but can also become inefficient because of the need to collocate everyone together, which takes time away from design work.

The second option gives a few important members of the group the power to decide what is best for the collective. This means that everyone has to accept that the final choices may not be satisfactory or optimal for all. In engineering design, this often occurs at key gates where the product is fixed. A small selection of chief engineers would meet and often make decisions on behalf of all disciplines in order to meet set deadlines. This approach is not only vulnerable to human factors in design, but also limits the knowledge used in the decision making process. After all, only a few experienced engineers cannot

represent the collective knowledge of several hundred designers that work behind the scenes [128].

A third option (not discussed by [156]) forces every member to message their preferences to one another. That way, each person is also considering the preferences of others when proposing an alternative to the main meeting slot. A variant of this solution has been used by Boeing and Airbus in the development of the B777 and A380 aircraft. "Design Build Teams" [109] as they would be called in Boeing or "Component Management and Integration Teams in Airbus [40] would combine experts from several organisational domains. This strategy forces teams to work together and consider other's constraints in the face of conflicting aspirations.

While still considering the meeting planning problem, an altogether alternative strategy is to let everybody propose a option and let the organiser find common ground from the preferences. The organiser can then resubmit a smaller list of choices for another round of selection and repeat the process until a single choice is left. This crude metaphor illustrates the basics of set-based concurrent design. Such strategies seek to encourage the exploration of multiple designs concurrently in order to avoid the re-work drawbacks of the point-based design process [156]. The difficulty with this description is that set-based design is an organisational principle. For those that wish to apply it, the generic description does not specify how it would be realised as a computational entity. In other words, given a problem to be solved, the set-based design principle gives a guideline, which is a long way from a working solution.

Organising meetings has been addressed electronically by platforms such as Doodle poll, Google forms and Microsoft calendar. Yet, the industrial design community still lacks dedicated platforms for set-based design. Although there exist numerous mathematical and computational proposals, this thesis focuses on two categories of methods to help advance the research on set-based design. The first uses the blackboard model, which has proven suitable for managing communications across design teams working in concurrent engineering. A database and controller unit manage the information that is output by the various organisational domains and brings them into agreement over successive iterations. Meanwhile, the maturity of optimisation methods and a growing use of computational analysis tools in design, has motivated the development of a range methods in the field of Multidisciplinary Design Optimisation (MDO). These methods combine the analyses tools with formal optimisation search algorithms, to help engineers exploit the tight domain interactions in order to generate better products [98].

The focus of the present research combines aspects from both classical MDO architectures and the blackboard model. The work has been specifically developed for the early preliminary aircraft design process, because it is an area that is yet to benefit greatly from such modern design methodologies. It is however important to stress that the work

has a wider applicability to other MDO problems and could easily be adapted to suite ship, automotive and other forms of multidisciplinary design.

1.3 Hypothesis, Aim and Objectives

It is argued that by combining elements from the blackboard model and some classical MDO architectures, researchers can achieve a framework that is more suitable for industrial design application, rather than the existing selection of distributed MDO architectures. Therefore the project's aim has been to improve and test a legacy blackboard framework to establish its suitability for industrial aircraft design.

This was achieved by addressing the following objectives:

1. Identify the main interaction and conflicts between domains in the current industrial aircraft design process.
2. Develop a new robust system level rule base, a database and graphical user interface for the legacy the blackboard structure.
3. Benchmark the rule base's performance on several test cases and validate its output against competing classical MDO architectures.
4. Develop a data mining module and investigate its ability to speed up the convergence of the new process.
5. Test the intuitiveness and the ease of use the new blackboard system on a team-based design problems.

1.4 Thesis Outline

This thesis is organised into seven chapters. Chapter 2 reviews the literature on the topic. It begins with a survey of some of the most widely used methods in design optimisation and follows with a section on the aircraft preliminary design process. It concludes with a review of the current state of the art in MDO, highlights the opportunities for further research and introduces the contributions in this thesis. Chapter 3 covers the main aspects of the legacy framework and presents the main software architecture developed as part of this work - the Multidisciplinary Pattern Search (MDPS). The following chapter validates the performance of the method using two aircraft design test cases. An Unmanned Aerial Vehicle (UAV) problem is used to calibrate the internal parameters of the Multidisciplinary Pattern Search and a transonic wing design problem validates

the performance of the blackboard against two alternative MDO architectures. Chapter 5 covers the development and testing of a previously unexplored area of blackboard frameworks - the use of data mining module. Chapter 6 covers a team-based application of the new blackboard system and finally, the conclusions and avenues for future research are discussed in the last chapter of the thesis - Chapter 7.

Chapter 2

Literature Review

2.1 Introduction

What follows is a comprehensive review of multidisciplinary aircraft design optimisation. MDO on its own is a large research area spanning multiple topics. Before moving to specific MDO architectures, some of the most widely used numerical optimisation algorithms are explored in Section 2.2. Then, Section 2.3 describes the most important steps in the preliminary aircraft design process to highlight the needs of design engineers. Finally, Sections 2.4, 2.5, 2.6 and 2.7 review the state of the art in MDO and explain the apparent gaps that motivated this research.

2.2 Optimisation Methods

2.2.1 Introduction

In engineering design there are often numerous acceptable solutions that can satisfy a problem. Limited time and resources make it impractical to evaluate every possible solution manually, which means that a design that simply meets the requirements may be implemented regardless of whether it is the *best* [12]. Thanks to recent advances in computing there exist many optimisation methods are able to assist designers in the search for the *best* - or at least an improvement on the current best - design. They are particularly useful on problems with many design parameters or multiple objectives, as they can automatically explore regions that might otherwise be missed using manually driven trade studies [97].

Designers have a broad choice of optimisation algorithms that fall in one of two main classes: gradient-based and gradient-free. Gradient-based methods require the first and sometimes second order gradient of the objective function. This means they are highly

suitable for problems with smooth objective and constraint functions, as they have often been proven to converge significantly faster than gradient-free methods [97]. Gradient-free methods on the other hand, rely solely on previous values of the objective function to generate new trial designs. This means they are generally more suitable to problems where the objective function is noisy or has many discontinuities [60]. Of course there exist numerous exceptions to the contrary outside of these generic descriptions.

There are many different optimisation techniques in the literature. For an exhaustive list, the interest reader is directed to the textbooks by Arora [12], Nocedal and Wright [108], and Rao [124]. The following section reviews a number of methods that are commonly used by design teams in many industries. This makes them relevant to the broader scope of MDO, as well as to the specific research work described here. A significant section is devoted to the Hooke and Jeeves, and Tabu search methods, because much of their logic inspires of the new Multidisciplinary Pattern Search that follows in the next chapter. But Section 2.2.2 covers several gradient-based methods.

2.2.2 Gradient-Based Methods

2.2.2.1 The Newton Method

The Newton method is perhaps the most well-known gradient-based optimisation algorithm. It was originally developed to help solve non-linear equations, but has since has become widely used as an optimisation algorithm [124]. It is highlighted here, to give context to the more complex Sequential Quadratic Programming optimiser that follows suit. Newton's method assumes that the objective function can be approximated as a quadratic in the region around the optimum and uses not only the first, but also second derivative information to find a point where the gradient of the objective is zero. Equation 2.1 generates new candidate improvement points that minimise the single variable function $f(x)$. Here x_{j+1} is a new design point, x_j is the current design point, $f'(x_j)$ and $f''(x_j)$ are the values that the first and second derivatives.

$$x_{j+1} = x_j - \frac{f'(x_j)}{f''(x_j)} \tag{2.1}$$

For the multivariate case, the method generalises to:

$$\mathbf{x_{j+1}} = \mathbf{x_j} - \mathbf{H}^{-1}\mathbf{J}, \tag{2.2}$$

where $\mathbf{H}$ and $\mathbf{J}$ are the Hessian and Jacobian matrices respectively. Standalone this method has the fastest convergence property (known as *quadratic convergence*), because it uses both the first and second order gradient information of the objective function

[124]. Despite this benefit, it is not guaranteed to converge if the objective is non-smooth or if a poor starting position is selected. In addition, it is unsuitable for many practical engineering applications where the analytical gradients of the objective function are unavailable. This has led to the development of quasi-Newton methods, which use mathematical approximation techniques to calculate the gradients.

2.2.2.2 The Sequential Quadratic Programming Method

Arguably the most popular quasi-Newton method is Sequential Quadratic Programming (SQP) [19]. It has seen widespread use in design optimisation [6, 7, 59, 121, 155] primarily because it is efficient and robust even on certain non-smooth problems [90]. As per most recent developments in optimisation, it is not one single algorithm, but rather a combination of several different concepts. It combines Newton's update formula with the method of augmented Lagrange multipliers to handle the constraints [19]. Where the Newton method requires the analytical gradients, SQP iteratively updates a model of the inverse Hessian, which is particularly advantageous in engineering optimisation where the precise Hessian is often unavailable. A description of the SQP method that uses the Broyden Fletcher Goldfarb Shanno formula follows here.

First, the method approximates the objective function using a Taylor series quadratic model, which can be expressed as:

$$f(\mathbf{x_{j+1}}) = f(\mathbf{x_j}) + \Sigma f'(\mathbf{x_{j+1}} - \mathbf{x_j}) + \frac{1}{2}\Sigma f''(\mathbf{x_{j+1}} - \mathbf{x_j})^2. \tag{2.3}$$

In keeping with the previous notation and differentiating Equation 2.3, the gradient of the quadratically approximated objective function at the point $\mathbf{x_j}$ can be obtained by:

$$f'(\mathbf{x_{j+1}}) = \mathbf{J_j} + \mathbf{H_j}(\mathbf{x_{j+1}} - \mathbf{x_j}). \tag{2.4}$$

Newton's method sets gradient $f'(\mathbf{x}) = 0$ to determine the next iteration point. By rearranging Equation 2.4 with the left side set to 0, the optimum step can be found to be:

$$\mathbf{x_{j+1}} - \mathbf{x_j} = -\mathbf{H_j}^{-1}\mathbf{J_j}. \tag{2.5}$$

Equation 2.5 is mathematically equivalent to the Newton's formula given by Equation 2.2. The left side of equation represents a step in the direction of the minimum. The right side of the equation contains the Jacobian and Hessian matrices, for which accurate approximations are needed. The Jacobian can be easily approximated by means of numerical differentiation, such as finite differencing for example. There are also other

differentiation methods such as complex step and adjoint method for this purpose, but these are not considered any further.

As mentioned previously, the inverse of Hessian can be iteratively approximated. SQP begins with an initial guess of the Hessian (usually the identity matrix [108]) and updates with every new point. Although there are several ways to approximate the Hessian [119], the method by Broyden Fletcher Goldfarb Shanno is presented here as it seems to be most commonly used. An approximation of the inverse Hessian can be shown to equal

$$\mathbf{\tilde{H}_{j+1}^{-1}} = \mathbf{\tilde{H}_{j}^{-1}} + \frac{\mathbf{w}\mathbf{w^T}}{\mathbf{w^T s}} - \frac{\mathbf{\tilde{H}_{j}^{-1}} \mathbf{s} \mathbf{w^T} \mathbf{\tilde{H}_{j}^{-1}}}{\mathbf{w^T} \mathbf{\tilde{H}_{j}^{-1}} \mathbf{s}}, \tag{2.6}$$

where

$$\mathbf{d} = \mathbf{x_{j+1}} - \mathbf{x_j} \tag{2.7}$$

and

$$\mathbf{w} = f'(\mathbf{x_{j+1}}) - f'(\mathbf{x_j}). \tag{2.8}$$

Without a method of handling constraints, SQP is not very useful for many engineering applications. SQP uses the method of Lagrange multipliers to satisfy constraints. The Karush Kuhn Tucker conditions state that the gradient of the Lagrangian function should be equal to zero. The Lagrangian can be represented by:

$$L(\mathbf{x}, \lambda_\mathbf{g}, \lambda_\mathbf{g}) = f(\mathbf{x}) - \lambda_\mathbf{g}^\mathbf{T} \mathbf{g}(\mathbf{x}) - \lambda_\mathbf{h}^\mathbf{T} \mathbf{h}(\mathbf{x}), \tag{2.9}$$

where $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ are the inequality and equality constraints and $\lambda_\mathbf{g}$ and $\lambda_\mathbf{h}^\mathbf{T}$ represent the Lagrange multipliers.

$$\begin{aligned} \mathbf{g}(\mathbf{x}) &\geq 0 \\ \mathbf{h}(\mathbf{x}) &= 0 \end{aligned} \tag{2.10}$$

So to find the point $\nabla L(\mathbf{x}, \lambda_\mathbf{g}, \lambda_\mathbf{h}) = 0$, the Lagrange multipliers need to be solved separately using the method of multipliers in an iterative scheme.

2.2.3 Gradient-Free Methods

2.2.3.1 Genetic Algorithm

Genetic algorithms are one of the most widely used gradient-free optimisation methods. Arguably the most popular multiobjective optimiser - Non Dominated Genetic Algorithm II (NSGA-II) [34] - has seen applications in morphing wing design [16], wing control surfaces [84], families of aircraft [114], orbital re-entry vehicles [94] and hypersonic nose cones [35]. Unlike the previously discussed gradient-based methods, this gradient-free method generates new query points using only the output of the objective and constraint functions - or fitness functions as normally referred in the optimisation community.

Genetic algorithms mimic the biological natural selection process to generate new trial solutions. The optimisation procedure starts with an initial pool of trial solutions, which makes up the first generation of designs. Each design is then ranked and assigned a probability that reflects their objective (or fitness) function value. For the multi-objective case NSGA-II however, the Pareto rank is used to reflect this fitness. Designs that improve the objective function assume a higher probability of selection, meaning they are more likely to be chosen for the next step in the process. The second generation of designs is created using pairs of designs from the previous generation. A crossover method mixes design attributes in the pairs to produce new designs as offspring. These make up the second generation of trial solutions. The process is repeated until either the maximum allowable number of iterations/generations has been exhausted or there is little or no change in objective function over successive generations. Mutation and elitism are two phenomena that also play a role in the algorithm. Elitism guarantees that the best designs are kept so over successive generations, while random mutations in the attributes of offspring ensure that (at least in theory) all regions in the design space are searched [24].

Although these methods are commonly criticised for being slow to converge, they possess a number of appealing features. They are particularly good at avoiding local minima and can perform well on problems that are both discrete or noisy [24]. Some are also able to generate an entire Pareto Frontiers in a single optimisation run [34], making them well suited for multiobjective problems. These are some of the reasons why they remain a popular choice in academia and industry.

2.2.3.2 The Hooke and Jeeves Pattern Search

The Hooke and Jeeves method pre-dates genetic algorithms and is one of many pattern search methods that exist in the literature [57]. Early versions (also known as systematic or neighbourhood searches) usually divide the available design spaces into sectors, which

are systematically explored until a minimum was found. The region that contains the lowest observed value was magnified and new points with smaller separation generated in the vicinity of the optimum. Although very simple and robust, these methods are slow to converge because they ignore prior historical information when generating new designs. Newer pattern search methods put emphasis on the use of previously generated solutions to infer or disregard possible successive designs. The Hooke and Jeeves search [57] is one of the most popular algorithms and was described by its authors as:

> "the sequential examination of trial solutions involving comparison of each trial solution with the *best* obtained up to that time together with a strategy for determining (as a function of earlier results) what the next trial solution should be."

The method begins with an initial evaluation of the objective function at the starting *base point*. A new trial solution is selected by perturbing one of the variables by an initial step in an arbitrary direction. The trial solution is kept if it improves the objective function, or otherwise a move in the opposite direction is explored. If neither move improves the objective, the step is rejected and the procedure is continued to the next variable. If however a move was successful, the design is set as the new point. An iteration of the search concludes when the list of design variables has been exhauste, which then establishes the new base pointd. Subsequent trials of this logic showed that the movement direction is usually conserved over consecutive searches. Bell and Pike [15] proposed that the successful directions of the moves should be used as the primary direction in the next iteration of the search in order to reduce the number of function evaluations. Similar use of memory was exploited in the original formulation in the form of a pattern move [57]. The algorithm stores the magnitude and direction of successful moves and applies a single move across all those variables that resulted in an improvement at the end of a neighbourhood search. The move accumulates and grows over successive searches when its outcome is a success. Overall it acts as a short-cut bypassing what would otherwise be several neighbourhood searches in the same direction. If the pattern move is rejected however, the legacy information stored within it is no longer useful and it is therefore re-set.

The final part of the method sees the step size reduce if both the search and pattern move fail to produce improvements. This ensures that the precision of the search increases when no further progress can be made during the searching stage. The process is deemed converged when the algorithm reaches a predetermined step size limit.

2.2.3.3 The Tabu Search

The Hooke and Jeeves search makes only limited use of memory to speed up the process. This is perhaps because it was conceived during an era when computers had limited

memory. As computational performance started to increase, an improvement on the original was developed in the form of a Tabu search [46]. It uses several sophisticated memory cycles in order to avoid the re-evaluation of designs and help avoid local minima.

The author of the Tabu search [46], and others after [154], recognised that the Hooke and Jeeves method can sometimes re-evaluate designs that have been queried before. This could be easily avoided by storing all moves in a database, which could be checked before each new evaluation for duplicates. It is worth noting that in the description of the Multidisciplinary Pattern Search that follows in Section 3.2, the this is rule is referred as "duplicate check", even though it is called a "Tabu move" in the scheme of the Tabu search.

The Tabu search makes more sophisticated use of memory rather than simply checking for duplicate designs. While a full description can be found in a number of references [30, 46, 47], the description that follows uses the version presented in the publication by Connor and Tillery [30]. Using the database, the algorithm would sometimes accept results that degrade the objective function in order to explore wider areas in the design space. To prevent re-evaluation and loss of optimal design, several memory cycles are used to update the lists of forbidden or otherwise known as "Tabu" moves. Short term Tabu moves are continuously updated in the database on a first in and first out basis, giving the algorithm time to settle in a new area before removing them from the "Tabu" list. Long term Tabu moves are kept to prevent the re-evaluation of the same designs. These cycles were controlled using a set of rules supplementing the Hooke and Jeeves search. The original method was developed for combinatorial problems, but has been shown to work on continuous problems as well [61].

Although there exist a myriad of optimisation toolboxes containing sophisticated gradient-based and other gradient-free methods, these relatively simple heuristic optimisers still remain in use to date [92]. Their relative simplicity and robustness makes them particularly attractive for rapid in-house adaptation, which is why their logic underpins the Multidisciplinary Pattern Search. These descriptions will prove useful to the reader as Section 3.2 makes a comparison between the newly developed Multidisciplinary Pattern Search and these two optimisers.

2.2.3.4 Constraint Handling Techniques

Both gradient-free and gradient-based optimiser augment the objective function in some way to deal with constraints. The most popular form of constraint handling is the penalty function. They work by augmenting the objective function with a value proportional to the constraint violation. In practice, there is a trade-off in selecting the severity of the applied penalty. If the optimum lies on a constraint boundary and the penalty function is too high, the algorithm will likely be pushed inside the feasible region and

stall to a sub optimal design [138]. Alternatively if the penalty is too low, the search time may spend a lot of time exploring the infeasible design space and risks prematurely converging to an undesirable design. Finding the optimal balance allows the penalty function to score infeasible designs worse than feasible ones without adversely affecting the convergence [29]. This can often be achieved using adaptive penalty functions that vary depending on the constraint failure.

Separation approaches are also commonly used as constraint handling [117]. The underlying idea is that feasible solution is better than an infeasible one. In other words, the algorithm should first locate the feasible region, before searching for the optimal solution inside it. As such, the method switches between minimising the objective or minimising constraint failure throughout the search. Three rules [33] manipulate the search algorithm to focus on improving feasibility or the objective. These are:

- When comparing two feasible solutions, the one with the best objective function is chosen.
- When comparing a feasible and an infeasible solution, the feasible one is chosen.
- When comparing two infeasible solutions, the one with the lowest sum of constraint violation is chosen.

An obvious disadvantage to these is they tend to be less effective in heavily constrained problems [95].

Early surveys that reviewed penalty methods [29, 138] come to very similar conclusions. Often individual penalty functions are given with elaborate mathematical justifications, but most are derived from intuitive strategies much like the heuristics optimisers that they are used with [138]. In reality they may or may not work on problems outside those tested in the publications. It is up to the user to select and fine tune a strategy according to the problem at hand [29]. A much more recent survey encompassing a wider array of constraint handling techniques [102] highlights that the three rules are the most widely used method in nature inspired constrained optimisation, while custom penalty fictions still remained popular

2.3 Preliminary Wing Design Process

This section moves away from optimisation methods and instead focuses on the preliminary wing design process used at Airbus UK. The aerodynamics team is the first

discipline in a sequence. In theory, being first should allow them to dictate the external shape of the wing, but experience from preceding designs heavily limits the available design space. For example a maximum span limit is usually introduced to enable to aircraft to fit inside set airport gates or a maximum wing root thickness to allow sufficient spaces for the landing gear and fuel. The aim of this team is to define the external wing shape such that it satisfies the desired stall and flight handling characteristics, without degrading the aerodynamic efficiency.

Once the external shape is more or less fixed, the design moves to the loads department, where engineers model the loads that the aircraft will experience during its use. Experienced engineers then manually select several critical cases [105], which are next passed to the structures team for analysis. It is noteworthy that the loads department have their own analysis tools and do not explicitly rely on the Computational Fluid Dynamics (CFD) tools from the aerodynamics domain, which allows them to work fairly independently from the aerodynamics and structures domains. Of course they have very little freedom to change the design, as their main purpose is to generate the necessary datasets for the structures domain.

From then on, there is very little opportunity to make major changes to the core external shape without expensive and time consuming re-works. Instead, structural engineers have significant freedom to design the internal wing structure that is not only able to withstand the critical aerodynamic loads, but is also lightweight, cheap and easy to manufacture, maintain and inspect. Separate teams of engineers look after the mass, costing and manufacturing objectives, but they generally work closely together with the structures domain. The final wing structure is usually decided by manually trading the available design variables using Finite Element Modelling (FEM), however there is some use of formal optimisation methods at this stage as well [49, 85].

When both the internal and external wing geometries have been fixed, the design moves to the manufacturing domain. Using designated analysis tools, engineers modify the wing structure to meet manufacturing constraints. In some cases the geometry is changed from where the fabrication process is unable to the desired tolerances of the optimised wing. Finally, the wing design moves to the costing and performance domains where the contractually binding attributes of the design can be estimated in more detail.

Figure 2.1 illustrates the current, typical arrangement of the organisational domains in preliminary wing design process at Airbus UK. While there are numerous other domains including systems, controls, propulsion integration, etc., they tend to emerge later on in the preliminary design process. Here they have been deliberately excluded to keep the description of the process simple and to illustrate the desired application of the research described in this thesis.

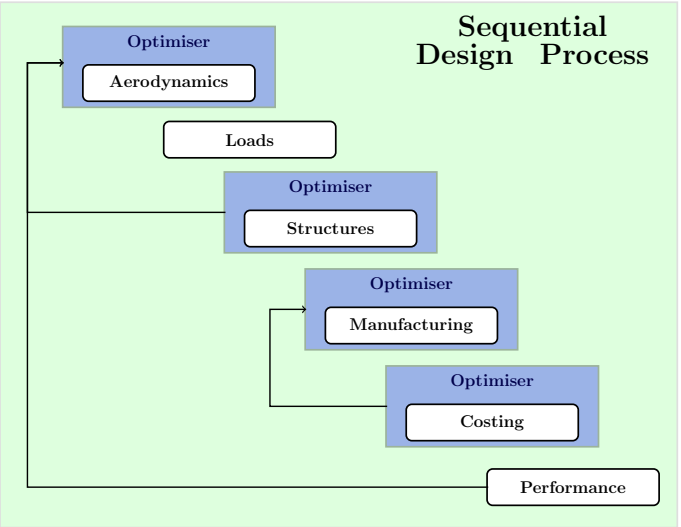

Figure 2.1: Overview of the sequential design process followed in industry

These independent optimisations in series draw criticism for a number of reasons. The most important design decisions are made at key gates, where designers from all of domains meet to discuss different design options. Particularly in the early stages where analysis data is scarce, the process often relies on experienced engineers to pick between different design solutions. This means the process becomes heavily reliant on a few experienced individuals, which may be vulnerable to unintentional biasing from the human aspects of design [73]. Secondly, the process is notorious for developing bottlenecks, which occur because the domain analyses run at different speeds. The loads department is a prime example of this, as engineers working in this domain require a considerable time to evaluate and decide which critical load cases to pass down to the structures domain. And finally, there is the danger that engineers from the early domains may make poor decisions because they lack information on how these will affect other domains down the line. This can result in expensive and time consuming re-design, particularly because the process has to satisfy the numerous internal communication loops, some of which are illustrated in Figure 2.1.

The aircraft manufacturers have previously attempted to address these issues using managerial changes. During the design of 777 (1988-1995), Boeing recognised that past aircraft projects were overrun by communication problems between the many different divisions in the organisation [130]. One somewhat successful solution has been the introduction of "Design Build Teams", which included representatives from different divisions with the aim of addressing design problems through inter-disciplinary communication. Although was applied at the detailed design stage, it was also implemented in such a way as to promote group problem solving in the preliminary stage [109]. Airbus applied a similar approach in the design of the A380 (1994-2005). Aircraft Component Management teams were established to oversee the design of major parts of the system (e.g. wings, fuselage, empennage). Component Management and Integration Teams

coordinated with Component Design and Build teams, each consisting of 40-50 people [40].

Although the design process of the Boeing 777 is considered a success [130], the design of A380 was seen as failure in terms of adhering to budgetary and time constraints [40]. This highlights that there continues to exist problems with the sequential design process, particularly when applied to clean sheet designs. This is why many academics continue to advocate the use of more elaborate concurrent design and optimisation methods [86, 98].

2.4 Classical Multidisciplinary Design Optimisation (MDO) Architectures

2.4.1 Introduction

MDO is a collection of methods and architectures which are able to solve complex problems partitioned into a number of disciplines, subsystems or domains. In the early days, academics tackled MDO problems using a decomposition strategy specifically developed for the problem at hand [38, 69, 101]. Successes in aero-structural optimisation however, championed the development of more transferable methods that could be used across multiple problems. These have now become a standalone research field, where the focus is on the process, rather than the outcome of the design optimisation.

The works by Haftka [51] and Sobieszczanski-Sobieski [141] can be considered as the beginning of MDO. Since then, numerous other architectures have been developed from two contrasting trains of thought: monolithic or distributed. Figures 2.2 and 2.3 illustrate differences between the two classes of methods using the disciplines from the aircraft design process.

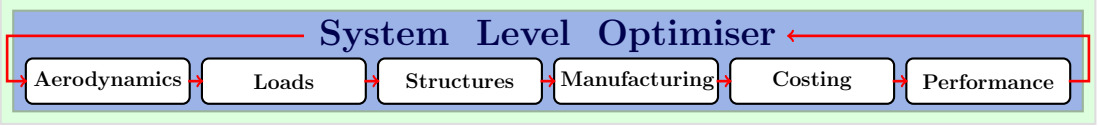

Figure 2.2: Monolithic MDO design model

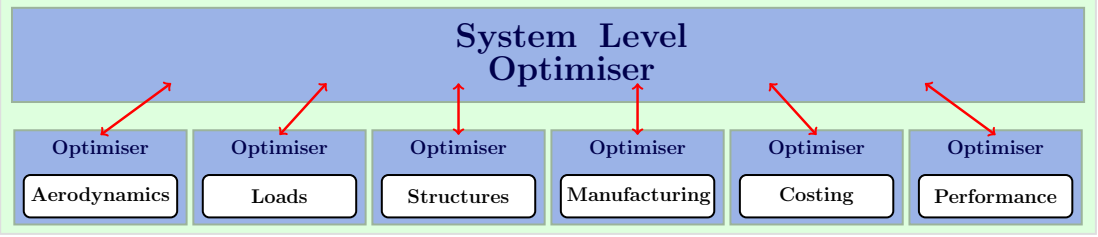

Figure 2.3: Distributed MDO design model

Monolithic architectures combine all discipline analyses under a single optimiser, which controls the entire design process. One can simply imagine the monolithic approach as a way of converting the multidisciplinary optimisation problem into a single disciplinary one. These strategies can be quick and robust to converge when used with gradient-based or surrogate-assisted optimisers. Because they are generally able to solve complex problems in fewer analysis evaluations than distributed approaches [120, 146], their application in certain aspects of industrial design engineering has seen a significant increase [2].

There are three accepted ways to combine the multiple disciplines under a single optimiser. These were given the names Simultaneous Analysis and Design (SAND) [51], Individual Discipline Feasible [32] and Multidisciplinary Feasible [32] by their authors. Simultaneous Analysis and Design is covered in more detail later in Section 2.4.3 because it is used for comparison in the later sections of this work, while the others are not discussed further here.

There remain a plethora of problems that hinder the merger of all analysis tools under a single optimiser. In the preliminary aircraft design stage for example, proposed designs need to be analysed by various black box programs. These are often spread across multiple divisions in the organisation and require regular input from skilled design engineers. In these cases, monolithic architectures would be unsuitable because they can become notoriously difficult to operate and maintain. This is especially the case when the analysis programs undergo regular updates [21] or necessitate different types of optimisers. Furthermore to benefit from the higher rate of convergence, these architectures often require gradient or surrogate-based optimisers, which have separate obstacles to implementation when gradients over the entire process are needed.

To make MDO applicable to organisationally dispersed problems, several multi-level (or distributed) MDO architectures have emerged in recent years. Simply put, these permit low level domains to control and optimise an aspect of the design using their preferred methods, while a system level optimiser coordinates their interactions. This means that organisational domains are able to conduct their individual analyses concurrently and in isolation from others, all while utilising low cost distributed computational resources.

The main challenge in distributed optimisation is how to coordinate the multiple domains into a single cohesive design. From a mathematical point of view, what links the multiple organisational domains is the shared and state variables. Shared design variables are those that are common across multiple domains, while state variables are output from the analysis in one domain and are the inputs to another domain. One attractive method to ensure convergence on the shared and state variables is to introduce a common vector of *targets* that all domains can work towards. In other words, one can envision the chief engineering team setting a set of design and state variables targets that each domain tries to achieve. If these targets are unattainable, they are revised and new ones are

generated based on the feedback from the individual design teams. Hence all domains work towards a common vector of targets, thus ensuring system feasibility at the end of the process. This model offer a strategy that is in theory well suited to industry,a As aircraft design is often driven by contractual requirements rather than optimum performance. This hypothetical model of the design process can be mathematically simulated using several stages of optimisers. Sections 2.4.4, 2.4.5 and 2.4.6 review a family of target-based architectures. These architectures are among the most widely studied in the MDO field (Figure 2.4) suggesting greater simplicity and fewer obstacles to implementation, which is why they have been reviewed next.

Although this thesis predominantly focuses on MDO application in the *aircraft design* process, it must be stressed that MDO architectures have been applied to a wide variety of problems outside the aerospace sectors including, but not limited to, water pumps [8], supply chains [123], internal combustion engine [93], ship design [53, 163], stiffened panels [25] and automotive suspension [71, 72, 79], chassis [96] and layout [80, 100]. A cursory examination of the literature on some of the most widely known methods shows a growing interest in both academia and industry. What follows next is a review of some of the most popular MDO architectures found in academia. Some of these are illustrated in Figure 2.4, which shows the citation count of each architecture and highlights the continuing growth of this academic topic.

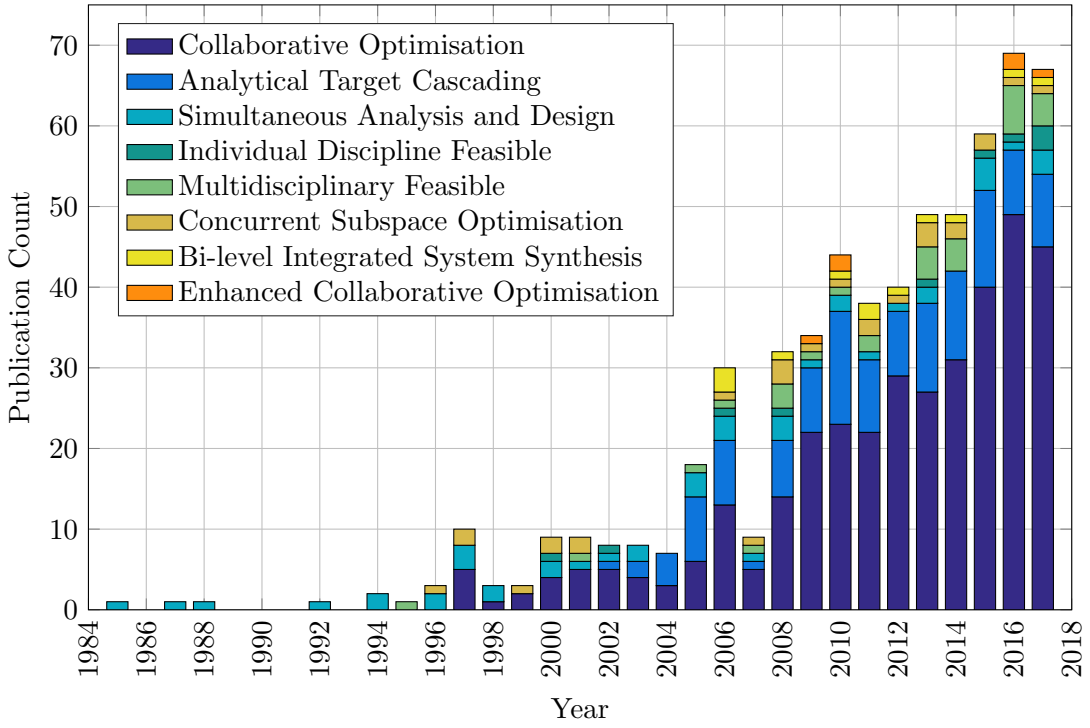

Figure 2.4: Yearly publication count for several MDO architectures (data sourced from http://apps.webofknowledge.com on 04/07/2018

2.4.2 A Note on Notation and Architecture Presentation

It is common practise in academia to describe MDO architectures in a mathematical format. Before reviewing some of the most popular MDO architectures, a generic problem notation is introduced to make the mathematical descriptions more digestible for the reader. In its most general form, the global objective can be formulated as a weighted sum of multiple domain objectives:

$$f_0 = \sum_{i=1}^{N} a_i f_i(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y}). \tag{2.11}$$

Lower level domains perform analyses that generate coupling (or also known as state) variables as given by:

$$\mathbf{y} = \mathbf{R_i}(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y}) \tag{2.12}$$

and/or domain local objectives.

$$f_i = R_i(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y}). \tag{2.13}$$

Each local objective may have a number of domain local constraints in the form of

$$\begin{aligned} \mathbf{g_i}(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y}) \leq 0 \quad for \quad i = 1, ..., N \\ \mathbf{h_i}(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y}) = 0 \quad for \quad i = 1, ..., N. \end{aligned} \tag{2.14}$$

The variables $\mathbf{x_s}, \mathbf{x_l}$ and $\mathbf{y}$ are vectors of domain local, shared and state variables, which are the inputs to the local analyses. These analyses can be simple analytical functions, dedicated black box routines or even experimental procedures, which are represented here by the symbols $\mathbf{R_i}$, $\mathbf{g_i}$ and $\mathbf{h_i}$.

It is unusual to find design problems without limits (or otherwise known as bounds) to the shared and local design variables. In the generic problem description, these bounds can be expressed as:

$$\begin{aligned} \mathbf{lb_s} \leq \mathbf{x_s} \leq \mathbf{ub_s} \\ \mathbf{lb_l} \leq \mathbf{x_l} \leq \mathbf{ub_l} \end{aligned} \tag{2.15}$$

where **lb** and **ub** denote vectors of the upper and lower bounds for the design variables **x**, and the subscripts **s** and **l** clarify if they apply to the shared or local design variables. It is also important to note that limits on state variables are actually constraints (as

given by Equation 2.14), as state variables are the outputs of analysis routines and not selected by a designer or optimiser.

Moving forward, it is also common for authors to provide graphical descriptions of their proposed MDO architectures. Box diagrams are the most common way to graphically represent new architectures [79, 86], however there has been recent a development on more sophisticated eXtended Design Structure Matrix (XDSM) diagrams [88, 98]. Although these offer much more detail and are less ambiguous than box diagrams, they can be intimidating to readers outside of the MDO community. In the following sections the mathematical formulation of each architecture, is combined within a box diagram graphical format to help illustrate the decomposition structure. Any additional notation specific to architectures will be given when necessary.

2.4.3 Simultaneous Analysis and Design (SAND)

Simultaneous Analysis and Design (SAND) is the oldest and also fittingly the first architecture in this review [51]. It combines all disciplinary analyses under a single optimiser. To ensure a consistent final design with respect to the state variables, the analyses used to generate them are treated as constraints functions. Copies of the state variables are put under the control of the optimiser, which tries to minimise the difference between the outputs of these functions and the copies that were generated. As all variables are under the control of a single optimiser, no distinction needs to be made between shared and local variables, which is why all design variables here are covered by the vector $\mathbf{x}$. Figure 2.5 shows the mathematical decomposition of this architecture, which is also inherently similar to single disciplinary optimisation.

Several aspects of the Simultaneous Analysis and Design architecture are noteworthy. Simple MDO problems can often be solved very quickly with a gradient-based optimisers [98]. This is assuming that the domain analyses are free from internal discontinuities. Where this is not the case, non smooth analyse can render the gradient optimiser ineffective. Although a gradient-free optimiser can be used instead, the convergence rate advantage over the distributed architectures tends tp be lost.

2.4.4 Review of Collaborative Optimisation (CO)

Collaborative Optimisation is one of the earliest distributed architectures. It was conceived in 1994 by researchers motivated to decompose multidisciplinary problems in a way that would fit the natural divisions of aerospace companies and their preferred methods of design [86]. A system-level optimiser minimises the global objective, while domain level optimisers try to reduce the disagreements between various disciplines. The main idea focusses around the concept of target variables.

Simultaneous Analysis and Design

minimise:

$$f_0 = \sum_{i=1}^{N} a_i f_i(\mathbf{x}, \mathbf{y})$$

subject to constraints:

$$\begin{aligned} \mathbf{y} - \mathbf{R_i}(\mathbf{x}, \mathbf{y}) &= 0 \\ \mathbf{g_i}(\mathbf{y}, \mathbf{x}) &\leq 0 \\ \mathbf{h_i}(\mathbf{y}, \mathbf{x}) &= 0 \end{aligned}$$

and variable bounds:

$$\mathbf{lb} \leq \mathbf{x} \leq \mathbf{ub}$$

Figure 2.5: Mathematical formulation of Simultaneous Analysis and Design

The model assumes that the system level optimiser contains the global objective function and each domain contains the analyses to generate the state variables. The system level optimiser sets targets for the shared and state variables ($\mathbf{x_t}$). Domains then try to match these targets my minimising a local consistency function J_i. During one system level iteration, each domain has multiple attempts to reach those targets. If these targets are unattainable, domains communicate this in the form of a constraint failure directly to the system level optimiser. This forces the system level optimiser to modify the targets over successive iterations until all domains are able to find the point where theit consistency function $J_i = 0$. This is illustrated in Figure 2.6.

Since the analyses are solely computed at the domain level, this architecture is particularly suitable for problems that do not have a natural hierarchical ordering, but rather a collection of equally important domains [8]. Furthermore this decomposition strategy eliminates communications between domains by channelling all analysis and design information through a system level optimiser. This has made it suitable for a variety of engineering applications including aircraft wing design [70, 87], launch vehicle design [22], supersonic aircraft [115] and unmanned aircraft [139].

Collaborative Optimisation has been extensively studied and modified over the years. In spite of the organisational advantages, several major shortcomings are observed in the mathematical formulation. A number of researchers showed that it suffers from slow convergence [4, 81, 120], as well as poor robustness [3, 86, 120] particularly when applied to problems which have a high degree of disciplinary cross coupling and a large number of shared variables. Alexandrov and Lewis [3] attributed these failures to the architecture's

Collaborative Optimisation

minimise:

$$f_0 = \sum_{i=1}^{N} a_i f_i(\mathbf{x_t})$$

subject to constraints:

$$\mathbf{J_i} = 0$$

Domain i

minimise:

$$J_i = (\mathbf{x_s} - \mathbf{x_t})^2 + (\mathbf{y} - \mathbf{x_t})^2$$

subject to constraints:

$$\begin{aligned} \mathbf{g_i}(\mathbf{y}, \mathbf{x_l}, \mathbf{x_s}) &\leq 0 \\ \mathbf{h_i}(\mathbf{y}, \mathbf{x_l}, \mathbf{x_s}) &= 0 \end{aligned}$$

and variable bounds:

$$\begin{aligned} \mathbf{lb_l} \leq \mathbf{x_l} \leq \mathbf{ub_l} \\ \mathbf{lb_s} \leq \mathbf{x_s} \leq \mathbf{ub_s} \end{aligned}$$

where:

$$\mathbf{y_i} = \mathbf{R_i}(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y})$$

Figure 2.6: Mathematical formulation of Collaborative Optimisation

problem decomposition. They observed that for certain problems the system level constraints can become non-smooth. Moreover, this feature was also confirmed by Tapetta and Renaud [145] in the multiobjective formulation of the architecture. This therefore hinders the application of most gradient-based optimisers, which explains why it has been observed to fail when used with gradient-based optimisers on certain problems. DeMiguel and Murrey [37] changed the format of the consistency functions J_i to address this problem. Their "Modified Collaborative Optimisation" architecture demonstrated better robustness using an exact penalty function in the system level constraints.

Overall, the main benefit of the original mathematical formulation is that it is relatively simple in the scheme of MDO architectures. It also has very few tunable parameters, which is perhaps why it continues to be widely used in academic comparison studies [79, 81, 115, 120, 146, 128].

2.4.5 Review of Analytical Target Cascading (ATC)

Michelena et al.[103] proposed the architecture termed Analytical Target Cascading, to enable system level performance targets to be cascaded through the organisational hierarchy of design teams in the automotive industry. The basic organisation of the architecture is identical to the Collaborative Optimisation method. Where Analytical Target Cascading differs, is in the mathematical formulation of the system level objective and constraint functions. Instead of the nested approach, a penalty function is introduced in the system level objective to drive the difference between domain local and target variables to zero. Here ϵ_x and ϵ_y make up penalty parameters that define the precision of the final convergences. This change was shown to improve the speed of convergence on tests performed by Roth [128] and Kim et al. [79] when compared to Collaborative Optimisation.

Figure 2.7 shows an earlier version of Analytical Target Cascading by Kim et al. [79]. Similarly to the previous architectures, Analytical Target Cascading has also undergone several refinements over the years. Perhaps the most notable derivative is a version by Tosserams et al. [151], which showed that the Karush Kuhn Tucker criteria can be satisfied using a Lagrangian penalty function. Other noteworthy variants explored a non-hierarchical problem decomposition [152] as well as modifications for multiobjective problems [71].

In theory, Analytical Target Cascading can be seen as a mathematical derivative of Collaborative Optimisation. The fundamental differences between the two is the presence of additional penalty parameters, which means that the subsystem analysis are disjointed from the system level optimiser, unlike the nested formulation used in Collaborative Optimisation [128]. This offers some benefits, primarily the freedom to cascade target variables down a hierarchy of design teams. However the original version by Kim et al. [79] requires users to select suitable penalty parameter which can be difficult in practice. Nevertheless this architecture has been applied to a wide variety of design problems including, but not limited to automotive [71, 72, 79, 93], aerospace [9, 152] and supply chain design optimisation [123].

2.4.6 Review of Enhanced Collaborative Optimisation (ECO)

The original formulations of Collaborative Optimisation and Analyticity Target Cascading restrict inter-domain communications and channel decisions about target variables solely through higher levels. In 2008, Roth developed a non-hierarchical MDO architecture called Enhanced Collaborative Optimisation (ECO) with the motivation to eliminate the majority of the numerical difficulties associated with Collaborative Optimisation and increase the influence of low level domains to better reflect the processes followed in industry [128].

Analytical Target Cascading

minimise:

$$f_0 \;=\; \sum_{i=1}^{N} a_i f_i(\mathbf{x_t}) \;+\; \sum \epsilon_x \;+\; \sum \epsilon_y$$

subject to constraints:

$$\begin{array}{rcl} \epsilon_{x_i} \;-\; (\mathbf{x_{s}}i \;-\; \mathbf{x_t})^2 & \leq & 0 \\ \epsilon_{y_i} \;-\; (\mathbf{y}_i \;-\; \mathbf{x_t})^2 & \leq & 0 \end{array}$$

Domain i

minimise:

$$J_i \;=\; (\mathbf{x_s} \;-\; \mathbf{x_t})^2 \;+\; (\mathbf{y} \;-\; \mathbf{x_t})^2$$

subject to constraints:

$$\begin{array}{rcl} \mathbf{g_i}(\mathbf{y}, \mathbf{x_l}, \mathbf{x_s}) & \leq & 0 \\ \mathbf{h_i}(\mathbf{y}, \mathbf{x_l}, \mathbf{x_s}) & = & 0 \end{array}$$

and variable bounds:

$$\begin{array}{rcccl} \mathbf{lb_l} & \leq & \mathbf{x_l} & \leq & \mathbf{ub_l} \\ \mathbf{lb_s} & \leq & \mathbf{x_s} & \leq & \mathbf{ub_s} \end{array}$$

where:

$$\mathbf{y_i} \;=\; \mathbf{R_i}(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y})$$

Figure 2.7: Mathematical formulation of Analytical Target Cascading

The core idea of the architecture is that design teams should control the objective function, rather than chase targets imposed by chief engineers (i.e. the system level optimiser). Instead the chief engineering team should try and bring the teams into agreement. Therefore in Enhanced Collaborative Optimisation, a system level optimiser minimises inconsistencies between the domains using the target variables $\mathbf{x_t}$, while low level domains minimise their portion of the global objective function.

The inter-domain communications occur in the form of constraint preferences, which are communicated across different disciplines. Mathematically, the system level optimiser is unconstrained and solely aims to minimise the disagreements between the domains.

The formulation of the domain objective function is substantially more complicated in comparison to Collaborative Optimisation and Analytical Target Cascading. It consists of a quadratic model of the global objective, a compatibility penalty function to reduce differences between shared and state variables and a set of slack functions, to ensure feasibility. Furthermore each domain includes additional linear constraint functions from other domains in addition to any domain local constraints. Here λ_C and λ_F are compatibility constants, which the user sets before starting the process, while the slack functions $\mathbf{s}$ are models of constraints from other domains.

Enhanced Collaborative Optimisation

minimise:

$$J = \sum_{i=1}^{N} \left((\mathbf{x_s} - \mathbf{x_t})_i^2 + (\mathbf{y} - \mathbf{x_t})_i^2 \right)$$

subject to constraints:

No Constraints

Domain i

minimise:

$$f_i = a_i f_i(\mathbf{x_l}, \mathbf{x_s}, \mathbf{y}) + \lambda_C (\mathbf{x_s} - \mathbf{x_t})^2 + \lambda_C (\mathbf{y} - \mathbf{x_t})^2 + \lambda_F \sum \mathbf{s}$$

subject to constraints:

$$\begin{aligned} \mathbf{g_i}(\mathbf{y}, \mathbf{x_l}, \mathbf{x_s}) &\leq 0 \\ \mathbf{h_i}(\mathbf{y}, \mathbf{x_l}, \mathbf{x_s}) &= 0 \end{aligned}$$

and variable bounds:

$$\begin{aligned} \mathbf{lb_l} \leq \mathbf{x_l} \leq \mathbf{ub_l} \\ \mathbf{lb_s} \leq \mathbf{x_s} \leq \mathbf{ub_s} \end{aligned}$$

where:

$$\mathbf{y_i} = \mathbf{R_i}(\mathbf{x_s}, \mathbf{x_l}, \mathbf{y})$$

Figure 2.8: Mathematical formulation of Enhanced Collaborative Optimisation

Applications of Enhanced Collaborative Optimisation have demonstrated a considerable computational advantage over Collaborative Optimisation and Analytical Target Cascading [127, 129], and there also exists a formal proof of convergence [128]. However the most notable drawback is the complexity of the objective and constraints functions within each domain. The architecture is shown in Figure 2.8 In its most basic form. In practice however, the full detailed strategy, which can be found in the Thesis by

Roth [128], is often needed for more complex problems. This is arguably why applications of Enhanced Collaborative Optimisation remain rare. These exist only two other publications with this architecture to the knowledge of the author [162, 164].

2.5 Towards Multidisciplinary Design Optimisation for Industry

Considerable progress has been made towards integrating industry tools into monolithic MDO frameworks. This has been aided by development in integration software systems, such as iSIGHT^{TM} and ModelCenter^{TM}, which enable users to integrate analysis tools using their optimisation algorithm of choice in a single framework. For example, Piperni et al. [116] combined Bombardier's own KTRAN CFD solver with Nastran^{TM} and the authors' own TWSAP structures analyses tools inside the iSight software. The subsequent coupled aero-structural optimisation was performed in a monolithic MDO environment. In later work [116], the authors increased both the number of design domains and the analysis fidelity in an attempt to extend the framework outside of the conceptual design stage. Although promising, all the disciplines in the design process remain under the control of a single optimiser, which is undesirable for the preliminary design stage.

The above survey of distributed methods is by no means exhaustive, but rather offers a snapshot of some of the MDO methods available in academia. In fact the most recent survey in the field, by Martins & Lambe [98], which summarises the already mentioned monolithic and distributed architectures, along with a number of others listed as follows: Concurrent Subspace Optimisation [141], Bi-Level Integrated System Synthesis [142], Quasi-Separable Decomposition [52], Exact and Inexact Penalty Decompositions [36] and MDO of Independent Subspaces [137]. All these architectures illustrate that there is an abundance of academic literature on the topic. Despite all this work, most of these have failed to routine secure applications in industry, which is perhaps the best indicator that more work and different is needed in this area [2, 98].

This is partly because researchers working on the "classical" distributed architectures have often devoted significant effort to improve the speed of convergence, while disregarding the organisational and cultural functionalities of their architectures [2]. The many derivatives of Collaborative Optimisation, and formal convergence proofs of Analytical Target Cascading [78, 150] and Enhanced Collaborative Optimisation [127] are examples of just that.

Furthermore, the distributed architectures covered here assume that designers are homogeneous agents that will make the correct choice in every decision [14]. In industrial design, this is certainly not the case. When developing MDO methods, one must not

lose sight that it will be human engineers (and not optimisers) monitoring domain analysis and controlling the overall process. Hence why the closest industrial application of a target-based method shares only a few similarities with the existing MDO literature [96]. That specific application was applied at Schlumberger, where top level engineers generated targets for the coupling variables using a sampling plan of simulation analyses. Low level teams would then perform exhaustive searches to find groups of satisfactory designs that met those targets as closely as possible. At the end, top level engineers would manually evaluate the possible designs and pick one to be carried forward for prototyping. The authors reported that this process enabled a more thorough search of the design space, while retaining the ability for top level engineers to pick the final design. Moreover the method demonstrated a way for low level domains to work concurrently and still satisfy top level performance targets. Yet because the existing design process at Schlumberger favoured prototyping over simulation, a direct comparison between the two methods was very difficult [96]. The key observation to make is that this method does little by way of formal numerical optimisation, which highlights the need for distributed MDO methods to take better account of organisational design.

Using the assessment from the work by Agte et. al [2], three groups of obstacles facing distributed MDO methods have been put together: technical, organisational and cultural. An architecture that meets the following specifications, some of which were compiled from an assessment of the works by [2, 45, 98, 115], would hopefully become more appealing for use in industry.

Technical functionalities:

1. Have a high probability of convergence to a feasible result.
2. Converges in a similar number of analysis evaluation as the current design method in use.
3. Enables automatic data transfer between domains.
4. Allows chief engineers to prematurely terminate the process and still obtain a feasible design.
5. Can be extended to include multi-objective and reliability-based optimisation.

Organisational functionalities:

6. Enables designers to use their preferred analysis tools and search methods.
7. Allows of interchange of analysis tools for various levels of fidelity.
8. Allows easy addition, removal and division of organisational domains.

9. Promotes a problem formulation compatible with the current organisational design structure.

Cultural functionalities:

10. Allows chief designers to monitor and guide the search process at a system level.
11. Enables domain level engineers to monitor progress in other disciplines.

These functionalities highlight that methods that are compatible with the existing industrial processes are more likely to be applied in practical design because they will have lower risks to implementation. Current state-of-the-art methods try to maintain the way low level domains carry out their design. The underlying assumption is that a domain generates inputs to an analysis by way of an optimiser or manually driven method, in an attempt to minimise a local objective or satisfy a constraint. Models that deviate from this assumption risks becoming organisationally unsuitable for industrial applications.

One recent advance in this area is a version of the Bi-Level Integrated System Synthesis currently under development at Institute of Technological Research (IRT) Saint Exupery [45]. A top level optimiser controls the shared design variables and domain level optimisers set the values for the local variables. Otherwise the internal procedures by which domains perform design is kept the same. Over successive iterations between the two levels, Gazaix et al. [45] showed that the process is able to converge to a feasible optimal result when applied to academic and industrial test cases. One of the main disadvantages of this method however, is that the shared variables are exclusively controlled at a system level. This means that domains that only use shared variables will have limited control over the search process and will be at the mercy of the top level optimiser.

2.6 Design Space Reduction Frameworks

A broader search of the literature on distributed MDO yields many alternatives to the system level optimiser including the use of classifiers [5, 135] and game theory approaches [91, 165]. This section however focuses on design space reduction methods that are considered more suitable for industrial application. Design space reduction frameworks use a controlling algorithm to iteratively move and/or reduce the bounds on the design variables until they concentrate in a region small enough to represent a single design.

The first of the two surveys by Wujek and Renaud [161] examine a various different "Move-Limit Management Strategy". In many ways this is a survey of trust region methods, because it covers the bound move strategies controlling only approximations

of the functions to be optimised. The focus of their work however shifts to an application to MDO in the second survey [160]. There the authors apply the bound movement strategies as the top level optimiser in the Concurrent Subspace Optimisation architecture. The findings from the case studies concluded that strategies that rely solely on gradient information or that use fixed bound reduction are inadequate for solving MDO problems decomposed in the Concurrent Subspace Optimisation format.

Shahan and Seepersad [134], Price et al. [120] and Hannapel et al. [53] all proposed MDO frameworks that reduce the design space by changing the bounds on the shared design variables. Unlike the architecture by Gazaix et al. [45], organisational domains are free to alter both their shared and local variables so long as the proposed designs lie within the search space that is governed by the system level controller. Simply put, these methods can be viewed as distributed MDO architectures, where the domains are free to optimise their specific engineering objectives (such as drag, mass, cost, etc..) and a system level controller coordinates the process using the bounds on the shared variables. Overall bound changes are governed by the outputs from the disciplines at each iteration.

Hannapel et al. [53] and Hannapel [54] used gradient-based optimiser to alter the bounds on the shared design variables, whereas Price et al. [120] developed a set of rules to control the bounds. The method by Price et al. [120] was successfully able to find the minimum when applied to several academic and aerospace design test cases. However for problems with a high number of shared variables, the rule base took significantly more analysis evaluations than several competing MDO architectures. The method by Hannapel et al. [54] on the other hand was only applied to a ship design problem. Although the authors do not offer a comparison with alternative MDO methods or comment on the rate of convergence, they compare the method with one that uses independent optimisation in series and conclude that it is valuable to utilise the space reducing technique before approaching the problem with a single point-based method.

Shahan and Seepersad [134] build an opinion poll in the form of a probability density function for each shared variable. By overlapping these probability density functions, the authors were able to mathematically represent the design preferences from each domain and therefore contract the bounds in areas where the probability is lowest. The authors demonstrate the method on a UAV design problem, but fail to give a comparison with a competing distributed MDO method. Furthermore the chosen test problem is unconstrained and therefore it is difficult to establish if the information from probability density functions would be sufficient to satisfy contained problems.

The methods of Shahan and Seepersad [134], Price et al. [120] and Hannapel et al. [53] are standalone in the field of MDO and seem to have no continuation. This may be because they have not reached maturity due to the earlier discussed failures. Nevertheless,

such approaches remain appealing primarily because they keep the preferred analysis and search routines with the various organisational domains.

The work by Ollar et al. [110] is perhaps the most recent development in the area and is an application of a trust region optimisation method applied to MDO problems. Because at its core it uses the monolithic MDO architecture, it is not directly comparable to the other three methods already discussed in this section, but none-the-less it is worth noting. The proposed framework uses the Multidisciplinary Feasible architecture with the mid-range approximation method [148, 149] as the optimiser of choice. This effectively means that the optimiser iteratively reduces the bounds on selected design variables at a top level. At a subspace level however, a sampling plan is used to build surrogate models of the individual domain's responses. The authors demonstrate that the number of analysis evaluations can be reduced using this method, provided that the design has made correct assumptions on which variables are significant for each domain.

2.7 Architecture Benchmarking Challenges

When faced by so many different options for distributed MDO, academics need ways to assess emerging MDO architectures. They often apply their own architecture on test cases and compare its performance against other established MDO architectures. The most often used metrics for comparison are efficiency - how many analysis evaluations does it need to converge, and accuracy - how close is the outcome to the (usually) known global minimum.

There are many factors which influence how quickly and accurately a given architecture will converge. The following list includes some of those: chosen problem, problem decomposition, optimisers used, optimiser settings, starting design, starting point strategy, dissimilarities in programming style, chosen internal architecture tuning parameters and for methods that use surrogates models, a number of factors related to the type of surrogate, data selection and tuning. This makes the process of comparing new, against well-established alternatives extremely vulnerable to unintentional biasing. There exist specific benchmarking frameworks, such as iSight, ModelCenter, pyMDO[99], OpenMDO [55], that allow MDO problems to be defined once and reused across multiple architectures. However because optimisers and surrogate models may serve different purposes in the architectures, the use of such frameworks does not guarantee a fair comparison. It is also worth noting that they require considerable additional effort for academics unfamiliar with their use.

In general, few authors can replicate the exact results provided by others in comparison studies. At best the results capture generic trends, but there are cases where they disagree. For example Tedford and Martins [146] observed 8601 fewer analysis calls than Perez et al. [115] when Collaborative Optimisation was applied on an identical

problem. On the other hand, the paper by Perez et al. presented 1382 fewer analysis calls in their application of Concurrent Subspace Optimisation in comparison to Telford and Martins. Similar differences can be also found in other publications [3, 10, 26, 81, 83, 99, 120], which is why accuracy and efficiency can be a misleading metrics when ranking architectures.

It is also not possible to fully compare architectures with the accuracy and efficiency metrics alone. A number of researchers have therefore ranked architectures using qualitative metrics [3, 115, 168]. Yi et al. [168] used a "Required Information" metric to describe complexity of a given architecture. Perez et al. [115] proposed three alternative qualitative metrics. Simplicity describes the ease of implementing on a problem and is measured by the total number of optimisation variables. Transparency is a metric that describes the ease with which an outsider can modify the architecture to include capabilities such as probability-based analysis and multiobjective formulation. It is derived by the subjective view of the mathematical complexity of an architecture. The third and final metric, portability, quantifies how easily an architecture could be implemented in an existing organisational design structure. Although much more encompassing and arguably more important, such assessments are only valid if they are made by independent researchers not involved in development of the architectures being tested.

These difficulties motivated some developers to forgo the classical benchmarking techniques and instead to compare their proposed methods against a model of the sequential design process [27, 53]. This presents a different challenge because to accurately model the design process is not easy. Both Hannapel et al.[54] and Chittick and Martins [27] use a model of a design process that optimises one domain after the other, but their test problems have constraints only in one of the two domains. Such comparisons would be difficult on problems where more than one domain has internal constraints as there would be no formal mathematical way to communicate constraint information across domain boundaries. Besides, these models also fail to capture the experience that industrial designers have, which ultimately risks under representing the abilities of real-world design teams.

2.8 Concluding Remarks

The preliminary aircraft design process is often defined as sequence of analyses, performed by various organisational domains one after the other. Current methods largely rely on the designer's experience to foresee how changes in one domain, may affect others down the chain. Because this requires considerable human input, there is a shared consensus in the research community that this process can be improved by using dedicated MDO architectures, which can better exploit the interactions between the various organisational domains [27].

Monolithic MDO architectures have matured and seen increasing application on industrial problems. Although they can be very efficient and robust, they are less suitable for organisationally dispersed design problems, such as ones facing engineers in the preliminary aircraft design stage. There is an abundance of more suitable distributed MDO architectures that claim to emulate the industrial design processes, however, the review presented here highlighted numerous drawbacks and obstacles to implementation.

Despite the growing number of state-of-the-art distributed MDO architectures, there is evidence that aircraft manufacturers continue to use well-established sequential design processes [50, 106, 125]. This leads to the conclusion that many of the existing distributed MDO methods and architectures fail to work in industry simply because they do not fit the organisational structure of the company [2]. A MDO method that deviates too far from the established processes, can be perceived as risky and therefore deter designers from using it. Current state-of-the-art methods seek to incorporate a MDO strategy by using the existing practices used by design teams. Many are still under development or have not reached the level of maturity to allow for industrial application. This highlights an interesting opportunity to develop a MDO framework that is more industrially suitable than existing methods.

Out of all the methods reviewed in this chapter, the work by Price et al [120] is the most promising avenue for further research. Although it sacrifices any formal guarantees for global convergence, its formulation offers engineers an intuitive mechanism to control multidisciplinary domains. The following chapters describe progress in this area and how the work has addressed the gaps in the literature.

Chapter 3

The Blackboard Framework

3.1 Introduction

Aircraft design requires the cooperation of a large number of individuals with expertise in different fields. How to get these individuals to cooperate in the face of conflicting objectives and also enable a concurrent working environment has been the subject of substantial recent research. Current industrial practices deal with these conflicts via regular face to face meetings involving chief engineers representing the various organisational domains. Each representative then attempts to influence decision makers towards their optimum by presenting engineering analysis or historical experience to support their design solution [106]. The growing number of sophisticated digital analysis, storage and communication tools has motivated research in so called "blackboard frameworks" [41, 58, 107, 157]. These share numerous similarities with the classical MDO architectures and have been previously proposed for use on concurrent engineering and distributed design problems.

Nii [107] described the generic blackboard model as a collection of three elements: knowledge sources, a controller and a database. Much as in distributed MDO problems, the model assumes that individual knowledge sources (the organisational domains) are incapable of solving the problem alone, but instead must cooperate with others to reach a solution. Communication between domains is done solely via the database. Each domain periodically updates the database with information that they have generated. Equally, other domains can extract information from the database, which they require for individual analyses. The database in other words facilitates the exchange of analysis data between domains. The entire process is managed by a controlling algorithm, which periodically evaluates the information in the database and guides the knowledge sources to a feasible solution[107].

This generic description offers nothing more than guidance for developing blackboard frameworks, without a step-by-step recipe for its application on specific problems. Price

et al. [120] used this concept to formulate a design space reduction framework that was shown to solve MDO problems. Figure 3.1 illustrates how this would typically work for a wing design problem, while the process can be described in the following phases:

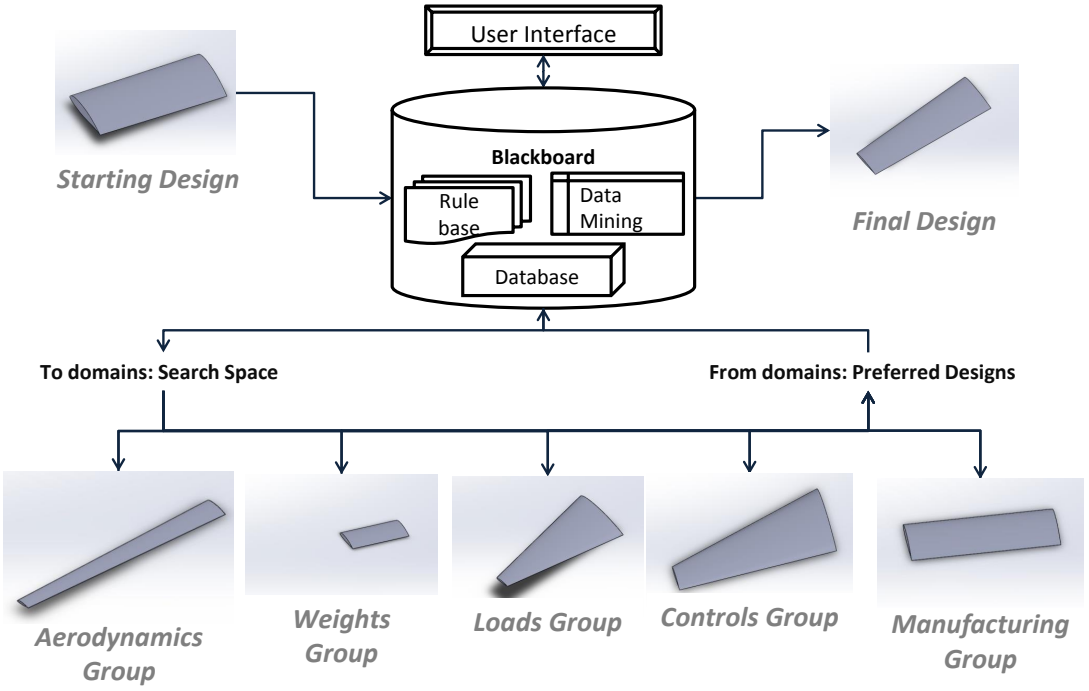

Figure 3.1: Blackboard framework layout for MDO (adapted from Price et. al [120])

Phase 1: Chief engineer selects a common starting design and defines the available search space in the form of upper and lower variable bounds.

Phase 2: Engineering teams (the knowledge sources) can then explore the available design space using their preferred optimisation and analysis methods. They are free to develop what they consider to be an *optimal* design in relative isolation from all other domains.

Phase 3: As domains are analysing and altering their preferred designs, they populate a central repository (a database), which is accessible to all other domains. Therefore when one domain requires coupling information from another, they can simply query the database and obtain the latest design status from there.

Phase 4: At the end of a search, each domain uploads their preferred design to the database.

Phase 5: A rule set then evaluates the uploaded designs and proposes new upper/lower bounds on the shared design variables accordingly. The process is repeated from *Phase 2* until the available design space is so small that it can be assumed to represent a single final design. This convergence can be illustrated using a two variable MDO problem as shown in Figure 3.2.

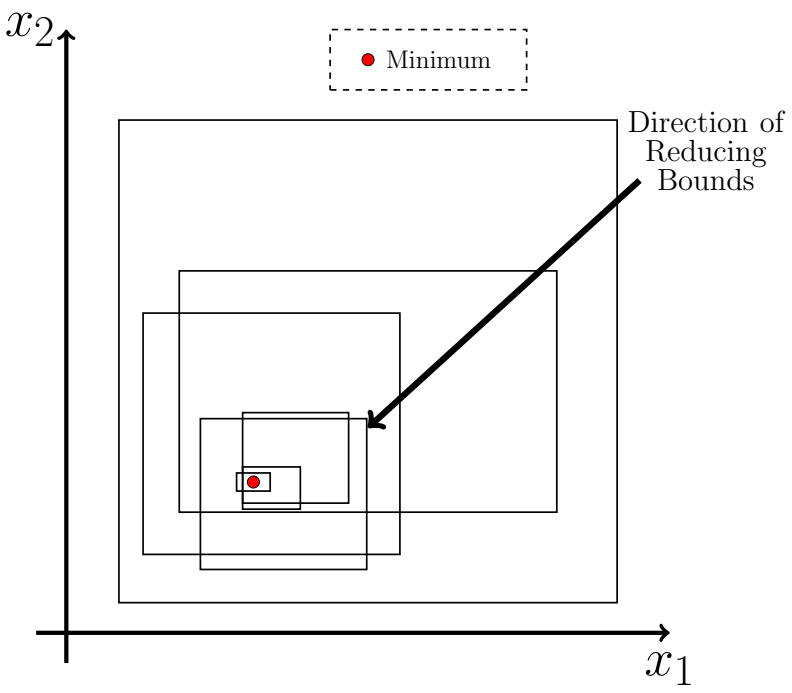

Figure 3.2: Bound reduction strategy illustrated on a problem with two shared variables

This method can be viewed as a distributed MDO architecture, where organisational domains optimise their specific engineering objectives and a system level blackboard coordinates the process. As already discussed in Section 1.2, it is often preferable to narrow down a design from multiple good solutions, rather than iteratively change a single design. These attributes make the blackboard process particularly suitable for problems in the early preliminary design stage where a target concept is already selected and parametrised, but the values for the key geometric variables are not yet fixed, just enclosed by upper and lower bounds.

The three main elements in the blackboard serve separate functions. The rule base plays the role of the chief engineer and contracts the bounds towards a design minimum. The user interface allows chief engineers to monitor the process and steer low level domain searches. And finally, the database enables information to flow between the various domains involved in the design process. Therefore a transition from the present process that uses independent optimisation in series to the proposed concurrent MDO method should in theory be straight forward, because the inner structure of the design domains remains the same, while already existing data communication, storage and management protocols can be easily adapted to accommodate the blackboard.

While the original framework described in the reference is sound, it lacks a user interface and data mining modules, and its rule base suffered from a number of problems. The original rule base made limited use of the history of previously evaluated solutions, which meant it was slow to converge on problems with multiple shared variables [120].

Furthermore, the legacy rules were challenging to comprehend and can be shown to fail on problems outside those tested in the original publication [64]. Moreover, Price et al. [120] offered limited guidance with respect to suitable magnitudes for of bound contraction/movement at each step. This work aims to address many of these shortfalls by proposing a the newly developed rule set, which has been given the name "Multidisciplinary Patten Search" (MDPS).

3.2 The Multidisciplinary Pattern Search (MDPS)

3.2.1 Introduction

The Multidisciplinary Pattern Search manipulates the domain level optimisations by changing the bounds on the shared design variables. Its primary role is to automatically evaluate a number of locally optimal designs and set the search space for the following iteration. The rules that govern the bound changes have gone through an extensive development over the course of this research project. An early version of the algorithm [64] used either multidisciplinary feasibility or optimality as the deciding factor which guide the bound movements to a singular point. This final version of the algorithm however shares a number of key concepts with two legacy pattern search optimisers from Section 2.2.3: the Hooke and Jeeves and Tabu searches. Both these methods are based on sound and well tested heuristics and have a well established reputation for being simple and robust [92], which is why their logic makes up the foundation of the Multidisciplinary Pattern Search. Unlike the legacy optimisers which only set the next evaluation point, the Multidisciplinary Pattern Search controls the position of the design space described by the upper and lower bounds. This requires the original logic to be modified, with the differences illustrated by Figures 3.3 and 3.4.

The description that follows contains a number of independent sections to help explain the logic and draw parallels where possible with the legacy optimisers described in Chapter 2. Table 3.1 summarises the main rules, while Figure 3.5 illustrates the process structure.

3.2.2 The Global Fitness Function

The Multidisciplinary Pattern Search calculates a global objective from the various output designs of the individual domains. So strictly speaking the global objective comprises of a set of disjointed locally optimal designs from each domain, which are used as an assessment of a search region, rather than an evaluation of any single design as in the conventional optimisation terminology.

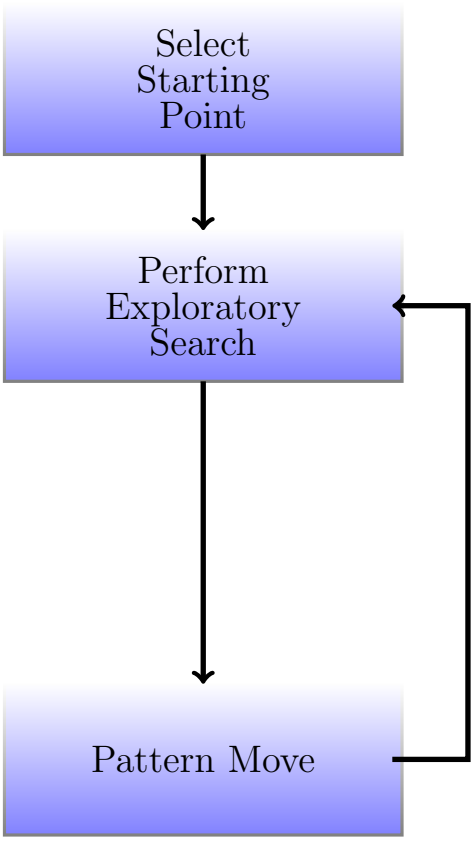

Figure 3.3: Actions in the Hooke and Jeeves pattern search

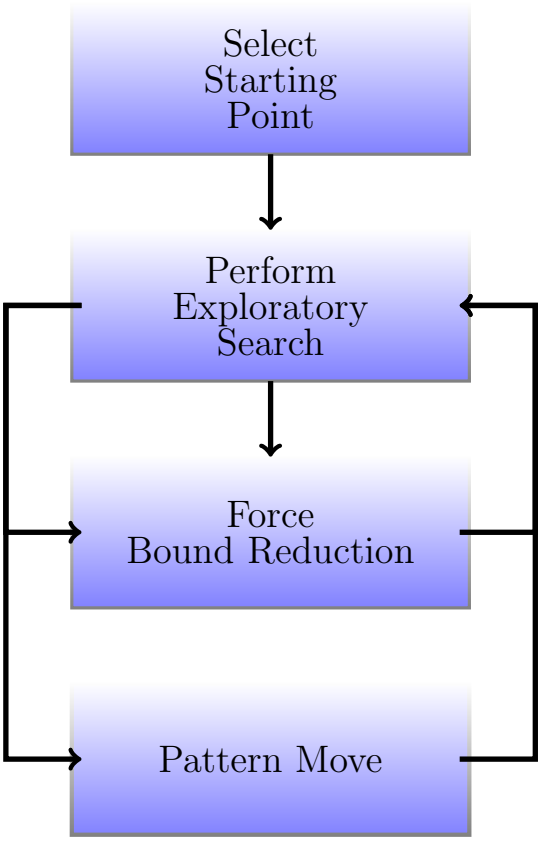

Figure 3.4: Actions in the Multidisciplinary Pattern Search

The global objective function is assumed to consist of a weighted sum of the domain local objectives, as given by $f_0 = \sum a_i f_i$. This type of objective function is commonly found in academic multi-objective optimisation test cases [143], yet the author is aware that in practical design the choice of weights for each objective can be rather arbitrary. So an alternative approach may use a goal-based objective function, which aims to minimise the difference between a user desired target and the actual output objectives. Numerical experiments concluded that this function can also work equally well with the Multidisciplinary Pattern Search, provided that the user defined targets are well selected. The function format in this case would be $f_0 = \sum (f_i - Q_i)^2$, where Q_i are user defined objective targets, while f_i and a_i are the local objectives and their associated weightings. Because this function forgoes the measure of global optimality, it has not been used further here.

The global function includes a separate penalty to deal with constraint violations. These often occur if a domain is unable to find a feasible design in the available search space that has been set by the Multidisciplinary Pattern Search. The review of constraints handling techniques in Section 2.2.3.4 concluded that the choice of penalty function is largely at the discretion of the developer, which is why the following penalty function that worked well for the problems at hand was used here.

Table 3.1: Summary of main rules and actions in the Multidisciplinary Pattern Search

Rules	Actions
R1 - Bound scope is below the current convergence factor?	**Yes** - Lock bounds on variables that have converged and proceed to **R2**. **No** - Perform exploratory search.
R2 - Are bounds on all variables locked?	**Yes** - Reduce the convergence factor; unlock variable bounds and perform exploratory search. **No** - Perform exploratory search.
R3 - Exploratory search resulted in *success**?	**Yes** - Set move as base point, perform pattern move and proceed to **R4**. **No** - Reject move and force bound contraction for all unlocked variables.
R4 - Pattern move resulted in *success**?	**Yes** - Set move as base point and proceed to **R5**. **No** - Reject move and proceed to **R5**.
R5 - Is the current convergence factor below the minimum convergence factor?	**Yes** - Terminate Search. **No** - Proceed to **R1**.

**success* denotes an outcome that improves the objective function or reduces any excessive constraints violations.

First a scaled constraint failure is calculated:

$$c_{i_{FAIL}} = \frac{1}{\tilde{m}}(g(x) - \tilde{m}) \tag{3.1}$$

$$c_{i_{FAIL}} = \frac{1}{\tilde{n}}|h(x) - \tilde{n}| \tag{3.2}$$

for constraint functions in the form of

$$h(x) - \tilde{n} = 0 \quad \text{and} \quad g(x) - \tilde{m} \leq 0. \tag{3.3}$$

Here $\tilde{m}$ and $\tilde{n}$ are the constraint limits and $g(x)$ and $h(x)$ are the analyses that compute the constraints. A penalty proportional for each constraint violation is then calculated from

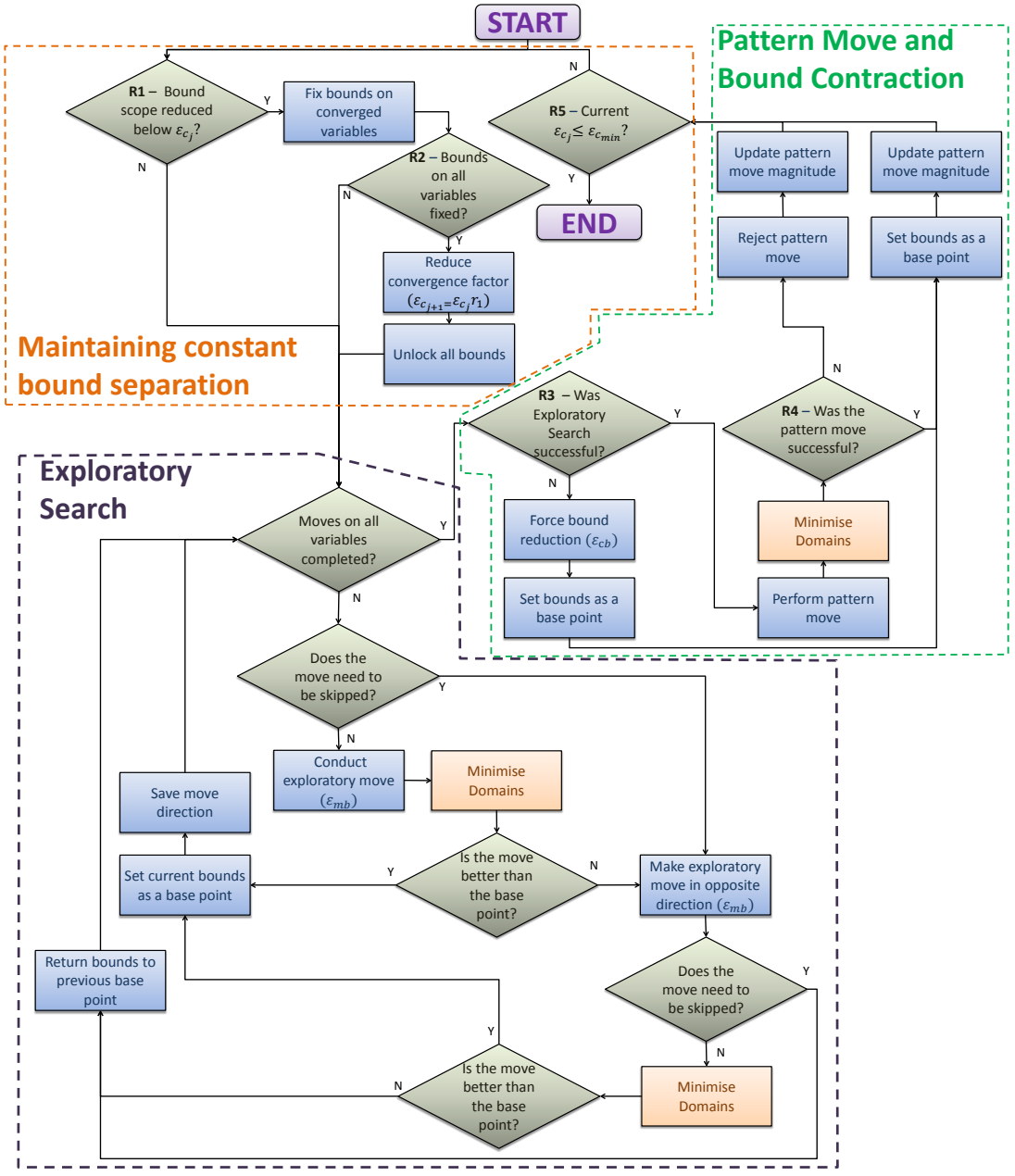

Figure 3.5: Process flow for the Multidisciplinary Pattern Search

$$p_i = \sqrt{|c_{i_{FAIL}}|}. \tag{3.4}$$

When a constraint violation occurs the following logic modifies the global objective function.

$$f_0(\mathbf{ub_s}, \mathbf{lb_s}) = \begin{cases} f_0 & \text{if, } \forall c_{i_{FAIL}} < 0 \\ f_0 + \sum_{i=1}^{N} p_i & \text{if, } 0 < \forall c_{i_{FAIL}} \leq 0.01 \\ max(p_i) & \text{if, } 0.01 < \forall c_{i_{FAIL}} \end{cases} \tag{3.5}$$

This complexity is introduced to avoid the condition where feasible designs might be rejected in favour a one which is infeasible, but has a better objective value. Of course for Equation 3.5 to work best, f_0 should be scaled using the weighing parameters somewhere in the range of $-1 \rightarrow 1$.

3.2.3 The Neighbourhood Search

In its most basic form, the Multidisciplinary Pattern Search uses the neighbourhood search logic with several additional rules that exploit the history of already evaluated solutions. Although its logic is equivalent to the description given in Section 2.2.3.2, it differs in that a perturbation describes a move of both upper and lower bounds in a given direction, rather than a change in design variables. At the start both bounds on each shared design variable are moved together and by an equal distance either upwards or downwards, effectively changing the searchable design region for the domains. Each domain then independently searches this new region and returns a point that minimises their local objective and satisfies their local constraints. The term *preference* describes this local domain minima, which, in the context of a real world design problem can be obtained from an optimisation search, a single domain analysis or even an accurate engineering guess. This means that local domains are not required to use any formal optimisation at all, which should appeal to industrial aircraft designers where trade studies are often used in favour of more sophisticated optimisation methods.

In this context, an iteration of the multidisciplinary neighbourhood search terminates when all possible bound movements have been exhausted. In the event that success was observed, a sole pattern move follows or otherwise a bound contraction follows. The bound contraction forces reduces the searchable design region and ensures that the algorithm converges. Often however, this causes short term deterioration in the objective function that can be corrected by the successive exploratory search. The rules do not allow for a design space expansion, as the main philosophy in use here is that the bounded space should gradually shrink as the design approaches the optimum. Alone the combination of the multidisciplinary neighbourhood search rules with bound contraction can converge MDO problems to a user desired final bound spacing. Yet much like the single disciplinary case, they are inefficient without the additional pattern move and data mining checks.

3.2.4 The Pattern Move

When an exploratory search generates an improvement across some or all of the variables, the direction the bound movement that generated the improvement is saved and stored in the form of a pattern move. When all variable directions have been exhausted, a pattern move is applied in the direction of previous successful moves, in the hope that it will bypass what might otherwise be several exploratory searches following the same path. This logic is very similar to the pattern move by the Hooke and Jeeves search.

By how much the pattern move alters the bounds depends on the success rate of previous neighbourhood searches. At first the bound movement magnitude equals that of the original move, but as pattern moves are accepted, the movement distance doubles after every successful application. If the outcome of the pattern move is rejected, the pattern bound movement factor is reset. For its application in the wider scope of the algorithm, see rules R3 and R4 in Table 3.1.

3.2.5 The Rule-Based Data Mining

The generic blackboard model gives the Multidisciplinary Pattern Search access to a database, which can be used to extract information of previously evaluated design regions. The notion of using short and long term memory to prevent certain moves from firing has been applied here in order to speed up convergence.

3.2.5.1 Duplicate Rule

Duplicate evaluations of the same design region often occur after two successive neighbourhood searches. To prevent this, every design region is encoded and stored in the database for later use by a duplicate check rule. Before each new region is evaluated, the duplicate check queries the database and bypasses a move that has already been evaluated. This rule is very similar to the short term memory application in the Tabu search [30] and updated version of the Hooke and Jeeves search purposed by Vazquez et al. [154].

3.2.5.2 Bell and Pike Rule

The Bell and Pike [15] proposal on the original Hooke and Jeeves search was also implemented here. The rule stores the direction of a successful the bound movement on a variable, so that it can be used as the primary search direction for the next neighbourhood search. This rule assumes that the objective function follows a set movement direction and it is more likely to observe success and kept without evaluating the opposite direction. Figure 3.5 shows how this rule is implemented here.

3.2.5.3 The Tabu Rule

Towards the later stages of the neighbourhood search, the same bound movements tend to cause constraint violations over consecutive iterations. A rule, similar to that used in the Tabu search algorithm has been added to identify the bound moves that consistently break the constraints by a significant margin. More specifically, the rule identifies when a move causes two consecutive constraint failures over previous iterations.The rule skips these "Tabu" moves over the next three neighbourhood searches, giving enough time for the search procedure to settle to an area where the move can be reinstated.

On a side note, this opportunity to avoid recalculation of moves that are likely to produce high constraint violations has not been implemented in a single disciplinary search algorithm to the author's knowledge. Therefore this simple rule can be introduced to the single disciplinary optimiser, such as the Hooke and Jeeves search for instance, as a means to further improve the rate of convergence on constrained problems.

3.2.6 Additional Bound Checks

The generic description of the Multidisciplinary Pattern Search covers the reasoning behind the main bound control rules and checks. Without a number of additional rules, unguided bound reductions and movements can cause the algorithm to exhibit unwanted behaviour.

3.2.6.1 Constant Bound Separation

Figure 3.5 shows the first set of additional checks under the term "Constant Bound Separation". Successive bound movements in one direction can cause the search space on a variable to be pushed towards one of the absolute limits. This can lead premature tightening on the bounds on certain variables and thus ultimately trap the algorithm in an infeasible search space. To avoid this, the algorithm uses an outer loop to maintain a constant bound separation among all variables. The process locks the bounds on the variables that reach a predetermined bound separation, thus preventing further bound contractions and movements. When all variables reach this bound separation, the bounds are unlocked and the threshold for bound separation is reduced. Rules R1 and R2 in Table 3.1 cover this logic.

3.2.6.2 Pre-Optimisation Checks

Before the Multidisciplinary Pattern Search sets a new space to the various domains, three rules check the feasibility of the design space. They ensure that the new bounds

remain within the absolute values set by the user and the starting point for each domain falls inside the new search space. Table 3.2 summarises these checks.

Table 3.2: Summary of the rules and actions before a search space is communicated to the domains

Rules	Actions
R6 - An upper or lower bound is outside of its absolute value?	**Yes** - Make the exceeding bound equal to the absolute value and proceed to **R7**. **No** - Continue to rule **R8**.
R7 - Has the bound change caused the lower bound to exceed the upper bound or is the bound spacing below the current minimum bound spacing?	**Yes** - Modify the bounds so they have a bound spacing equal to the current minimum bound spacing. **R3**. **No** - Continue to rule **R8**.
R8 - Does the current starting point for all domains falls inside the bounds?	**Yes** - Proceed with local domain optimisation. **No** - Move the starting point so they fall inside bounds and proceed with local domain optimisation.

3.2.7 Control Factors

The original Hooke and Jeeves search had two control factor - the initial step size and the step reduction factor. The additional complexity of moving design variable bounds requires several more control factors in the Multidisciplinary Pattern Search. This section describes five factors that can be set by the user to control the algorithm convergence. These are the starting convergence factor $\epsilon_{c_{init}}$, the final convergence factor $\epsilon_{c_{min}}$, the factor reduction value r_1, the bound contraction factor ϵ_{cb} and the bound movement factor ϵ_{mb}. To help with the explanation, the term bound separation is defined to mean the difference between the upper and lower bounds. The convergence factor (ϵ_{c_j}) is a term used to describe the ratio between the current and the starting bound separations. So for a single variable, the convergence factor can be given by Equation 3.6, where ub and lb are the current ub_{init} and lb_{init} denote the initial bound limits.

$$\epsilon_{c_j} = \frac{ub_s - lb_s}{ub_{init} - lb_{init}}. \tag{3.6}$$

Earlier in Section 3.2.6.1 described why all variables need to maintain a constant bound spacing during a given iteration. The convergence factor ϵ_{c_j} acts as a threshold for

maintaining this constant bound separation. In practice, it is reduced by the reduction value r_1 in an outer loop over successive iterations. Hence why the algorithm needs the additional starting convergence factor $\epsilon_{c_{init}}$ and the relaxation value r_1 to reach the final convergence factor $\epsilon_{c_{min}}$. The final convergence factor $\epsilon_{c_{min}}$ describes the desired ratio of initial to final bound spacing that will terminate the search. For example $\epsilon_{c_{min}} = 0.01$ will terminate the Multidisciplinary Pattern Search when the current bounds reach 1% of the starting bound spacing. Users can set these as they desire, but values around 0.5 were observed to produced good results for both $\epsilon_{c_{init}}$ and r_1. The bound movement ϵ_{mb} and contraction ϵ_{cb} factors control the magnitude of bound movements and contractions. Both move or contract the bounds proportionally to the current bound spacing. Hence for example, a bound movement factor of 1 or higher will move the bounds in either direction with no overlap, whereas a bound contraction factor of 0.5 will narrow them by 50%. Tests in Section 4.2.3 show that a contraction factor between 0.2-0.5 and the movement factor of 0.5 can produced good results.

3.2.8 Pseudo-Code of the Multidisciplinary Pattern Search

This section covers the mathematical control of the factors and mound movements in the form of pseudo-code. Illustrated within are the two nested iterations loops of the algorithm. The outer loop consists of the "Bound Convergence", "Algorithm Convergence" and "All Bounds Locked" checks, which maintain constant bound separation and terminate when the desired bound separations is reached. The remaining rules and moves cover the inner iteration loop, which searches the design spaces.

Algorithm Convergence Check (R5):

if $\epsilon_{c_j} \leq \epsilon_{c_{min}}$

Terminate search.

else

Proceed to **R1**.

end

Bound Convergence Check (R1):

if any $(\mathbf{ub_{s_j}} - \mathbf{lb_{s_j}}) \leq \epsilon_{c_j}(\mathbf{ub_{init}} - \mathbf{lb_{init}})$

Lock bounds on variables whose bound spacing is below threshold and continue to **R2**.

else

Continue to ***Exploratory Search***.

end

All Bounds Locked Check (R2):

if all variables are locked

Unlock all variables and reduce the current convergence factor by:

$$\epsilon_{c_{j+1}} = \epsilon_{c_j} r_1. \tag{3.7}$$

else

Continue to ***Exploratory Search****.*

end

Exploratory Search:

for all shared variables that have remained unlocked

Step 1 *Move bounds in the direction that previously resulted in success:*

$$ub_{s_{j+1}} = ub_{s_j} \pm \epsilon_{mb}(ub_{s_j} - lb_{s_j}), \tag{3.8}$$

$$lb_{s_{j+1}} = lb_{s_j} \pm \epsilon_{mb}(ub_{s_j} - lb_{s_j}). \tag{3.9}$$

Step 2 *Check if move is duplicate or Tabu and needs to be skipped.*

Step 3 *Apply the additional pre-optimisation checks.*

Step 4 *Minimise local domain objectives* f_i*.*

Step 5 *Evaluate the augmented global objective function according to:*

$$f_0(\mathbf{ub_{s_{j+1}}}, \mathbf{lb_{s_{j+1}}}) = \begin{cases} f_0 & \textbf{if, } \forall c_{i_{FAIL}} < 0 \\ f_0 + \sum_{i=1}^{N} p_i & \textbf{if, } 0 < \forall c_{i_{FAIL}} \leq 0.01 \\ max(p_i) & \textbf{if, } 0.01 < \forall c_{i_{FAIL}}. \end{cases} \tag{3.10}$$

if $f_0(\mathbf{ub_{s_{j+1}}}, \mathbf{lb_{s_{j+1}}}) < f_0(\mathbf{ub_{s_j}}, \mathbf{lb_{s_j}})$

Set current bounds as base point by:

$$f_0(\mathbf{ub_{s_j}}, \mathbf{lb_{s_j}}) = f_0(\mathbf{ub_{s_{j+1}}}, \mathbf{lb_{s_{j+1}}}), \tag{3.11}$$

$$\mathbf{ub_{s_j}} = \mathbf{ub_{s_{j+1}}}, \tag{3.12}$$

$$\mathbf{lb_{s_j}} = \mathbf{lb_{s_{j+1}}}. \tag{3.13}$$

else

if *this is the first move direction*

Repeat **Step 1** *to* **Step 5**.

else

Move to next unlocked variable.

end

end

end

Exploratory Search and Pattern Move Success Checks (R3 and R4):

if the **Exploratory Search** results in success:

Perform ***Pattern Move*** *on unlocked variable.*

else

Force bound contraction only on unlocked variables by:

$$\mathbf{ub_{s_{j+1}}} = \mathbf{ub_{s_j}} - \frac{\epsilon_{cb}}{2}(\mathbf{ub_{s_j}} - \mathbf{lb_{s_j}}), \tag{3.14}$$

$$\mathbf{lb_{s_{j+1}}} = \mathbf{lb_{s_j}} + \frac{\epsilon_{cb}}{2}(\mathbf{ub_{s_j}} - \mathbf{lb_{s_j}}). \tag{3.15}$$

Minimise local domain objectives f_i.

Set current the augmented global objective function as base point:

$$f_0(\mathbf{ub_{s_j}}, \mathbf{lb_{s_j}}) = f_0(\mathbf{ub_{s_{j+1}}}, \mathbf{lb_{s_{j+1}}}), \tag{3.16}$$

$$\mathbf{ub_{s_j}} = \mathbf{ub_{s_{j+1}}}, \tag{3.17}$$

$$\mathbf{lb_{s_j}} = \mathbf{lb_{s_{j+1}}}. \tag{3.18}$$

end

Pattern Move:

Move bounds in the directions of successful moves by:

$$\mathbf{ub_{s_{j+1}}} = \mathbf{ub_{s_j}} \pm \epsilon_{\mathbf{pat}}(\mathbf{ub_{s_j}} - \mathbf{lb_{s_j}}), \tag{3.19}$$

$$\mathbf{lb_{s_{j+1}}} = \mathbf{lb_{s_j}} \pm \epsilon_{\mathbf{pat}}(\mathbf{ub_{s_j}} - \mathbf{lbs_j}). \tag{3.20}$$

Repeat **Step 2** *to* **Step 5**.

if $f_0(\mathbf{ub_{s_{j+1}}}, \mathbf{lb_{s_{j+1}}}) < f_0(\mathbf{ub_{s_j}}, \mathbf{lb_{s_j}})$

Set current the augmented global objective function as base point:

$$f_0(\mathbf{ub_{s_j}}, \mathbf{lb_{s_j}}) = f_0(\mathbf{ub_{s_{j+j}}}, \mathbf{lb_{s_{j+j}}}), \tag{3.21}$$

$$\mathbf{ub_{s_j}} = \mathbf{ub_{s_{j+1}}} \tag{3.22}$$

$$\mathbf{lb_{s_j}} = \mathbf{lb_{s_{j+1}}} \tag{3.23}$$

else

Reject pattern move.

end

Here $\epsilon_{\mathbf{pat}}$ is the pattern bound movement factor for each variable, which builds over successive successful bound moves. All other symbols represent their already defined values.

3.3 The Database

The Multidisciplinary Pattern Search alone can solve decoupled problems with no interconnecting state variables, such as the conceptual aircraft test case in Section 4.3. Two or more domains can perform their analyses in isolation and solely communicate the objective and constraint statuses of their locally optimal designs directly via the rule base. However, many problems such as the UAV problem that follows in Section 4.2, have state variables or even whole data sets that are output from one domain and serve as inputs to others. Because coupled problems with multiple interconnecting/state variables are much more common in industry, the rule base requires a way to transfer information across organisational domains. The database facilitates this transfer, while also storing bound movements in an encoded list for later use by the duplicate check and Tabu checks.

In the sequential design process, organisational domains perform their analyses as a series of independent optimisations. This ensures that the design being evaluated is consistent with respect to the state variables. Because this can take a long time, a full linear multidisciplinary analysis is generally unwanted in a MDO architecture. In the proposed framework however, domains can work concurrently on their own individual designs. This brings about the question of how to ensure that the proposed designs at the end of a bound movement step are consistent. The simple answer is they don't! The method assumes that state variables have a benign behaviour and are free from severe non-linear variations in the search space. This allows the domains to use the latest available state variables from the database, regardless if it is one that will result in a consistent design. In the early stages of the search where the bounds are wide, it is accepted that the proposed designs from each domain will be inconsistent and globally infeasible. However as the bounds start to reduce to a smaller region, the consistency of each local design should improve as the domains are forced to search an ever reducing design space. Of course there will be risk that this assumption may mislead the Multidisciplinary Pattern Search at the early stages of the design process when the bounds are wide and if the state variables have significant variations. However it is expected that this assumption will generally hold true for industrial aircraft design, because even in the current sequential process, designers often work to engineering (not mathematical) precision and tend to have access to incomplete or uncertain data provided by other domains. One way to address this assumption without a full sequential multidisciplinary analysis is to use surrogate models of other domains' analyses. This certainly offers an avenue for future research, which has been briefly discussed in more detail in Chapter 7.

From an optimisation point of view, how often state variables are communicated across the domains has an impact on the solution. The database stores all coupling information in real time, so in theory one could simply allow each domain to load the latest state variables when they are available in the database. This however can generate internal oscillations, which can be sufficient to render the local domain optimisers altogether ineffective. Ultimately it was found that using the last state variable from the previous completed search in the other domain and keeping it fixed during a local searches (as shown in Figure 3.6) solved these issues.

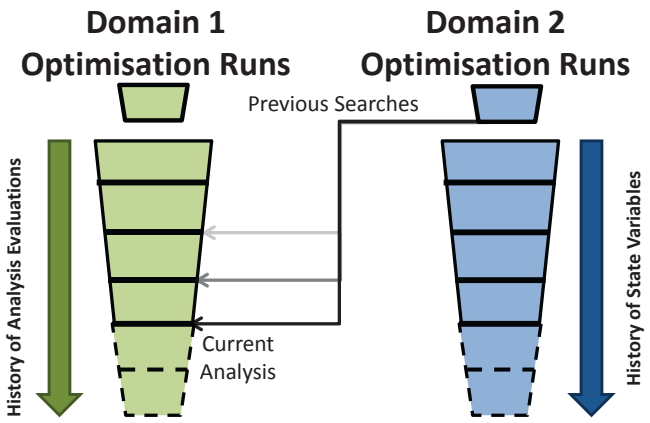

Figure 3.6: Fixed point iteration strategy for state variables

3.4 The System Interface

An interface between designers and the method is one area that is often overlooked by researchers working on MDO architectures. Most automatic MDO search architecture try to replace the chief engineers with a system level optimiser or algorithm as is the case in the proposed blackboard method. Without an adequate human-process interaction, the model can take the chief designer completely out of the loop. It is necessary to allow engineers maintain control over the system level controller and take action if the automatic process starts to converge to infeasible region.

One way for top level designers monitor in real time the process is via a Graphical User Interface (GUI). Figure 3.7 illustrates one way for chief engineers to monitor the process in real time. The figure illustrates many aspects of the convergence history for the simplified version of the UAV wing design problem described in Section 4.2.1.8. The interface uses the database to populate the eleven windows in real-time. The top four graphs illustrate where the locally optimum designs lie in the available search space and next two graphs show the objective and constraint statuses of these designs.

The five smaller graphs at the bottom of the interface show the constraints and the objectives, and update during the domain local searches. So for one bound movement,

the lower graphs update multiple times as they search for an optimal geometry in the available design space. The central display combines feasible designs from both domains on one parallel axis plot. Shown in red are the two optimal designs from each domain, in orange are the next five best designs and in green the remaining pool of feasible designs. Either side, two "Stop" buttons give the user the option to terminate the local domain searches prematurely in the event that the domains are failing to find feasible designs. Overall this can be easily scaled to more complex problems to allow top level designers to monitor and direct the searches of the individual organisational domains.

3.5 Concluding Remarks

Price et al. [120] developed a blackboard MDO framework that can force competing design domains to reach a compromise by iteratively contracting the shared search space. This framework is suitable for industrial MDO problems because it can preserve each domain's existing way of performing design, while using a set of pre-defined rules to guide the design process. Although the concept is sound, the original formulation had not reached a level of maturity to warrant its application to industrial test cases. Its rule set was difficult to comprehend and slow to converge on problems with many shared design [120].

The work in this thesis addresses some of the many issues in the legacy framework. More specifically the work in this chapter describes a newly developed rule base, given the name the Multidisciplinary Pattern Search. It borrows the logic from a number of well-established pattern search optimisers to control the design space for the various domains. Unlike the legacy rule base, the Multidisciplinary Pattern Search uses information learned from previous moves to better predict next move and discard a potential moves that are unlikely to produce feasible designs. This means it shares many of the same benefits as the gradient-free optimisers, which makes it suitable as a search space controller.

The Multidisciplinary Pattern Search however makes one key assumption about the MDO problem. It assumes that state variables are largely insensitive to changes in design. The means the process is unsuitable for highly non-linear problems where large oscillations in the state variables can occur. Given that in the current process, aircraft designers often work with incomplete or inaccurate information, such assumptions should hold true for many industrial applications.

The description of the blackboard search method and the Multidisciplinary Pattern Search contains sufficient information to now proceed with its application to a number of representative test cases.

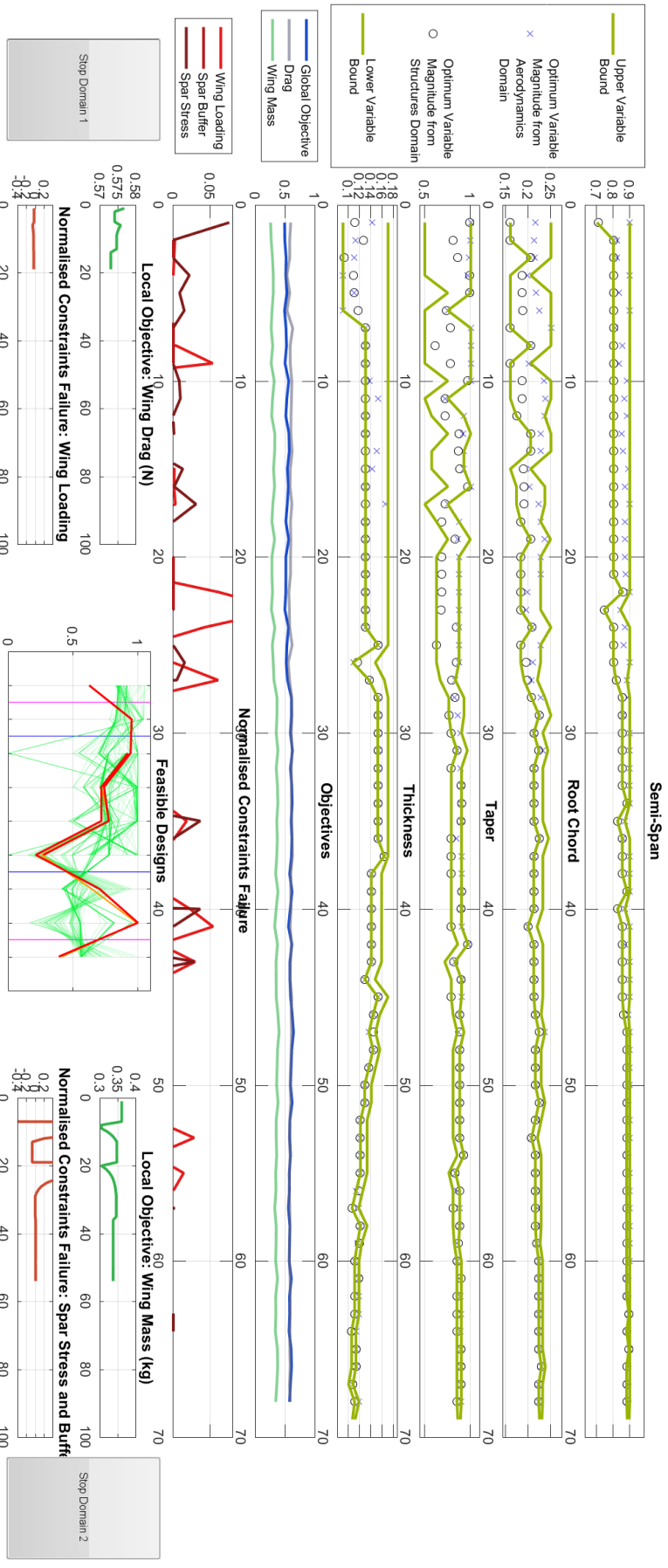

Figure 3.7: Top level user interface

Chapter 4

Application Case Studies

4.1 Introduction

Perhaps the biggest reason why so few distributed architectures have seen application in industrial design engineering is because they have mostly been developed to solve academic test cases that are often unrepresentative of the processes followed in industry. The now obsolete NASA MDO test suit contains many of the most common test cases that researchers have used in past studies [81, 146]. Outside this collection of problems, there exist the analytical two domain problem [64, 115, 127, 133, 146], the geometric programming problem [71, 78, 129, 150] and a supersonic business jet problem [45, 81, 115]. These often re-appear in comparison case studies.

Observations based on practical experience with the analytical two domain and the propane combustion problems, showed that such cases tend to have strong non-linearities between the coupling variables and exhibit landscapes with narrow chasms, which required very high precision to find feasible results. These features are unlike the preliminary aircraft design problem, where state variables relationships tend to be far more benign and the solution landscape is a lot more forgiving. After all, designers regularly make assumptions or work with incomplete or inaccurate data provided by other organisational domains in the sequential design process, and still come up with aircraft that meet the contractual objectives and regulatory constraints.

There is no shortage of alternative, problem specific engineering test cases [8, 9, 23, 27, 28, 21, 72, 76, 81, 120, 127, 134, 140], however, many are outside the scope of intended application and some use black box analyses, which render replication studies difficult. The two chosen test problems used here aim to capture aspects of the commercial aircraft design process, but at a level that could be handled by a single designer. The first uses simple empirical and physics-based formulas and simplified physics to make later replication studies possible. The second test case has more design variables and is similar in concept, but instead uses a calibrated industrial transonic wing design tool.

For the purpose of the tests and unless otherwise stated, the blackboard has been configured with the following default setting:

- the bound control factors are $\epsilon_{c_{init}} = 0.5$, $\epsilon_{mb} = 0.5$, $\epsilon_{cb} = 0.5$ and $\epsilon_{c_{min}} = 0.01$;
- the rule base was allowed to run its course, with the computational steering element omitted for most tests in order to exclude any human element from the tests;
- the local domain searches were performed in parallel;
- most tests were started from a set of independent starting points arranged in a Latin hypercube pattern to test the robustness of the process;
- each new domain local search uses the final value from a previous search as the starting point;
- Matlab^{TM}'s Sequential Quadratic Programming (SQP) optimiser within the function *fmincon* was used for the purpose of local domain optimisations.

4.2 The UAV Wing Design Problem

4.2.1 Introduction

The primary role of the UAV problem was to test the key features of the blackboard search scheme. The problem assumes that an initial sizing of the fuselage, empennage and power plant has been performed to address the mission requirements. Only the wing geometry is optimised in this exercise. The approach adopted here decomposes the wing optimisation process into seven sections that have been put together to reflect stages in the preliminary aircraft design process. Solely for the purpose of illustration, these have been arranged in a sequence as shown in Figure 4.1.

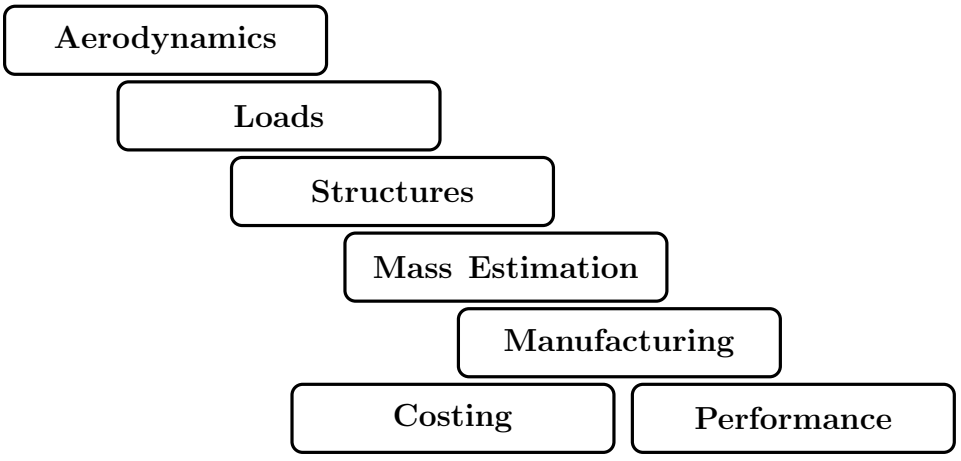

Figure 4.1: The UAV wing problem in a sequential decomposition

The problem assumes that the outer wing shape will be manufactured from insulation foam, while a single tubular main spar provides the stiffness. This class of wing structure has been successfully applied to a number of flying UAV designs in the 1-30kg take off weight class [75]. The ultimate goal is to identify the wing geometry that minimises the global objective and satisfies the constraints. The wing and spar are described by six variables in total, which are: span, root chord, taper, thickness to chord ratio, main spar thickness and main spar diameter. An example candidate geometry is shown in Figure 4.2 for clarity and detailed descriptions of the stages follow in Sections 4.2.1.1 to 4.2.1.7.

This problem has been specifically devised to simulate the preliminary aircraft design process described in Section 2.3. Where possible, different sections make reference on how each aspect of this problem compares with the industrial engineering process.

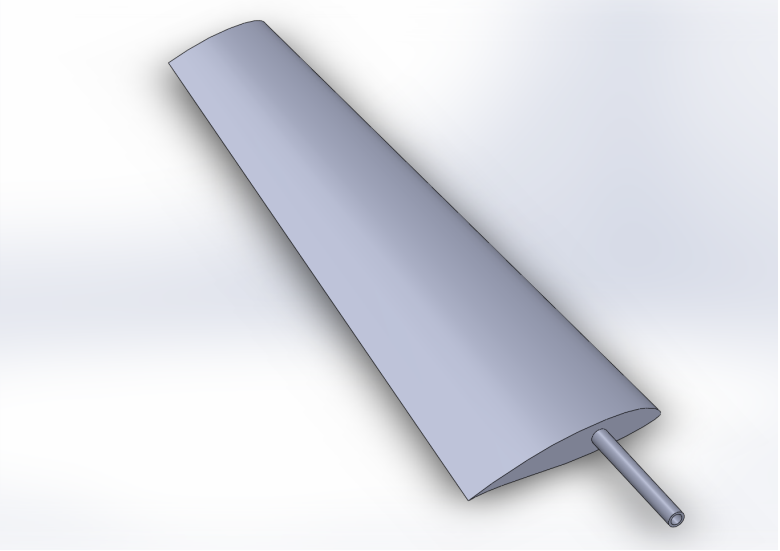

Figure 4.2: An example UAV wing geometry

4.2.1.1 Aerodynamics Domain

The aerodynamics domain has three outputs as illustrated by Figure 4.3: the wing lift L, drag D, and loading $\frac{W}{S}$. Total wing drag can be calculated using:

$$D = \frac{1}{2}\rho V^2 S \left(C_{Di} + C_{Dp}\right) \tag{4.1}$$

with the symbols ρ, V and S representing the air density, cruise speed and reference wing area, and C_{Di} and C_{Dp} denoting the lift induced and profile drag coefficients. These can be estimated from Equations 4.2 and 4.3 respectively:

$$C_{Di} = \frac{C_L^2}{\pi ARe}. \tag{4.2}$$

In Equation 4.2, C_L is the coefficient of lift, AR as the aspect ratio and e the span efficiency. The profile drag coefficient (C_{Dp}) is assumed to equal the product of the flat plate mean skin friction coefficient (C_f), thickness form factor (f_{tc}) and the ratio of the wetted (S_{wet}) and reference (S) wing areas from:

$$C_{Dp} = C_f f_{tc} \frac{S_{wet}}{S}. \tag{4.3}$$

The Blasius relationships for 100% laminar flow [147] is used for the mean skin friction coefficient

$$C_f = \frac{1.328}{\sqrt{Re}}, \tag{4.4}$$

where Re is the Reynolds number based on the mean aerodynamic chord. The form factor is obtained from Torenbeek's [147] empirical formula:

$$f_{tc} = 1 + 2.7 \left(\frac{t}{c}\right)_{max} + 100 \left(\frac{t}{c}\right)^4_{max} \tag{4.5}$$

with $\left(\frac{t}{c}\right)_{max}$ denoting the airfoil maximum thickness to chord ratio. The wetted wing area is estimated using an equation found in Gudmundsson's book [50] as given by

$$S_{wet} = 2 \left(1 + 0.5 \left(\frac{t}{c}\right)_{max}\right) S. \tag{4.6}$$

Finally the lift and the wing loading can simply be calculated using Equations 4.7 and 4.8, respectively:

$$L = m_{tot} g \tag{4.7}$$

and

$$\frac{W}{S} = \frac{m_{tot} g}{S}. \tag{4.8}$$

Here m_{tot} is the total mass of the aircraft, g is the gravitational constant and S is the reference wing area.

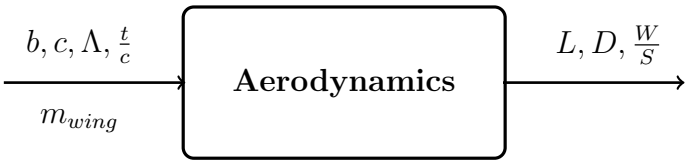

Figure 4.3: Inputs and outputs of aerodynamics analyses

4.2.1.2 Loads Domain

The critical loads in commercial aircraft design are usually calculated and selected by a separate engineering team. A single 7g vertical gust load case has been considered as the critical load case to replicate this to a suitable level of simplicity. The span-wise loads and subsequent bending moments were calculated using a method loosely based on lab notes from the Massachusetts Institute of Technology OpenCourseWare [153]. This method was chosen over other analytical [1, 132] or CFD-based methods as it does not require separate wing inertia estimates. The analysis neglects any torsional forces resulting from the drag or chord-wise lift and assumes a one dimensional distribution along the span. The chosen method assumes that the span-wise loading $q(y)$ is a function of local chord length as it changes along the span. It can be estimated by:

$$q(y) = \frac{nW_{excl.wing}}{b(1+\Lambda)}(1 + (\Lambda - 1)\frac{y}{b}), \tag{4.9}$$

with n, $W_{excl.wing}$ and y denoting the maximum load factor, the aircraft's weight excluding the wing and span-wise location. The span-wise moment can be subsequently obtained by twice integrating the span-wise loads,

$$M(y) = \int_0^b \int_0^b q(y)\, dy\, dy, \tag{4.10}$$

which was achieved here using the trapezium integration method with 50 points. This method also neglects any inertial relief from additional masses that might be located on the wing. Figure 4.4 shows the three inputs and single output of this domain.

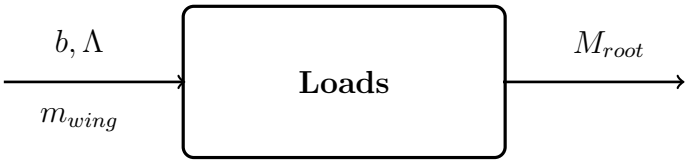

Figure 4.4: Inputs and outputs of loads analyses

4.2.1.3 Structures Domain

Structures teams in commercial aircraft design try to find the lightest or cheapest structure that can withstand the stresses arising from the critical load cases. The problem studied here assumes that the wing stiffness is provided entirely by a tubular main spar running the length of the span. Therefore the sole purpose of the structures domain is to calculate the stress and vertical deflections in the main spar resulting from the maximum bending moment. Therefore the stress in the main spar can simply be calculated using the formula:

$$\sigma = \frac{M_{root} z}{I_y} \tag{4.11}$$

with M_{root} representing the root bending moment, z denoting the vertical distance from the neutral axis and I_y referring to the second moment of area around the neutral axis. The maximum tip deflection, δ, can be calculated using:

$$\delta = \frac{M_{root} b^2}{2EI_y} \tag{4.12}$$

where E is the modulus of elasticity of the chosen material and the remaining symbols maintain their already defined values. Figure 4.5 shows the main inputs and outputs in this domain.

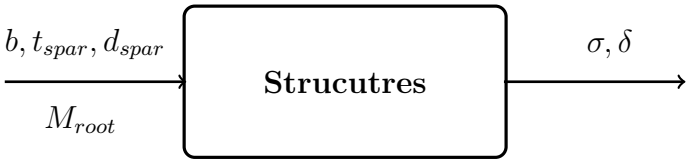

Figure 4.5: Inputs and outputs of structures analyses

4.2.1.4 Mass Estimation Domain

The wing mass is calculated from the sum of the foam section and spar masses. To keep the problem simple, it is assumed that the wing profile can be described using the combined areas of a semi-ellipse and trapezium as shown in Figure 4.6. This way the wing mass can be calculated without defining a specific airfoil. The step-by-step guide for the mass calculation follows next. First, the profile area can be estimated using Equation 4.13,

$$A_{root} = p_{cut}\left[\frac{1}{4}\pi c^2 t_{\%}\left(\frac{t}{c}\right)_{max} + \frac{1}{2}c^2(1-t_{\%})\left(\frac{t}{c}\right)_{max}\left(1+0.03+\frac{t_{\%}}{10}\right)\right] \tag{4.13}$$

where $t_{\%}$ is a fixed the chord-wise location of maximum thickness and p_{cut} is a the cut-out fraction calculated from Equation 4.14.

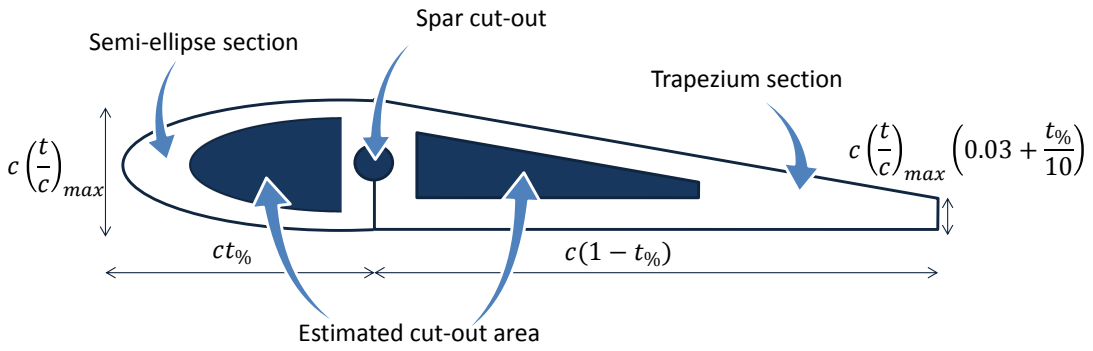

Figure 4.6: Cross-section of the wing profile and internal structure

$$p_{cut} = 1 - \left(\frac{t}{c}\right)_{max}. \tag{4.14}$$

The mass of the foam section can be calculated using the equation for the volume of a frustum:

$$m_{foam} = \frac{2}{3} b(A_{root} + \sqrt{A_{root} A_{tip}} + A_{tip})\rho_{foam}. \tag{4.15}$$

Here A_{root} is the profile area at the root, ρ_{foam} is the density of the foam and A_{tip} is the profile area at the tip obtained by:

$$A_{tip} = \Lambda^2 A_{root}. \tag{4.16}$$

The mass of the spar is simply:

$$m_{spar} = 2A_{spar} b \rho_{spar} \tag{4.17}$$

where A_{spar} is the cross-sectional area of the spar and ρ_{spar} is density of the spar material. Therefore the total mass is then:

$$m_{wing} = m_{foam} + m_{spar} + m_{aux} \tag{4.18}$$

where m_{aux} denotes any auxiliary masses on the wing (i.e. servos, wiring, sensors, ets...). One noteworthy feature of this domain is that it lacks state variables, which is shown in Figure 4.7.

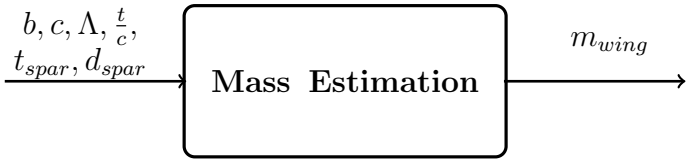

Figure 4.7: Inputs and outputs of mass analyses

4.2.1.5 Manufacturing Domain

As shown in Figure 4.2, the main spar has to fit flush inside the foam geometry, with sufficient buffer between the edges of the spar and upper and lower surfaces of the wing. The model assumes that the spar is positioned at the maximum thickness to chord location chord-wise and calculates the spacing at the tip, where it is usually smallest because of any taper that may be present in the wing. The manufacturing domain calculates the spacing between the spar and the wing surfaces using Equation 4.19:

$$t_{buff} = \left(\frac{t}{c}\right) c\Lambda - t_{spar}, \tag{4.19}$$

where $\frac{t}{c}$ is the maximum thickness to chord ration, c is the chord length at the root and Λ is the taper ratio. The input/output diagram for the manufacturing domain is shown in Figure 4.8. This domain also does not require any state variables in its analysis.

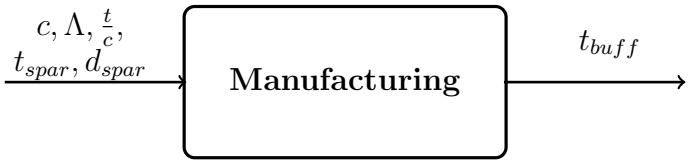

Figure 4.8: Inputs and outputs of manufacturing analyses

4.2.1.6 Costing Domain

This domain estimates the material costs needed for the wing. Table 4.1 provides the information to estimate the cost of the main spar. The foam material used for the body of the wing comes in standard sections with dimensions 0.6m by 2.5m by 0.1m. The method estimates how many pairs of wings can fit in one standard sheet of foam. Each pair of wings wing then absorbs a fraction the total cost of a foam block. The first step estimates how many semi-span (b) and chord (c) sections can fit in the width and length of the sheet. Equation 4.20 and 4.21 represent this by:

$$No_X = \lceil \frac{0.6}{c} \rceil \tag{4.20}$$

and

$$No_Y = \lceil \frac{2.5}{b} \rceil. \tag{4.21}$$

Here No_X and No_Y represent the number of wings that can fit in the width and height dimensions of the block, whereas c and b represent the wing chord and semi-span. The model adopts a logic statement for the thickness of the block. If the combined thickness of the wing tip and root profiles exceeds a minimum value, only one wing half is considered to fit inside the thickness of the foam block. Otherwise an estimate for the number of full wings can be calculated.

This logic can be described in pseudo-code by:

if $t(1 + \Lambda) + 0.02 < 0.1$:
  $No_Z = \lceil \frac{0.1}{t(1+\Lambda)} \rceil$
else
  $No_Z = 0.5$
end

The symbol No_Z denotes the number of wings that can fit in the height dimension of the foam block and, t and Λ signify the absolute root profile thickness in m and wind taper respectively. Equation 4.22 estimates the total number of wings that can fit side a sheet of foam by

$$No_{wings} = No_X No_Y No_Z. \tag{4.22}$$

From Equation 4.22, the cost of foam per wing pair can be calculated as follows:

$$C_{foam} = \frac{C_{block}}{No_{wings}}. \tag{4.23}$$

The combined cost of the foam and the spar make up the cost of the wing using:

$$C = C_{foam} + C_{spar}. \tag{4.24}$$

Figure 4.9 shows the inputs and outputs in this domain and highlights that this domain also does not have any state variables.

Table 4.1: Costing data for a carbon fibre main spar. All cost entries exclude VAT and reflect the cost of 1m long, off the shelf spar from Easy Composites Ltd. in September 2017.

Type of Spar	Thickness, t_{spar}	Diameter, d_{spar}	Cost, C
Pultruded Rod	0.0015	0.0030	3.300
	0.0020	0.0040	4.850
	0.0025	0.0050	8.150
	0.0030	0.0060	9.600
	0.0040	0.0080	13.10
	0.0050	0.0100	17.55
	0.0060	0.0120	21.15
Pultruded Tube	0.0005	0.0040	2.900
	0.0010	0.0050	5.950
	0.0010	0.0060	6.850
	0.0010	0.0070	7.200
	0.0010	0.0080	8.600
	0.0010	0.0120	11.55
Roll Wrapped Tube	0.0010	0.0100	12.61
	0.0014	0.0127	12.84
	0.0014	0.0155	13.97
	0.0014	0.0167	14.62
	0.0014	0.0208	17.37

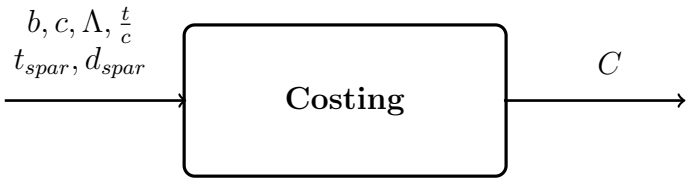

Figure 4.9: Inputs and outputs of cost analyses

4.2.1.7 Performance Domain

The final domain calculates the still air range of the aircraft assuming certain properties about the fuselage drag and the propulsions system on the UAV. The equation used in this domain is given by:

$$R = \frac{1}{1000} E^* \eta_{total} \frac{1}{g} \frac{L}{(D + D_{fuse})} \frac{m_{batt}}{(m_{wing} + m_{fuse})} \tag{4.25}$$

where E^* is the mass specific energy content of the battery in Wh/kg, η_{total} is the total efficiency of the power plant, g is the gravitational constant, L is the lift produced by the wing, D and D_{fuse} are the wing and fuselage drag contributions, m_{batt} is the mass of the battery and, m_{wing} and m_{fuse} are the mass of the wing and fuselage. It is noteworthy that all the input variables for this domain are output variables from other domains as shown in Figure 4.10.

Figure 4.10: Inputs and outputs of performance analyses

4.2.1.8 Two Versions

Sections 4.2.1.1 to 4.2.1.7 provide the building blocks for the UAV wing design problem. How to best organise these inside the blackboard model is an interesting research question, but one that is outside of the scope of this thesis. Here two slightly different decomposition strategies are proposed for two distinct versions of this problem.

The global objective function (f_0) in the first version comprises of range R (in km) and cost C (in £) objectives. This is because in most real world design problems, engineers often have to trade performance with cost. Equation 4.26 gives the global objective mathematically as:

$$f_0 = C - R. \tag{4.26}$$

No weighting factors were used in f_0, because the magnitudes of range and cost were comparable. In addition, there are five constraints in this problem. The stall speed is generally the leading constraint in industrial aerodynamic design. But because of the absence of simple empirical or physics-based methods for estimating stall speeds accurately, the maximum wing loading ($\frac{W}{S}$) limit was used instead. This is given by Equation 4.27 as:

$$\frac{W}{S} \leq 40. \tag{4.27}$$

The following two constraints apply to the aspects of the design. As discussed earlier, the internal wing structure has to withstand numerous critical loads and satisfy minimal deflection and twisting movements to avoid aerodynamic flutter and divergence phenomenon. A maximum stress σ and deflection δ constraints aim to replicate these real world design limits and thus are given by Equations 4.28 and 4.29 by:

$$\sigma \geq FOS\sigma_{UTS} \tag{4.28}$$

and

$$\delta \leq 0.05. \tag{4.29}$$

The symbol in Equation 4.28 FOS refers to the selected factor of safety.

The final two constraints aim to capture some of the many manufacturing limitations that exist in the industrial design engineering process. Equation 4.30 ensures that there is significant buffer (t_{buff}) between the main spar and the upper and lower skin surfaces of the wing and Equation 4.31 ensures that the spar thickness (t_{spar}) does not exceed half the value of the diameter (d_{spar}):

$$t_{buff} \geq 0.01 \tag{4.30}$$

and

$$t_{spar} \leq \frac{d_{spar}}{2}. \tag{4.31}$$

Figures 4.11 and 4.12 illustrate how the different domains are organised within the blackboard. It is noteworthy that the structures, manufacturing and mass estimation domains are grouped under one optimiser. This effectively means these three domains evaluate the same design, which is generated by this optimiser.

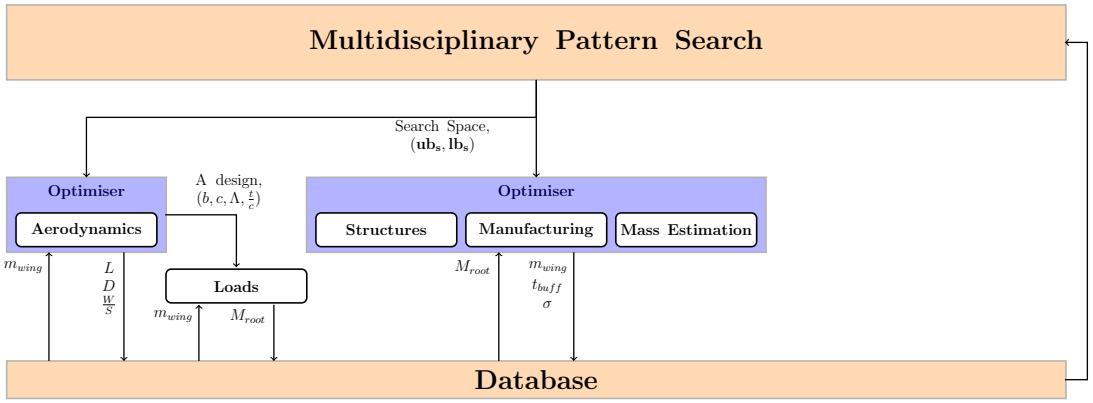

Figure 4.11: Domain arrangement in the simple UAV problem version

Now because multiple repeat runs of the blackboard were needed to tune and analyse the behaviour of the Multidisciplinary Pattern Search, it would have been prohibitively time consuming and therefore impractical to use this version of the problem for all tests. So a simplified variant that omitted the range and costing objectives, along with the deflection constraint, was used in many of the repeated tests that follow. Overall this lowers the computational budget, while still retaining a similar level of similarity to the industrial design process. The absence of the costing domain removes any discontinuities, which also allows the problem to lend itself better for the Simultaneous Analysis and Design architecture using a gradient-based optimiser. The simplified version has also been used in the team-based case study described in Chapter 6, while the more complex version has been on the surrogate assisted work described in Chapter 5.

A weighted sum of mass (m in kg) and the drag (D in N) was used for the simplified version of the problem in lieu of the combined cost and range objective function. It is given by Equation 4.32, where the weightings $\alpha_1 = 0.4$ and $\alpha_2 = 1$ were selected arbitrarily:

$$f_0 = \alpha_1 D + \alpha_2 m_{wing}. \tag{4.32}$$

Figure 4.11 illustrates the blackboard decompositions for the simplified variant. The reader will also notice that the analysis loads domain directly uses the optimal geometry from aerodynamics domain in the simplified version. In most sequential aircraft design processes, the loads analysis follows the aerodynamic analyses and thus this formulation tries to adhere to that model in the simpler version of the problem. In the more complex version of the problem however, the loads analyses comes after the structural optimisation domain. The rational behind this is that bending moment calculations are solely needed for the structural optimisation. Given that the aerodynamics and structural optimisation domains usually develop conflicting geometries in the blackboard format, it is more intuitive for the loads domain to analyse the final structural geometry. This difference is illustrated in Figures 4.11 and 4.12 by the input arrow to the loads domain.

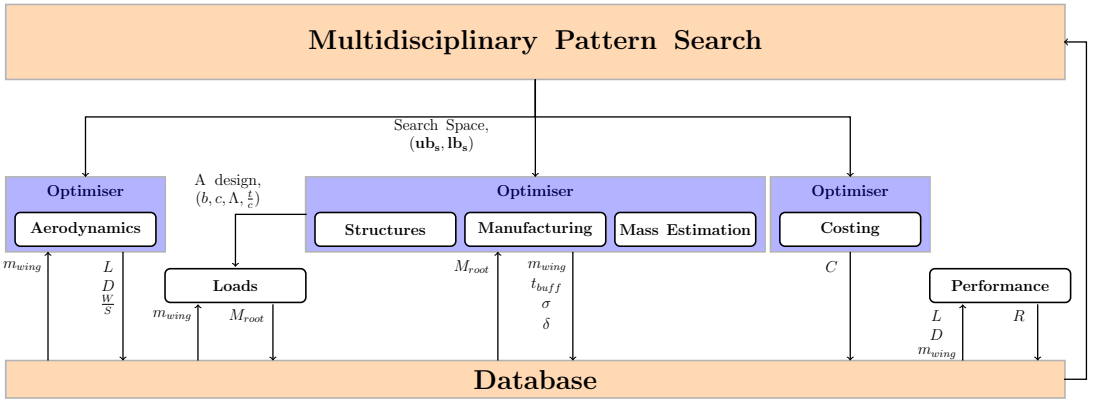

Figure 4.12: Domain arrangement in the complex UAV problem version

In addition, an exhaustive search over the simplified problem has been performed and using quadratic regression, a hierarchical axis plot has been put together to help visualise the available design space. Each small contour plot represents an exhaustive search across of wing taper (Λ) and thickness to chord ratio ($\frac{t}{c}$), while the semi-span (b) and root chord ratio (c) have been kept constant. Highlighted in colour is the range of global optimality of the feasible design region. Note that most of the design space is deliberately left infeasible to help visualise the blackboard convergence using Figure 4.15. Finally, Table 4.2 summarises all the variables, including a number of fixed quantities not mentioned above.

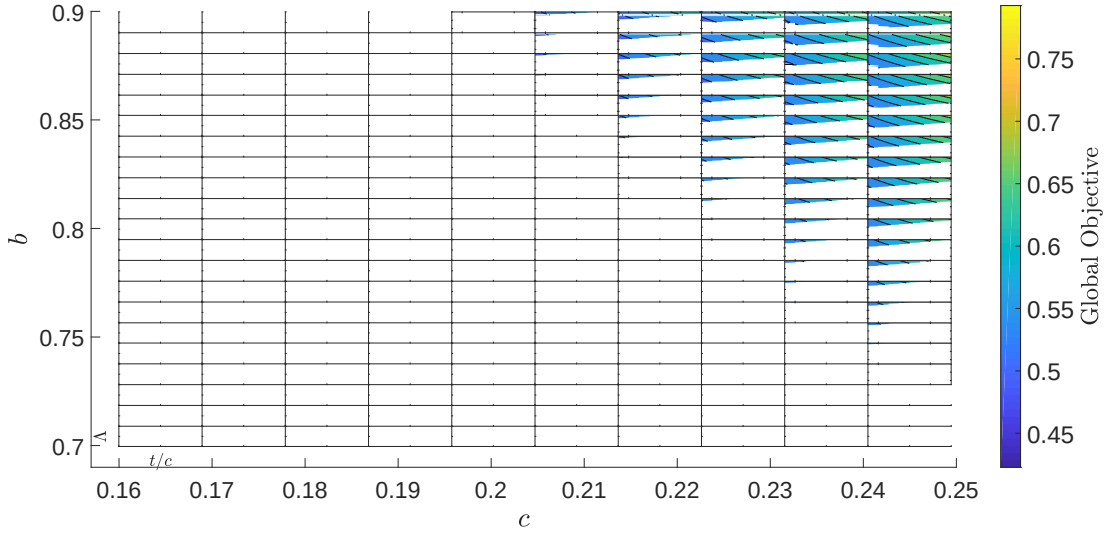

Figure 4.13: Hierarchical surface plot in four dimensions showing the global objective and constraints

4.2.1.9 Implementation

4.2.2 Both versions of the problem have implemented in Matlab. The local drag, mass, range and cost objectives are optimised using the Sequential Quadratic Programming optimiser built in the Matlab function *fmincon*. Each local constraint was scaled using the fixed limits as given by Equations 3.1 and 3.2 earlier in Section 3.2.2. The internal searches were terminated when the scaled failures reached a value below 0.001 or exhausted the maximum number of analysis evaluations, which were at 20 for the aerodynamics and 26 for the structures domains. In addition, domain optimisations were performed concurrently using the parallel computing toolbox in Matlab for the purpose of all tests. No user input was applied through the user interface throughout the entire search in order to obtain objectionable and comparative results.

4.2.2 Contribution of Each Rule

In an effort to improve the convergence rate of the algorithm, the Multidisciplinary Pattern Search has been augmented by the use of several additional rules beyond the simple neighbourhood search logic. This section investigates how each additional rule in Multidisciplinary Pattern Search affected the search process.

Starting from the basic multidisciplinary neighbourhood search (NS) logic, each individual aspect of the algorithm was enabled one after the other in the order of: the pattern move (PM), followed by the duplicate check (DC) and finally the Tabu rule (TR). Table 4.3 shows the mean objectives, constraint failures and analysis calls from 150 independent runs for these tests.

Table 4.2: UAV wing design objectives, variables and constraints. All atmospheric properties are fixed at International Standard Atmosphere (ISA) sea level conditions.

Objectives	Lower limit	Symbol	Upper limit	Units	Fixed value
Minimise:					
Global objective	-	f_0	-	-	-
Wing drag	-	D	-	N	-
Wing mass	-	m_{wing}	-	kg	-
Shared optimisation design variables					
Semi-span	0.7	b	0.9	m	-
Root chord	0.16	c	0.25	m	-
Taper	0.5	Λ	1.0	-	-
Maximum thickness to chord ratio	0.09	$\frac{t}{c}$	0.17	-	0.1200
Spar					
Wall thickness	0.0010	t_{spar}	0.0080	m	-
Outer diameter	0.0020	d_{spar}	0.0160	m	-
Fixed variables					
Chord-wise location of maximum profile thickness	-	$t_{\%}$	-	-	0.4000
Cruise speed	-	V	-	m/s	13
Oswald efficiency	-	e	-	-	0.84
Fuselage and empennage mass	-	m_{fuse}	-	kg	1.150
Auxiliary mass on wing	-	m_{aux}	-	kg	0.150
Factor of safety	-	FOS	-	-	1.4
Fuselage and empennage drag	-	D_{fuse}	-	-	0.55
Spar density	-	ρ_{spar}	-	kg/m^3	1550
Foam density	-	ρ_{foam}	-	kg/m^3	30
Maximum allowable stress	-	σ_{UTS}	-	MPA	590
Modulus of elasticity	-	E	-	GPA	228
Load factor	-	n	-	-	7
Battery mass	-	m_{batt}	-	kg	0.188
Battery energy content	-	E^*	-	Wh/kg	432000
Power plant efficiency	-	η_{total}	-	-	0.3
Cost of foam block	-	C_{block}	-		25.7
Gravitational acceleration	-	g	-	m/s^2	9.81
Domain local constraints					
Spar buffer	0.01	t_{buff}	-	m	-
Wing loading	-	$\frac{W}{S}$	40	N/m^2	-
Tensile stress	-	σ	421.4	MPa	-
Vertical tip deflection	-	δ	0.05	m	-

Table 4.3 shows a general decrease in the number of analysis evaluations with increased complexity of the rules. With the exception of the pattern move, all other additional rules reduced the number of analysis evaluations without significantly affecting the final outcome. The increase in function count for the pattern move evaluations can be attributed to two factors. Firstly, both the duplicate check and Tabu moves prevent certain rules from firing, therefore saving on what would otherwise be wasted analysis evaluations. On the other hand the pattern move adds an additional move at the end

Table 4.3: Effect of additional rules on final outcome

	MDPS (NS)*	MDPS (NS+PM)*	MDPS (NS+PM+DC)*	MDPS (NS+PM+DC+TM)*	MDPS (NS+DC+TM)*
Mean objectives:					
Global objective	**0.5204**	**0.5226**	**0.5231**	**0.5249**	**0.5208**
Drag	0.5363	0.5404	0.5419	0.5445	0.5368
Mass	0.3059	0.3064	0.3064	0.3071	0.3061
Mean scaled constraints failure					
	0.49%	0.23%	0.40%	0.21%	0.46%
Mean Iteration count					
Moves	132	135	116	113	127
Drag minimisation	1780	1849	1612	1600	1761
Mass minimisation	2468	2587	2228	2152	2361

*The abbreviations are denoted as follows: Multidisciplinary Pattern Search (MDPS), Neighbourhood Search (NS), Pattern Move (PM), Duplicate Check (DC) and Tabu Rule (TM).

of a neighbourhood search, which in some cases may be discarded, thus increasing the number of analysis evaluations.

Secondly, the blackboard uses a fixed point iteration scheme for the state variables between local domain searches. Because the current preference value depends on the outcome of the previous search (see description on state variable strategy in Section 3.3), there is an element of randomness in the process, which can perturb the search to follow a longer path to the minimum and increase the number of analysis evaluations. When the search was run without the pattern move (but including the duplicate and Tabu rules as shown in the final column of Table 4.3) the convergence rate increased. Furthermore when local searches were started from a random point, rather than the previous optimum value, the addition of the pattern move alone was observed to reduce the total number of analysis evaluations by 13%.

Ultimately, the rule convergence for one optimisation run can be visualised using Figures 4.15 and 4.14. These plots are arguably the most important in thesis as they visualise how the Multidisciplinary Pattern Search converges. Figure 4.14 is similar to the user interface presented in Figure 3.7, with the main difference being that all design evaluations are shown along the x-axes. The top four graphs in each column show how the bounds on each converging over successive iterations. The third line that falls between the bounds represents the value each variable takes at a given iteration. The bottom two graphs in each column represent the objectives and show the normalised constraint failures in each domain. Figure 4.14 overlays the changes in search space on the hierarchical axis plot, whereby the darkest region show the area where the bounds have converged.

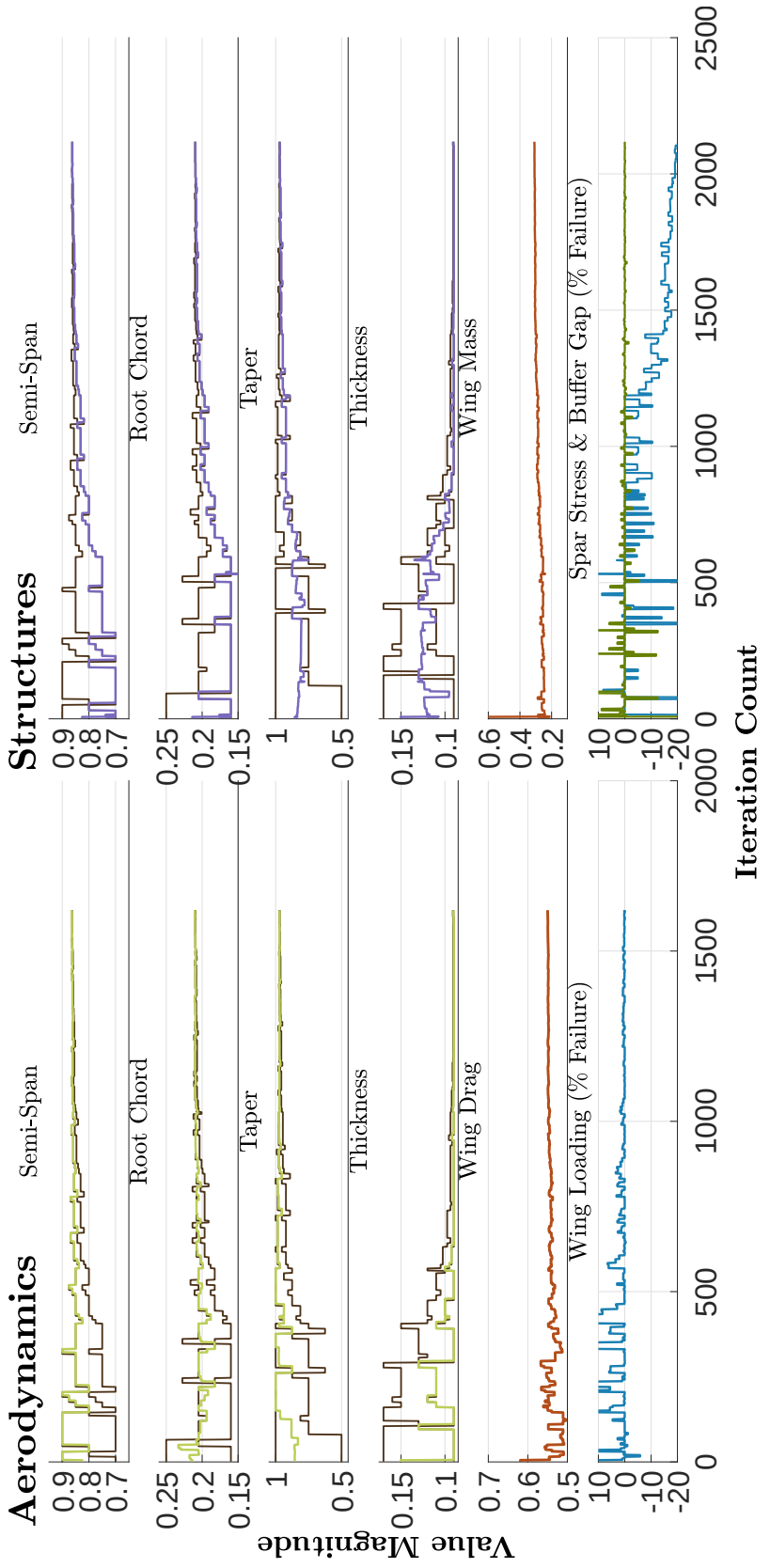

Figure 4.14: Bound movements superimposed on linear axis plot

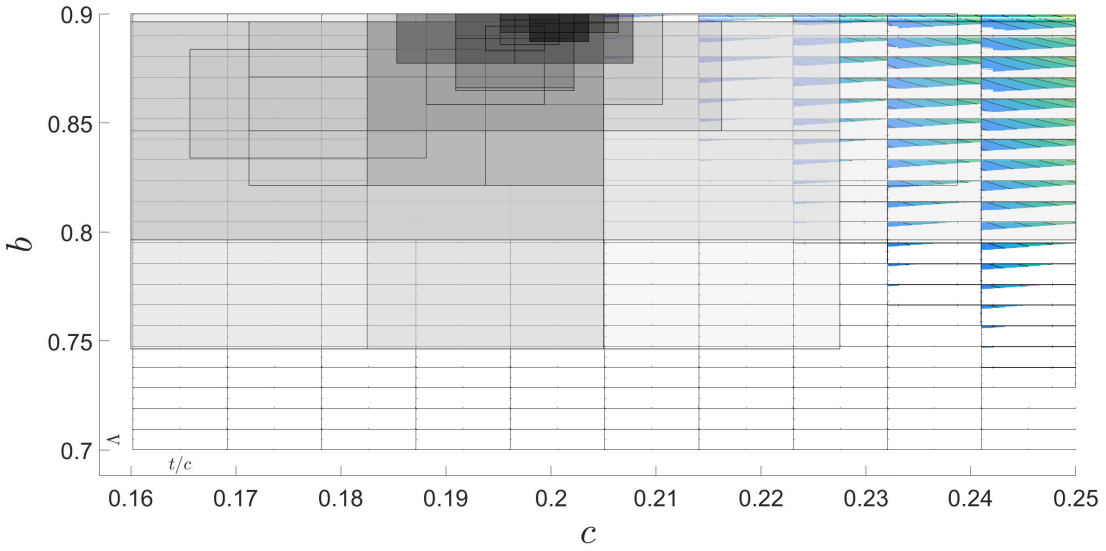

Figure 4.15: Bound movements superimposed on hierarchical surface plot

4.2.3 Bound Control Parameter

This section examines how changes in bound movement and contraction factors affected the performance of the rule base. The problem was started from the fixed starting point given in Table 4.2 and solved for an exhaustive range of bound movement and contraction factors, while keeping all other parameters fixed at their earlier defined default values.

Figures 4.16 and 4.17 show how these impact the final objective function and number of analysis evaluations respectively. The darker (blue and green) tones indicate a low objective value and a low number of analysis evaluations. It can be seen that the magnitude of the bound movement factor (as displayed by ϵ_{mb} on the y-axes) has very little impact on both the number of analysis evaluations and the objective as a whole. On the other hand the choice of bound contraction factor has a more noticeable impact, as can be seen by colour changes along the x-axes in both figures. As one would expect, the algorithm converges much faster with a higher bound contraction factor in general. With the higher bound contraction factor however, the algorithm is also more likely to converge to a local minimum or a design with a constraint failure, hence one can see a general worsening in the final objective value. Ultimately the outcomes of these tests motivated the selection of the previously detailed default bound control factors.

It is important to mention that these contour plots are made from 156 analysis evaluations for various combinations of factors. Regions between each data point are interpolated by Matlab's contour plotting function to generate a smooth contoured surface. There is significant noise in the figures, which can be attributed to a number of factors. The main reason is that there is no guarantee that the two adjacent combinations of factors will converge to a similar result. This is because different combinations of factors could in theory take a different (or longer) path to the global minimum or even

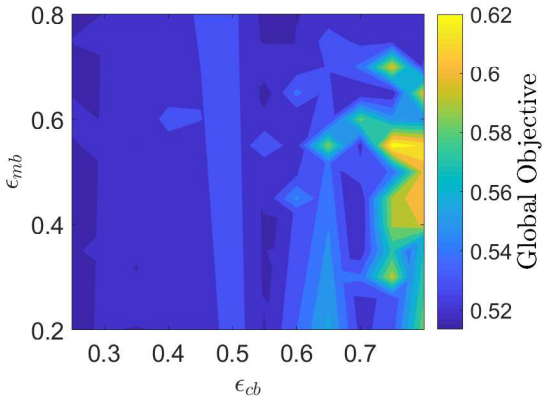

Figure 4.16: Effect of bound movement and contraction magnitude on the final design for simplified UAV version

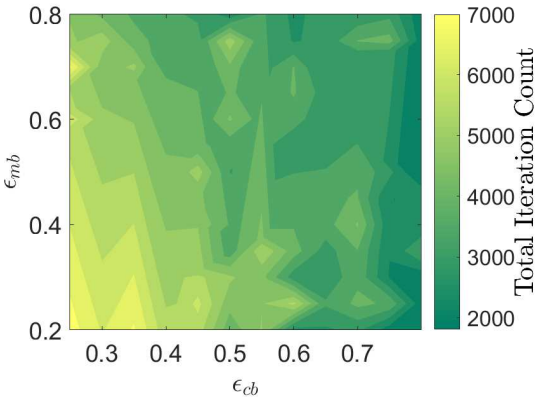

Figure 4.17: Effect of bound movement and contraction magnitude on the number of analysis evaluations for simplified UAV version

terminate to a local minimum. Secondly, the sharp steps and spikes are the result of the linear interpolation algorithm linking relatively sparsely populated points. With a higher density of points, one should see less severe discontinuities in the plots.

4.2.3.1 Starting Bound Separation

The Multidisciplinary Pattern Search has two main tasks: firstly to find the optimal design region and secondly reduce the searchable design space. So far, the Multidisciplinary Pattern Search has been run from the absolute until the desired bound spacing. It was interesting to determine if there was any benefit to starting the process from a smaller bound spacing (still within the original bounds) and let the rule base focus more on the searching task. Five separate tests investigated the ability of the algorithm to find the global minimum from different starting bound spacing. All tests were started from the same 150 starting points, which were arranged in a Latin Hypercube and their initial bounds were centred around the starting point.

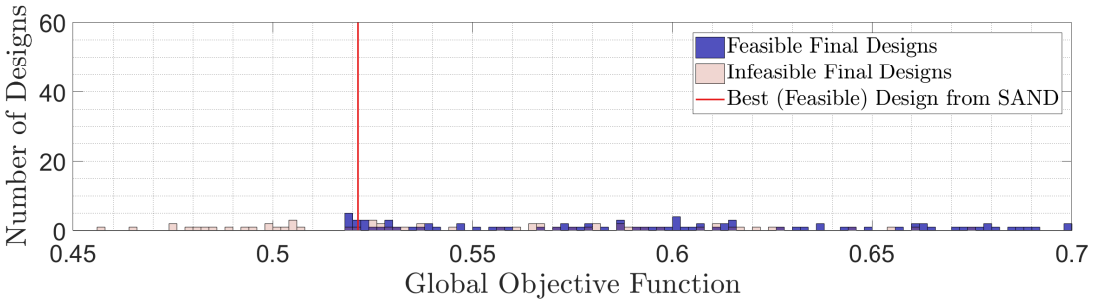

Figure 4.18: Starting bound separation 5%

Figures 4.18 to 4.22 show that starting the process from tighter bounds was more likely drive the final result to a local minimum. This is visible in the difference in scatter

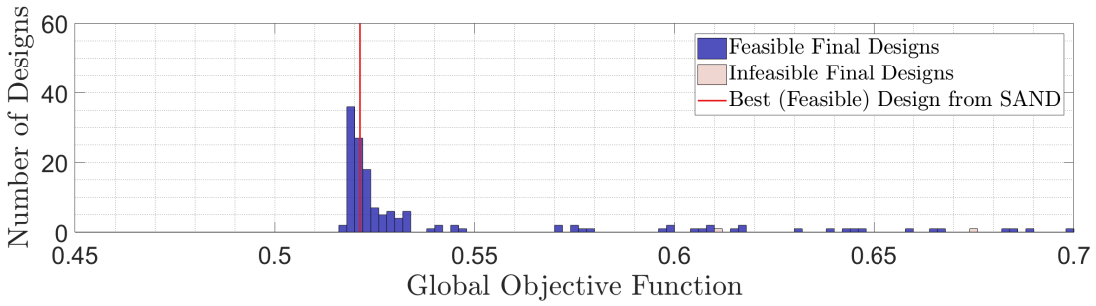

Figure 4.19: Starting bound separation 25%

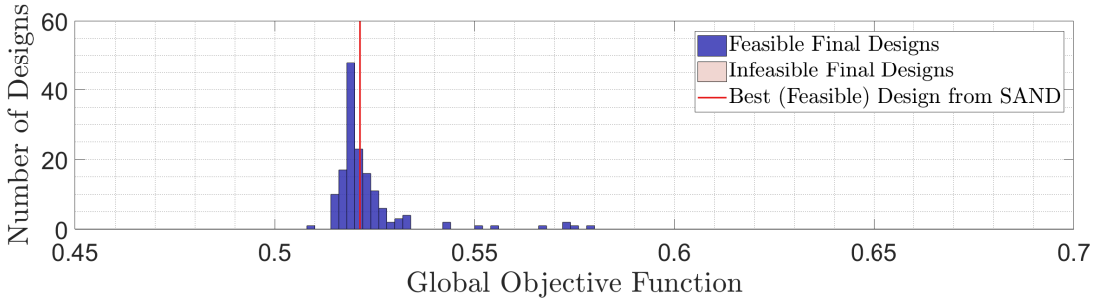

Figure 4.20: Starting bound separation 50%

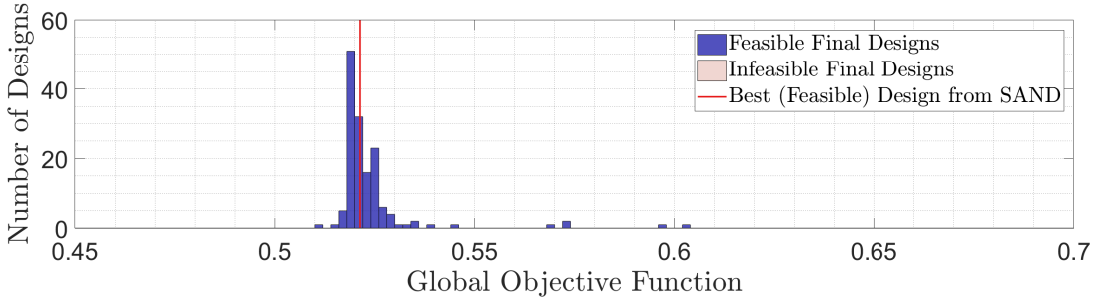

Figure 4.21: Starting bound separation 75%

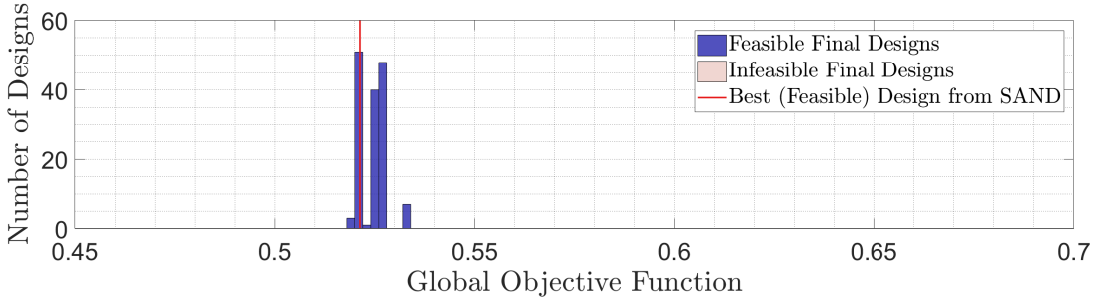

Figure 4.22: Starting bound separation 100%

between the five figures. Although that the process converged slightly faster when started from narrower bounds, there seems to be little benefit in starting the search from the narrower bound spacing.

The results presented in Figures 4.18 to 4.22 also show that the Multidisciplinary Pattern Search sometimes finds a slightly better result than the Simultaneous Analysis and

Design architecture. This is because the blackboard method generated results with a slightly higher constraints failure, although still below our scaled 1% threshold that has been used in Figures 4.18 to 4.22.

4.2.4 Comparison with Existing MDO architectures

The final tests serve as a comparison between the proposed blackboard approach and the two classical MDO architectures. The original formulation of Collaborative Optimisation is designed to minimise a single objective, which means that the problem structure has to be changed slightly. The most suitable problem decomposition requires all domain objectives to be computed at the system level, with only certain selected analyses performed at domain level. If the internal analyses modules were modified to accommodate this decomposition, the UAV problem would fail to emulate the assumed industrial engineering process, which it has been specifically designed to do. So instead, the tests were run with the same analysis modules used in the Multidisciplinary Pattern Search, but using the Multiobjective Collaborative Optimisation (MOCO) decomposition [145], which better suits this type of problem. The Simultaneous Analysis and Design on the other hand has a set problem decomposition, which was implemented without modification.

Table 4.4: Comparison between three MDO architectures

Performance metrics and design variables	**Mean results**			**Lowest objective design**		
	SAND*	MOCO*	MDPS*	SAND*	MOCO*	MDPS*
Objectives						
Global objective	**0.5225**	**0.5559**	**0.5249**	**0.5213**	**0.5224**	**0.5200**
Drag	0.5360	0.5709	0.5445	0.5355	0.5372	0.5355
Mass	0.3081	0.3276	0.3071	0.3071	0.3076	0.3058
Scaled constraints failure						
	0%	0%	0.21%	0%	0%	0.0018%
Iteration count						
Drag analysis runs	80	123811	1600	59	120659	1363
Mass analysis runs	80	122135	2152	59	90170	1786
Design variables						
Semi-span				0.9000	0.8962	0.8982
Root chord				0.2152	0.2171	0.1986
Taper				0.8445	0.8378	0.9948
Maximum thickness to chord ratio				0.0900	0.0902	0.0900
Wall thickness				0.0010	0.0010	0.0010
Outer diameter				0.0064	0.0063	0.0064

*SAND and MOCO are abbreviations for Multiobjective Collaborative Optimisation and Simultaneous Analysis and Design. MDPS stands for Multidisciplinary Pattern search used with the blackboard method.

Table 4.4 shows that the Multidisciplinary Pattern Search is significantly faster to converge than Multiobjective Collaborative Optimisation and achieves results comparable with the Simultaneous Analysis and Design, but in a fully distributed and concurrent fashion. Although every effort has been made to ensure a direct and fair comparison between the architectures, the results presented in Table 4.4 should be viewed with a degree of caution. Both comparison architectures have some (though few) internally tuneable parameters, which makes them susceptible to unintentional biasing [64]. This can of course mislead some the conclusions, hence why the results presented in Table 4.4 are viewed as a broad comparison rather than a critique of any given architecture.

4.3 Transonic Wing Design Problem

4.3.1 Introduction

The second test case uses a decommissioned, cut down version of an industrial engineering tool for transonic aircraft design [31, 74]. The tool is a single entity that calculates numerous aerodynamic, structures and costing parameters using analytical and empirical functions. It has been previously used in sensitivity studies investigating how certain geometric parameters affect the performance of the aircraft. There are around 70 variables in total that define various aspects from the wing, fuselage, empennage and power-plant. To keep the problem manageable here however, most of these have been kept fixed to describe a regional aircraft in 220 seat category, with a fixed wing area of $130m^2$, maximum take off mass of 97 tonnes and operating at Mach 0.785. Only 10 wing variables were varied to find a compromise between the wing weight and aerodynamic drag. As the tool is single a black box evaluator, it was artificially decomposed into two domains for the purpose of this problem.

There are two key sections embedded in the tool for the wing analysis: aerodynamics and structures. The aerodynamics aspect of the problem uses an analytical method, which has been calibrated using commercial aerofoil sections to estimate transonic wing drag and the pitch-up parameter. The aerodynamic drag is the objective in this domains, while a pitch-up margin acts a constraint in the problem. Pitch-up is a form of high speed stall that can occur in swept wings flying in the transonic regime and well below the maximum coefficient of lift [131]. The total wing drag D/q is given in units of Newtons per unit dynamic pressure in Pascals, whereas the pitch-up value has no units.

The structures domain calculates the mass of the wing by splitting the structures into two categories. The primary load bearing structure is sized using Torenbeek's [147] stress-based method that was modified by Cousin and Metcalfe [31]. The aerodynamics loads that generate the torsional and bending stresses are calculated separately using the ESDU 83040 [1] method. A separate calibrated empirical method is used to size

the secondary wing components (such as flaps, ailerons, spoilers, slats etc..) to remain consistent with the load bearing elements. A minimum wing volume and undercarriage bay length form the two constraints in this domain.

Two separate versions of the tool were grouped together with Matlab's SQP optimiser and incorporated in the blackboard model. Figure 4.23 illustrates this decomposition. In some respects this problem is somewhat simpler than the previous one, because the domains are standalone and have no interconnecting state variables. The difficulty however comes from the higher number of design variables, as well as the internal domain non-linearities, which for this problem are largely unknown.

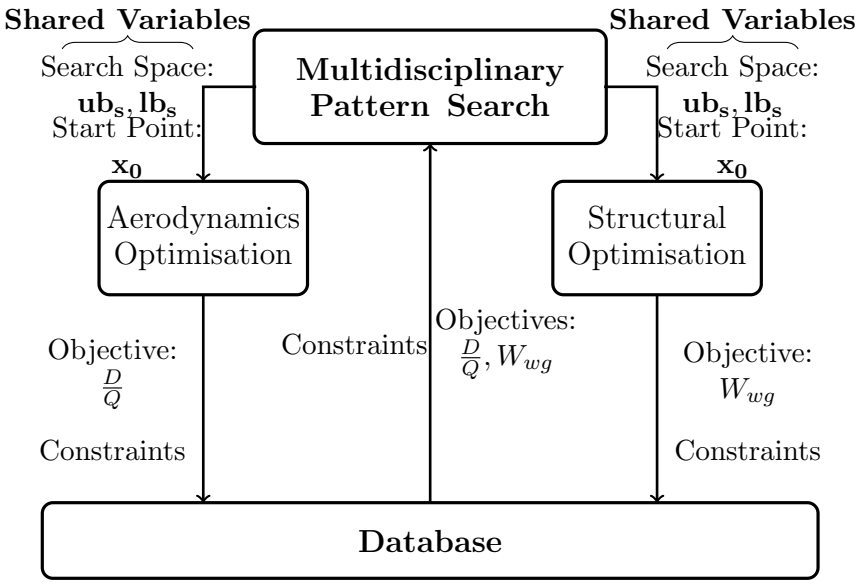

Figure 4.23: Blackboard data transfer for the transonic wing design problem

For the purpose of the MDO problem, a combined objective function

$$f_0 = 0.15D/q + 0.00000405W_{wg}, \tag{4.33}$$

and several constraint functions

$$\tilde{p} \leq 5.4, \tag{4.34}$$

$$\tilde{l_U} \geq 2.1, \tag{4.35}$$

and

$$\tilde{V} \geq 23 \tag{4.36}$$

were devised to collectively minimise the wing drag and mass at the system level. Here D/q and W_{wg} represent the ratio of wing drag to dynamic pressure and wing weight, while the two weighting factors for the objectives were chosen arbitrarily. The symbols $\tilde{p}$, $\tilde{l_U}$ and $\tilde{V}$ describe the pitch-up, undercarriage bay length and wing fuel volume constraints. The design variables, objectives and constraints are all summarised in Table 4.5.The problem was solved using the blackboard as well as the two classical MDO architectures for the purpose of comparison and the results are discussed in Section 4.6.

Table 4.5: Design variables, objectives and constraints in the transonic aircraft design problem

Performance metrics and design variables	Upper limit	Lower limit	Units
Find variables:			
Aspect ratio	6.0	12.0	-
Leading edge sweep	25.0	45.0	deg
Spanwise kink position	0.2	0.45	-
Inner taper ratio	0.4	0.7	-
Outer taper ratio	0.2	0.6	-
Root thickness to chord ratio	0.1	0.18	-
Kink thickness to chord ratio	0.06	0.14	-
Tip thickness to chord ratio	0.06	0.14	-
Wing washout	2.0	5.0	deg
Fraction tip washout at kink	0.65	0.85	-
That minimise:			
Objective, f_0	-	-	-
Drag, D/q	-	-	m^2
Weight, W_{wg}	-	-	N
Subject to:			
Pitch-up margin,p	5.4	-	-
Wing volume, $\tilde{V}$	-	23.0	m^3
Undercarriage bay length, l_U	-	2.1	m

4.3.2 Comparison with Existing MDO architectures

Table 4.6 presents the results from multiple analysis runs of the blackboard, Simultaneous Analysis and Design and Collaborative Optimisation architectures. It is well known from the findings in the report by Price et. al [120] that this problem does not suit many of the existing distributed and monolithic architectures. Collaborative Optimisation in particular failed to converge both here and when applied on a problem variant by Price et. al [120]. Failures with the architectures have been noted by Alexandrov and Lewis [3] and attributed to the problem decomposition. They observed that for certain problems the system level constraints can become non-smooth, a feature which was have also confirmed by Tapetta and Renaud [145] for the Multiobjective version of Collaborative

Optimisation that was used here. This therefore explains why the solutions failed under Matlab's SQP optimiser.

For provably convex problems, the application of the Simultaneous Analysis and Design with the SQP optimiser should always converge to the global minimum. As this problem has known discontinuities in the structures domain, the global optimum cannot be guaranteed and as such, the focus of the search shifts to a demonstration of an ability to improve the initial design.

The results presented in Table 4.6 confirm that the Multidisciplinary Pattern Search is better equipped at avoiding the local minima when compared to the Simultaneous Analysis and Design architecture. The blackboard converges on average to a better result and is able to find a lower absolute minimum value. It is still however considerably slower to converge to a result than this monolithic architecture.

It is noteworthy that the best outcome from the Simultaneous Analysis and Design architecture has a profile thickness at the kink that exceeds the thickness at the root. While this geometry is entirely feasible according to our problem definition, it is unlikely to be optimal in real life. This outcome is the result of the domain optimisers exploiting weaknesses in the tool rather than a deficiency in the chosen MDO method, as a similar outcome was observed in previous optimisation studies using the same analysis tool [73].

4.4 Concluding Remarks

Many legacy MDO architectures have been developed to solve academic test cases that are artificiality far more complex than the intended real world applications. This has sometimes driven researchers to overlook practical aspects of implementation in favour of mathematical guarantees of convergence. To avoid these pitfalls, two aircraft design problems have been put forward that aim to replicate intended applications for this method - the preliminary aircraft design process. The first test case is based on a UAV wing design problem that has been developed in-house to tune and validate the performance of the Multidisciplinary Pattern Search. The second problem uses a decommissioned industrial design tool, which has been artificially decomposed to simulate a multidisciplinary design environment. Both studies demonstrate an investigative application of the blackboard and provide measurable comparisons with two competing MDO architectures.

A tuning study investigated how altering the rules, starting design space and bound control factors affected the rate and accuracy of convergence for the UAV problem. Increasing the complexity of the rule base improved the convergence rate without significantly sacrificing the ability of the method to find the global minimum. The test

Table 4.6: Aircraft design problem variables and results

Performance metrics and design variables	**Mean results**			**Lowest objective design**		
	SAND*	MOCO*	MDPS*	SAND*	MOCO*	MDPS*
Objectives						
Global objective	**0.8754**	**fail**	**0.8692**	**0.8701**	**fail**	**0.8685**
Drag	3.2753		3.2720	3.2650		3.2652
Weight	94839		93442	93920		93522
Scaled constraints failure						
	0%		0%	0%		0%
Iteration count						
Drag analysis runs	430		18270	444		13763
Mass analysis runs	430		18270	444		21029
Design variables						
Aspect ratio				10.40		10.22
Leading edge sweep				26.61		25.10
Spanwise kink position				0.213		0.446
Inner taper ratio				0.696		0.662
Outer taper ratio				0.200		0.202
Root thickness to chord ratio				0.100		0.123
Kink thickness to chord ratio				0.128		0.096
Tip thickness to chord ratio				0.060		0.060
Wing washout				5.000		4.994
Fraction tip washout at kink				0.650		0.792

*SAND and MOCO are abbreviations for Multiobjective Collaborative Optimisation and Simultaneous Analysis and Design. MDPS stands for Multidisciplinary Pattern search used with the blackboard method.

highlighted however that the use of the pattern move can increase the number of analysis evaluation, when it is rejected too often. However when combined with the additional Tabu and duplicate checks, the search provided the fastest converge on average for the problem it was tested on. In addition, it is most beneficial to start the algorithm from the absolute bound limits, rather than a smaller search space within the absolute bounds. Starting from tighter bounds increased the likelihood for the process to converge to a local minimum or infeasible design, even though the convergence rate was slightly faster. The study also showed that given a default set values for the bound movement and relaxation factors, it is advisable that the bound contraction factor should lie in the range of $0.2 - 0.5$.

The blackboard method was also compared against two competing MDO architectures. It outperformed the alternative Collaborative Optimisation architecture for both test cases that it was tested on and the converged to better result than the Simultaneous Analysis and Design during the tests on the transonic wing design problem. This leads to the conclusion that a fully integrated system like Simultaneous Analysis and Design works best if the integration effort can be tolerated and if the problem is convex. If a much less intrusive approach is needed, the blackboard (with the Multidisciplinary

Pattern Search) allows this at a much lower cost than architectures such as Collaborative Optimisation. Although these results are encouraging and a step in the right direction, more work remains to be done to reduce the number of analysis evaluations used by the blackboard. One possible way to improve the convergence, is to better utilise the history of already evaluated designs to help guide future searches. This avenue of research is the subject of the next chapter in this work.

Chapter 5

Data Mining Module

5.1 Introduction

Since design analysis is generally time consuming to compute and data storage increasingly cheap, it is sensible to store as many design calculations as possible for later reuse [74]. Sections 3.2.5 and 4.2.2 have already demonstrated how the Multidisciplinary Pattern Search can make use of legacy data to avoid duplicating results and to make better decisions about bound movements. This chapter explores a radically different concept that uses the legacy data generated by the local domain optimisers to build and tune surrogate models, which can be queried in place of the local domain analyses. This forms the concept behind the data mining module in use by the final version of the blackboard.

In conventional surrogate optimisation, designers begin by sampling the available search space [42]. The aim is to use as few data points as possible to build a cheap-to-evaluate surrogate (or otherwise known as response surface), so that it can be queried by the optimiser instead of the expensive analysis. This is subtly different to surrogate assisted optimisation, where the data points are generated directly by the optimiser and the surrogate then "mines" information from the history of evaluated solutions, so that it can be used in some way to supplement the analysis process [65]. Such data mining efforts have a rich history in the optimisation literature.

Several recent surveys of the topic discuss numerous methods [39, 65, 136] that are mostly used in conjunction with gradient-free optimisation algorithms to help reduce the number of analysis evaluations. The topic is well documented with research studies focussing on clustering [122], radial basis functions [111, 112], polynomials [48] and Kriging [18, 144] regression for the data mining process. Some authors have even combined multiple data mining ideas together [60, 67, 166]. For example the work by Isaacs et al. [60] first organises the data in clusters and then fits a separate surrogate model to each cluster. Prediction of new candidate solutions is carried out by the surrogate instead only when its cross validation error is deemed to be sufficiently low.

Xu and Yang [166] on the other hand proposed a different approach. Their method builds a Gaussian meta-model from a initial sampling plan, which is optimised to generate an initial database of optimal designs. The best designs are used as a starting points for the NSGA-II algorithm, which populates the database. Their data mining module uses clustering to identify near replica query points in a generation of NSGA-II algorithm before they are evaluated. By omitting these near replica solutions from the analysis queue, the Xu and Yang observed an improvement in the convergence rate. These two examples also illustrate that surrogate models serve different roles in a data mining assisted optimiser. Surrogates can be used to analyse possible designs in a generation and then give an indication on which ones are most likely to result in an improvement [166], or they can directly replace the exact analysis routine [60]. As this area of research is more relevant to domain level optimisation, it is beyond the scope of this thesis. This short description however has been included to highlight the parallels between the current state of the art in surrogate assisted optimisation and the idea proposed in this chapter.

While these methods for reducing the number of analysis evaluations have been widely studied in conventional single objective optimisation, they have seldom been applied in distributed MDO. Surrogate models have been used in MDO architectures to achieve design consistency or to replace the analysis functions altogether. Example of architectures in the former category are the original versions of the Concurrent Subspace Optimisation, Bi-level Integrated System Synthesis architectures and Enhanced Collaborative OPtimisation which use simple linear approximations [128, 141, 142] to achieve inter domain consistency. More recent versions of those demonstrated the use of more sophisticated Neural Networks [133]. Polynomial [82] and Kriging-based [77] models also work well. The main drawback of these architectures is that they require relatively accurate surrogate models to converge. This means that organisational domains often have to perform additional analyses evaluations, outside of their local design search, to source data for surrogate models needed by the other domains. Ultimately this can be seen as organisationally undesirable as design teams are no longer standalone and require the integration of the surrogate models into their analyses.

The second category of methods replace local domain analyses altogether. Applications in Collaborative Optimisation, demonstrate that surrogate models can alleviate the non-smoothness problems sometimes observed in the system level constraint functions [25, 139, 163]. Although these examples illustrate an overall positive impact on the process, the formulation of the Collaborative Optimisation architecture remains largely unchanged. Therefore the design process would still have to conform to the Collaborative Optimisation strategy, which can be undesirable as has already been discussed.

The Multidisciplinary Pattern Search (the combination of the rule base and the database) has already been demonstrated to solve coupled MDO problems alone without the aid of surrogates. What is noteworthy however, is that the process generates large amounts

of data that is left unused between local domain searches. This data holds valuable information that links design inputs with analysis outputs, which can be used to build and tune surrogate models that represent the domain analyses. Much like the legacy surrogate assisted optimisation methods [74], the outputs from response surface models can replace the "true" analyses when deemed accurate. Contrary to the earlier described architectures, which also employ surrogates, the proposed strategy does not require additional sampling of the design space, nor does it totally replace the domain analysis with the surrogate. It simply provides a way to reduce the number of analysis evaluation by exploiting information in the existing dataset without affecting the organisational advantages of the blackboard - i.e. data mining.

To a large extent this idea is little bit like using a lower fidelity analysis, only when the model is deemed to represent the "true" (high fidelity) analysis accurately. The main benefit of course is that because response surface models only require access to the database, they can be built in parallel and therefore need not take computational resources away from the domain local analyses. Once built/tuned and validated, these models can be queried in the same way as the analysis function inside the optimisation routine. And finally, as they are generally cheap to evaluate [43] compared to a single CFD or FEM analysis evaluation, their computational cost can be neglected from the number of analysis evaluations. This idea forms the third and final aspect of the blackboard - the data mining module. Because it is a completely new application in the framework, it warrants its own chapter in the thesis.

5.2 A review of Kriging

The surrogate in use here is the regression modelling technique known as Kriging. There are of course many other methods that can serve the data mining purpose. The textbooks by Forrester et al. [43] and Bishop [17] provide overviews of this and other alternative surrogate modelling methods. For this application, Kriging regression has been selected because it is an unassuming method that tends to work well with a low number of data points [43]. This section offers a condensed description of the process using the derivation in the work by Jones [68], while more encompassing descriptions can be found in a number of other references [42, 68].

Kriging is a way of fitting a model to an existing dataset as a realisation of a stochastic process. The main assumption is that an output from the model at a point $\mathbf{x}$ can be mapped to a random variable $Y(\mathbf{x})$, which is normally distributed with a mean μ and variance σ^2.

This allows the model to correlate the random variables $Y(\mathbf{x})$ based on input data $\mathbf{x}$. In other words, the random variables $Y(\mathbf{x_1})$ and $Y(\mathbf{x_2})$ will have a high correlation when their input data points $\mathbf{x_1}$ and $\mathbf{x_2}$ are close together. How these correlate, can

be expressed via a mathematical correlation function commonly known as a kernel. In the wider field of Gaussian process regression, users can select different correlation relationships (kernels). Kriging however, uses a specific function as given in Equation 5.1:

$$\mathbf{B} = Corr[Y(\mathbf{x_i l}), Y(\mathbf{x_j l})] = exp\left(-\sum_{l=1}^{d} \theta |\mathbf{x_{il}} - \mathbf{x_{jl}}|^p\right). \tag{5.1}$$

This specific function has the intuitive property that if the Euclidean distance between two data points is small ($\mathbf{x_{il}} - \mathbf{x_{jl}} \to 0$), then the correlation is high and goes to 1. On the other hand, if the distance is large ($\mathbf{x_{il}} - \mathbf{x_{jl}} \to \infty$), the correlation goes to 0. The additional tuning parameters θ and p influence the "width" and "smoothness" of the correlation as described in Forrester et al. [43]. θ governs how far the influence of a data point extends, whereas p governs how the correlation between two points drops off. For example, large values of θ will produce outputs that change rapidly over small distances, where as lower values for p (near 0) will model a discontinuity in the function. Altogether, the random variables $Y(\mathbf{x_1})$ to $Y(\mathbf{x_d})$ for the d data points can be grouped under a single vector $\mathbf{Y}$ as given by:

$$\mathbf{Y} = \begin{bmatrix} Y(x_1) \\ \vdots \\ Y(x_d) \end{bmatrix}, \tag{5.2}$$

where the vector of query points is:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x_1} \\ \vdots \\ \mathbf{x_d} \end{bmatrix}. \tag{5.3}$$

Now because the model assumes that $\mathbf{Y}$ is a vector of random variables with a mean μ and variance σ^2, the covariance matrix can be represented as:

$$Cov(\mathbf{Y}) = \sigma^2 \mathbf{B}. \tag{5.4}$$

Here $\mathbf{B}$ is the correlation matrix made up of elements calculated using Equation 5.1. The distribution of $\mathbf{Y}$ characterises how the model output varies as one moves in different coordinate directions. It therefore depends on the mean μ, variance σ^2 and the two tuning parameters θ and p. To estimate the best values that these can take, one is interested in reducing the generalisation error of the model by maximising the likelihood of the observed data. In other words, choosing the tuning parameters that maximise

the likelihood function intuitively means that the model will be most consistent with the available data. Given that the vector of observed values is :

$$\tilde{\mathbf{y}} = \begin{bmatrix} \tilde{y_1} \\ \vdots \\ \tilde{y_d} \end{bmatrix}, \tag{5.5}$$

the maximum likelihood function can be written as:

$$LF = \frac{1}{(2\pi)^{\frac{n}{2}}(\sigma^2)^{\frac{n}{2}}|\mathbf{B}|^{\frac{1}{2}}} exp\left(\frac{-(\tilde{\mathbf{y}} - \mathbf{1}\mu)'\mathbf{B^{-1}}(\tilde{\mathbf{y}} - \mathbf{1}\mu)}{2\sigma^2}\right). \tag{5.6}$$

In practice it is more convenient to maximise the logarithm of Equation 5.6. Its derivative - ignoring constant terms - can be shown to equal:

$$\Delta = -\frac{n}{2}log(\sigma^2) - \frac{1}{2}log(|\mathbf{B}|) - \frac{(\tilde{\mathbf{y}} - \mathbf{1}\mu)'\mathbf{B}^{-1}(\tilde{\mathbf{y}} - \mathbf{1}\mu)}{2\sigma^2}. \tag{5.7}$$

This derivative can then be set to zero to produce equations for the optimal values for μ and σ^2 as given by:

$$\mu = \frac{\mathbf{1'B^{-1}}\tilde{\mathbf{y}}}{\mathbf{1'B^{-1}1}} \tag{5.8}$$

and

$$\sigma^2 = \frac{(\tilde{\mathbf{y}} - \mathbf{1}\mu)'\mathbf{B^{-1}}(\tilde{\mathbf{y}} - \mathbf{1}\mu)}{n}. \tag{5.9}$$

To obtain the optimal values θ and p, one simply needs to substitute the estimates for σ^2 and μ back into the log likelihood function and numerically optimise it:

$$-\frac{n}{2}log(\sigma^2) - \frac{1}{2}log(|\mathbf{B}|). \tag{5.10}$$

Because this, so-called concentrated log-likelihood function (as given by 5.10), is not easily differentiable, one must use a numerical optimisation technique to find the values for θ and p that will maximise it. Now that the values for μ, σ^2, θ and p that maximise the log likelihood function have been found, the Kriging model can be used to predict the observed value $\tilde{y}*$ at a new point $\mathbf{x}*$. Logic dictates that the best estimate for $\tilde{y}*$ will be one that maximises the likelihood functions at the point $\mathbf{x}*$. Therefore, to make a prediction, one can simply add the points $\mathbf{x}*$ and $\tilde{y}*$ to the vectors x and $\tilde{y}$ as the $(d+1)^{th}$ observation. From this, one can derive the formula for $\tilde{y}*$ that maximises the

likelihood function, which is given by 5.11. Solving this equation ultimately gives the Kriging predictor:

$$y(\mathbf{x}*) = \mu + \mathbf{r}'\mathbf{B}^{-1}(\tilde{\mathbf{y}} - \mu), \tag{5.11}$$

where $\mathbf{r}$ represents the vector of correlations between the observed data and the new prediction as given by:

$$\mathbf{r} = \begin{bmatrix} Corr[Y(\mathbf{x}*), Y(\mathbf{x_1})] \\ \vdots \\ Corr[Y(\mathbf{x}*), Y(\mathbf{x_n})] \end{bmatrix}. \tag{5.12}$$

So far this description covered the key concepts for Kriging interpolation, which is only useful for predicting with data that is free from noise. Regression is more useful for the purpose of data mining here, mainly because the fixed point iteration strategy for the state variables introduces what can be described as noise, in otherwise noise-free domain analyses. One way to build a regressing Kriging model is to add a small constant to the leading diagonal of the $\mathbf{B}$ matrix [43, 56]. Following the same logic as given by Equations 5.1 to 5.11, but this time with the added λ regression constant, the prediction equation can be shown to be:

$$y(\mathbf{x}*) = \mu + \mathbf{r}'(\mathbf{R} - \lambda\mathbf{I})^{-1}(\tilde{\mathbf{y}} - \mu). \tag{5.13}$$

The value of this constant λ can also be calculated during the search for the optimal θ and p as described earlier. This concludes the tuning and evaluation descriptions for the Kriging model.

5.3 Building and Trusting the Kriging Surrogate

5.3.1 Introduction

When building/tuning the Kriging model, the main assumption is that the database has more than sufficient data to build an accurate surrogate. The general intuition in that case, is that using all the data will generate a better model [126]. While this can be true, there are several reasons why one might wish to filter the available data before it is used in the tuning process of the Kriging model.

The main reason is to avoid numerical inaccuracies. Legacy datasets from domain local optimisation searches are rarely uniformly distributed over the whole design space. This

is often because they are not generated specifically for the surrogate model, but instead are the product of formal optimisation routines or some form of human driven trade studies. Formal optimisation routines are well known to distribute points in clusters and in regions near potential optima [20]. Using this unfiltered data can lead to ill-conditioning problems in the Kriging model, which can ultimately result in unreliable predictions [43].

Another reason why it may be beneficial to discard some of the data is to reduce the tuning time for the Kriging surrogate. The optimisation of the Kriging parameters requires many matrix solutions, which can be computationally time consuming when the datasets are large. Forrester et al. [43] showed that Kriging surrogates have data saturation limits. Beyond that, additional data points will simply increase the tuning time without improving the global model prediction. However, as the domain local analyses are assumed to be prohibitively more time consuming in the grand scheme of the MDO process, the tuning time for the Kriging surrogate is ignored when comparing results.

Cross-validation is yet another reason why one might choose not to use all the data. Cross-validation is a well known [17, 43] method for testing the prediction accuracy of a model and requires the partitioning of the data, usually at random, into training and validation sets. The former is used to tune the model, while the latter can be used to test the how accurately the model predicts on independent data points that have not been used in the tuning process.

The final aspect that needs consideration is the data outside the current bound space. It is important to remember that the Multidisciplinary Pattern Search iteratively moves the bounds on the shared variables. So at any stage the databases will be populated with data points, some of which will lie well outside of the region of interest at that particular iteration. It would be wasteful to build the model with all the available data when only a subset of that data is needed. How much of the data outside of the current area of interest is useful, is a question addressed by legacy trust region methods [148, 149]. Figure 5.1 illustrates an example re-cycle region idea used in the monolithic MDO method by Ollar et al. [110]. In a similar work, Toropov and Alvarez [148] give guidance that the re-cycle region should be between 1.5 to 2.0 times larger than the current search domain.

5.3.2 The Data Mining Module

The data mining module consists of a set of rules that automatically decide if there is sufficient data to build a Kriging model and if the this model can be trusted to make accurate predictions of the "true" analysis evaluation. Figure 5.2 illustrates the key steps in the process. The Kriging toolbox in use here is a modified version of the programs

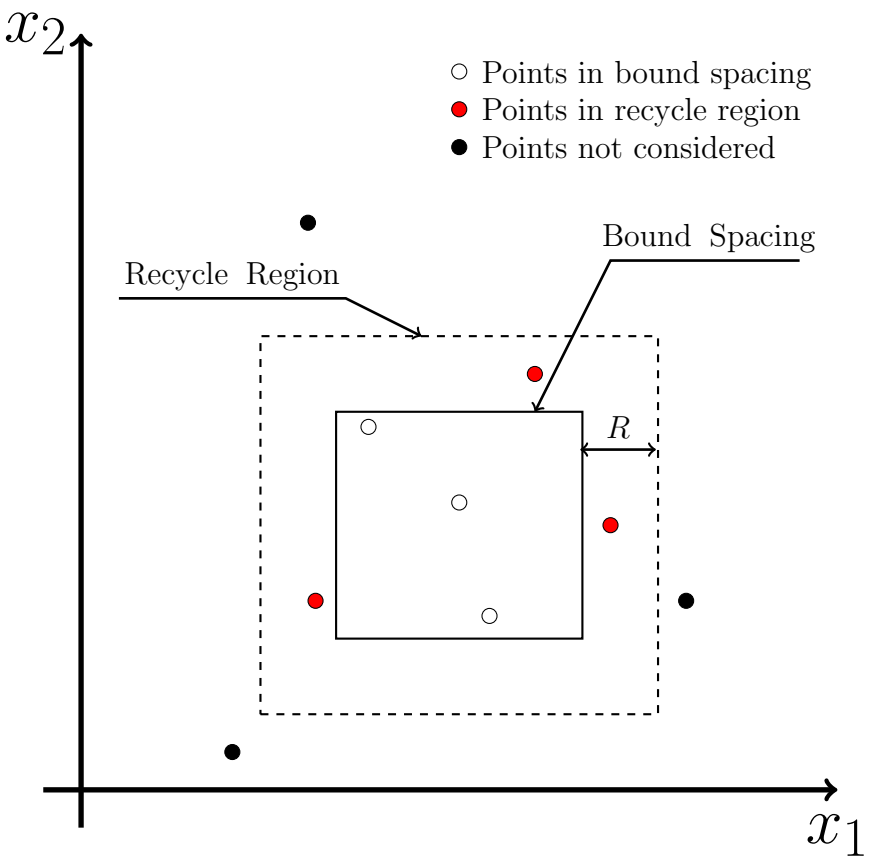

Figure 5.1: Points outside of the re-cycle region are not used for tuning the Kriging model (adapted from Ollar et al. [110])

published in the book by Forrester et al. [43], while the rules in the data mining module have been put together in-house.

The first step tackles the question of how to select a subset of the data for efficient Kriging training. Srivastava et al. [144] and Rennen et al. [126] tested several ways to filter large legacy data sets for Kriging generation. The comparison study of Rennen et al. [126] failed to provide a "best" method, but concluded that in general more points resulted in a better prediction. This was also observed in the results presented by Srivastava et al. [144], which is perhaps an indication that the original dataset was both rich and uniformly populated.

The data mining module here has been specifically developed for use on the more complex version of the UAV wing design test case, which is outlined in Section 4.2. This test case is arguably a worst case scenario, because the data is generated using the SQP gradient-based optimiser. This means that the available data is both scarce and found in very tight clusters where gradients have been calculated using the finite differencing method. So the first filter removes points that sit close together simply by rounding data points to 7 decimal places and discarding duplicate solutions. This was done to alleviate ill-conditioning problems, which may occur during the matrix inversions [43].

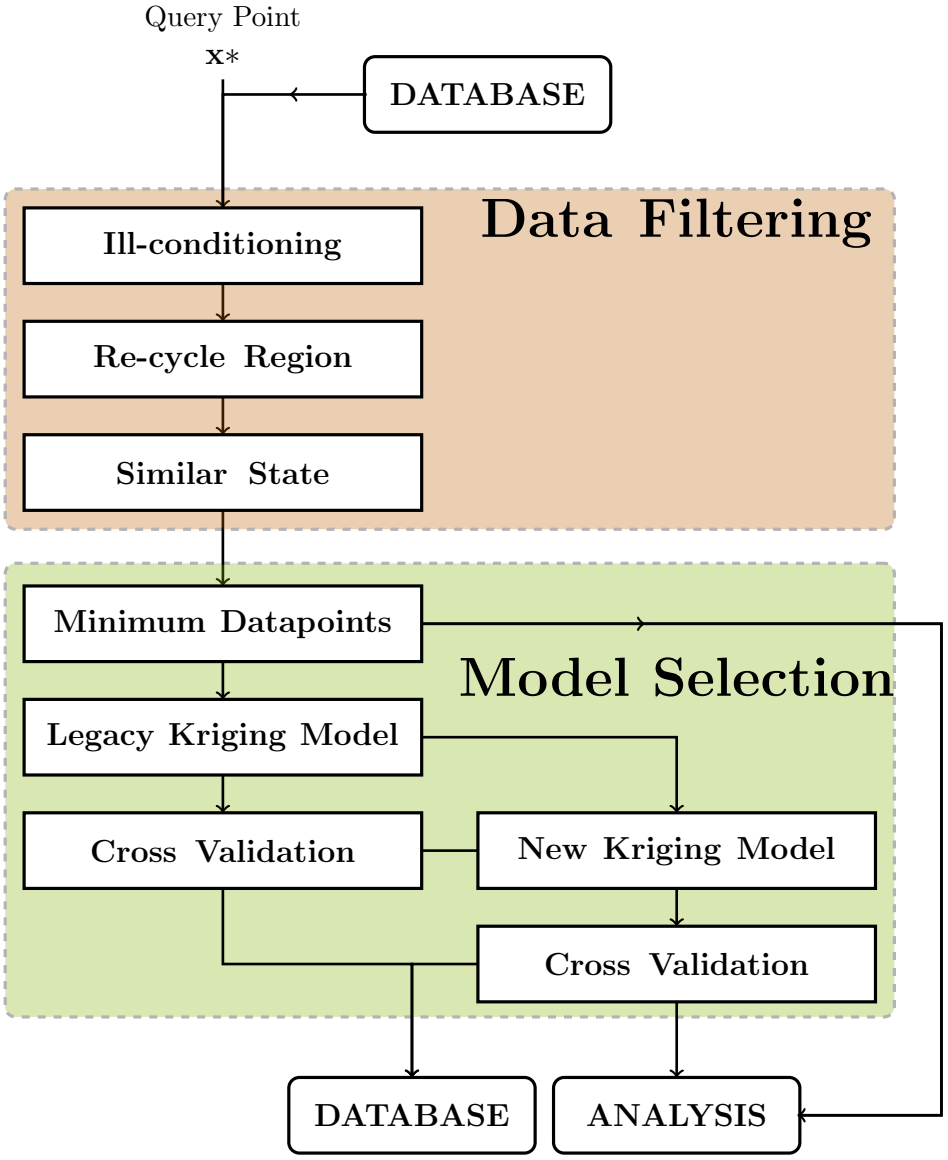

Figure 5.2: Data mining process flow chart

During a search, it is better to build a model that is accurate in the region of interest, rather than one that tries to capture the entire design space [60]. So the second filter removes points outside the vicinity of the current bound spacing region. The re-cycle region scheme was introduced to maximise the available data for building the Kriging model. Figure 5.1 illustrates this concept, while Equation 5.14 was found to generate a recycle region that varies with the bound spacing and RR worked well in practice:

$$RR = 0.3(ub - lb)\left(1 - \frac{ub - lb}{ub_{init} - lb_{init}}\right)^3. \tag{5.14}$$

As a reminder, the values ub and lb signify the current upper and lower bounds, while the symbols with subscript $init$ refer to the initial starting bounds.

The final data filter removes points with respect to the state variables. The data used to tune the Kriging model takes in set of design variables, which also includes state variables. This is because the prediction is generally more accurate when the Kriging model includes state variables relevant to the analyses routine being in the training set. However, the blackboard uses a fixed point iteration scheme, which means the input state variables remain fixed throughout a domain local search. It is therefore beneficial to discard points points from the training data which have radically different state variables from the current query point. More specifically, this filter removes data points that have more than 5% different, which can be defined as:

for all data points
  if $\frac{y*}{\tilde{y}} > 0.05$
    Disregard data point from training data.
  end
end

The symbol $y*$ is the fixed state variable in the current domain level iteration and $\tilde{y}$ represents the state variable from the training dataset. This concludes the data filtering section of the module. What follows next is the description of the pre-defined rules that determine whether the Kriging model will be used in place of the "true" analysis evaluation. The first rule checks if the remaining filtered dataset is sufficiently well populated for use by the Kriging model. If it is not, the data mining module returns a flag to indicate that a "true" analysis evaluation should be used instead.

The next step checks if a Kriging model was successfully used in the previous iteration. This aims to use legacy θ, p and λ tuning parameter, because optimising these can be a time consuming process [18]. Although it can be claimed that the time required to tune these parameters is small in comparison to a single CFD or FEM analysis evaluation [18], the need to run this process over multiple iterations necessitates every possible time saving. Furthermore, given that bound movements generally overlap between different areas of the designs, it would be wasteful to tune new Kriging parameters, when an set has been tuned on a near identical data set. So Kriging parameters are inherited when two successive datasets overlap by 80%.

In the event that no suitable previous Kriging parameters exist, the algorithm tunes a new model with the filtered data. Regardless of whether new or existing Kriging parameters are used, the "leave one out" cross validation method (using the filtered training data) assesses the prediction accuracy [43] of the model. In addition the model is also cross validated with the discarded data from inside the re-cycle region. Although, this has much lower impact on the decision of whether to use the Kriging model because this data is usually near identical to the data used in tuning the model.

A 2% mean prediction error is used as a threshold for accepting or rejecting a model. If either the new or legacy Kriging models pass this threshold, the prediction from the Kriging model is accepted and saved to the database. If however the legacy Kriging model fails, a new Kriging model is tuned using the dataset. A flag to indicate that the query point should be evaluated with the "true" analyses is generated if a new Kriging model fails the cross validation check. This concludes the description of the data mining module.

5.4 A Case Study Using the Complex UAV Wing Design Problem

This section evaluates the benefits of using the data mining module on the full version of the UAV wing design problem from Section 4.2.1.8. Separate Kriging models were built to evaluate the drag, wing loading, stress, mass and costing instead of the "true" analysis functions. One would expect that with the addition of the data mining, the Multidisciplinary Pattern Search would converge in fewer analysis evaluations. This is certainly the case as shown by Figure 5.3, which illustrates the convergence from the same point with and without the data mining module. More specifically, the figure shows that the additional data mining modules reduces the number of "true" analysis evaluations by around one third.

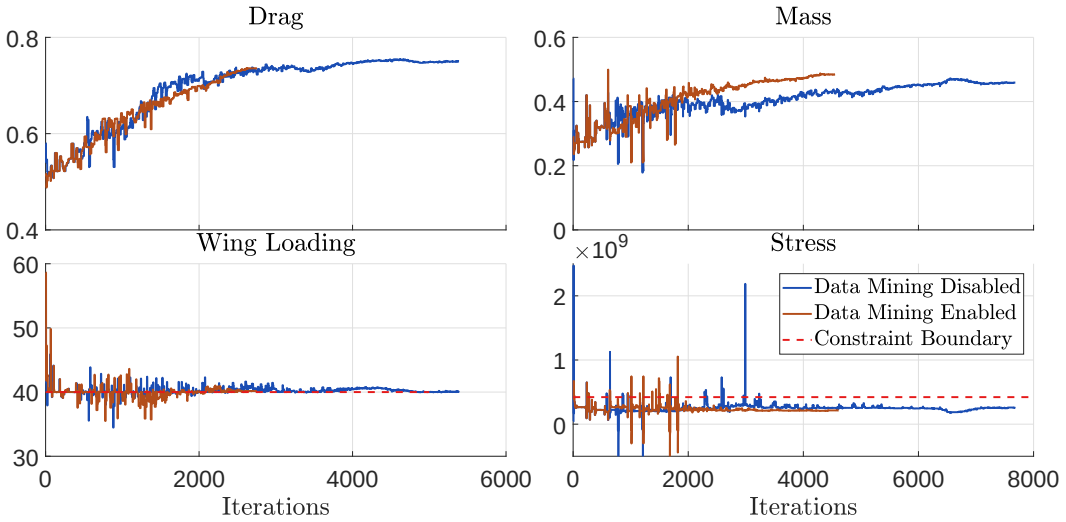

Figure 5.3: Demonstration of improved convergence rate using the data mining module

However, it is well known that to using the Kriging predictions can mislead optimisation searches. This phenomenon was observed when the blackboard method was started from multiple starting points. Figure 5.4 shows that the data mining module introduces much

larger scatter and more frequent constraint failures in the final designs when compared to the same version of the blackboard with the data mining disabled (Figure 5.5).

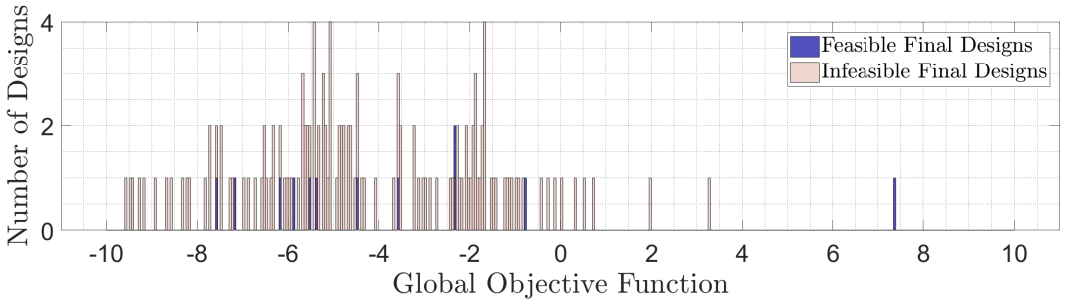

Figure 5.4: Data mining assisted Multidisciplinary Pattern Search convergence for 150 points on the more complicated version of the UAV problem

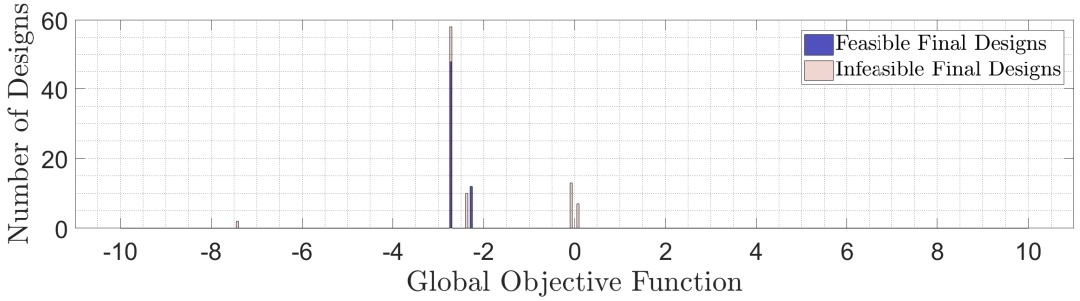

Figure 5.5: Multidisciplinary Pattern Search Convergence for 150 points on the more complicated version of the UAV problem without data mining

While there are numerous factors that affect the prediction accuracy of the Kriging model, the biggest factor that contributes to these issues is the available data. When building the Kriging model, most of the data has to be discarded to prevent ill-conditioning or because the state variables do not match. This usually leaves insufficient and poorly distributed data for tuning and cross-validation of the Kriging model, and as such makes the decisions on whether to use the Kriging model challenging. Tests that investigated tighter decisions thresholds showed diminishing improvements, further supporting the conclusion that the available data was insufficient

It is also noteworthy that the blackboard without the data mining exhibited a higher constraint failure than in previous applications. This indicates that the chosen penalty function becomes less effective in dealing with constraints, when the absolute value of the global objective increases.

Table 5.1 summarises the final designs points from both studies. It is interesting to note that the even without the data mining module, the average constraint failure is much larger than previous studies performed in Section 4.2.4. This is perhaps an indication that the chosen constraint handling techniques function is less effective for problems whose objective function takes on larger absolute value.

Table 5.1: Summary of results for 150 runs started from different points arranged in a Latin Hypercube pattern

Performance metrics and design variables	**Mean results**		**Lowest objective design**	
	Data mining disabled	Data mining enabled	Data mining disabled	Data mining enabled
Objectives				
Global objective	**-2.6257**	**-3.6486**	**-7.4011**	**-9.5953**
Cost	16.08	16.59	12.38	11.45
Range	18.71	20.24	19.78	21.05
Scaled constraints failure				
	1.83%	11.25%	3.99%	2.91%
Iteration count				
Bound moves	338	386	451	430
Total domain calls	58009	43428	81731	48236
Design variables				
Semi-span			0.8160	0.7728
Root chord			0.2345	0.2496
Taper			0.9973	0.8893
Maximum thickness to chord ratio			0.1472	0.0910
Wall thickness			0.0008	0.0007
Outer diameter			0.0095	0.0092

*SAND and MOCO are abbreviations for Multiobjective Collaborative Optimisation and Simultaneous Analysis and Design. MDPS stands for Multidisciplinary Pattern search used with the blackboard method.

5.5 Concluding Remarks

Distributed MDO methods generally require an order of magnitude more analysis evaluations than the monolithic counterparts. As shown in previous chapters, the proposed blackboard method with the newly developed Multidisciplinary Pattern Search is no different. This chapter introduces a concept that aims to improve the rate of converge simply by regressing among the existing data in order to replace the costly analyses evaluations where possible.

This data mining module included a set of Kriging models, which were built and updated only using legacy data from the database. Its use in the blackboard improved the convergence by around 25% but inaccuracies in the Kriging model forced the method to converge to designs with a significantly higher constraint failure. Basically the data mining works best if optimisers in use by the domains generate data that is rich and well spaced. When an optimiser, such as SQP is in use, which generally tends to produce poorly spaced clusters of points, the data mining assisted search should be used with caution as the Kriging models can mislead the search and drive it to converge to infeasible designs.

This final addition completes the work on the blackboard as a computational entity for MDO. The next important question is how intuitive is the process and can it be practically used by designers to help reach multidisciplinary optimal designs. This makes up the topic of the next chapter.

Chapter 6

Team-Based Application

6.1 Introduction

Work on the blackboard framework has been developed as part of a collaborative partnership between Airbus UK. and the University of Southampton. A focus of this effort is the continuous improvement of various aspects of the framework with the ultimate intent being its application in an industrial setting. When developing a MDO process, one must not lose sight of fact that the design teams will ultimately undertake the work. Most researchers working on MDO to date tend to focus on running computational experiments, which imitate the interactions between design teams. Although these are useful because they allow researchers to quickly identify and address performance issues, such experiments inevitably fail to capture the human factors involved in design. This has resulted in a great deal of distributed MDO literature that has seen relatively little use in industry, primarily because the architectures have been developed without due regard to the needs of the design teams that will ultimately use them [73].

Directly testing a novel MDO architecture in an industrial setting risks disrupting the design cycle and could contribute to costly delays inside the organisation. Hence why, excluding a small number of select cases studies [96], it is rare to see new distributed MDO approaches being directly tested by industrial design engineering teams. One way to avoid this issue, is to use student design teams instead. The abundance and shorter time scales of academic design projects, as well as the much lower consequences of failure, makes this form of real world testing a good stepping stone in the validation process of a new method.

There are numerous examples where engineering students have been subject to research studies. Perhaps the most relevant is a study by Austin-Breneman and Honda [14], where the authors investigated how student teams coped with distributed MDO problems. The results concluded that in all cases the student teams failed to find the optimal design

when compared to more formal optimisation strategies. This was largely attributed to the teams' use of trial and error-based approaches instead of formal optimisation methods. Factors related to team dynamics also played a role in the failure. Other studies have also investigated engineering negotiation strategies [66], team performance on academic projects [167] and how decision making differed between experts and students [13].

Computational trials using the blackboard have thus far provided satisfactory design solutions on representative problems. So the focus of this chapter has been the application of the blackboard in a team-based design environment, with the goal of obtaining qualitative feedback on its intuitiveness, effectiveness and ease of use.

6.2 The Research Approach

Undergraduates undertaking the *FEEG2001 Systems Design and Computing* module at the University of Southampton were tasked to design and build of an airplane wing as part of their coursework. They were provided with a standard fuselage, empennage and power plant for a sub 2kg UAV, and were given considerable freedom to design and build a wing. At the end of the module, the student built wings were scored on two main criteria as part of their assessment.

These criteria are the two main objectives in the student design problem. The first objective was to maximise aerodynamic efficiency. This was defined as the difference between the maximum speed in level flight and minimum (stall) speed. The second objective pushed students to maximise the structural efficiency of their wings. This is defined as the a ratio between the wing deflections and its mass when a point load was applied to the structure.

This problem lends itself very nicely to the analyses used by the simple UAV wing design problem described in Section 4.2. A spreadsheet-based design tool was put together to help minimise certain aspects of the two objectives in this exercise. The tool included a simplified variant of the blackboard framework to help size the main wing variables in the early preliminary design stage.To keep the tool simple to use, the automatic Multidisciplinary Pattern Search and data mining module were omitted from this formulation of the framework. Instead students had to play the role of the pattern searcher and manually move and reduce the bounds on the shared variables space to find an optimal design region. Microsoft Excel was specifically chosen to host the blackboard design tool, in order not to deter students with unfamiliar software. After all, the teams were free to use it at their discretion and therefore making the application itself easy to use was fundamental to the test. The tool incorporated the analyses from the simplified version of the UAV problem into two optimisation domains as shown in Figure 6.1. Each

domain was optimised with Excel's "Solver" optimiser and used the simplified problem decomposition as described in Section 4.2 with the added deflection constraint.

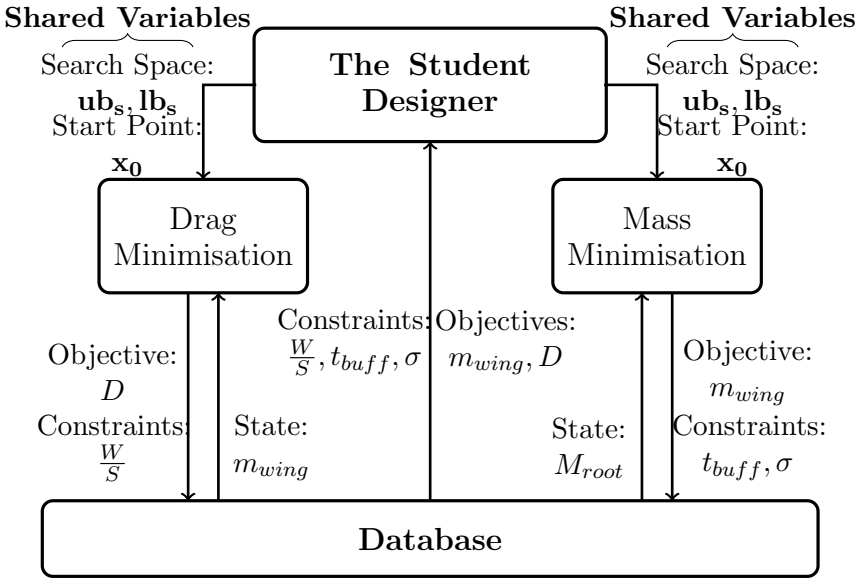

Figure 6.1: Blackboard analysis tool used for the student design exercise

Figure 6.2 shows the main GUI that facilitates the bound movements. Along the left side of the GUI, there are two scrollbars that control the bounds for each variable. The outermost scrollbar moves both bounds either upwards or downwards, while the other controls the bound separation. Once the student selects a desired search region with the scrollbars, domain level optimisations can be trigged using the "Start Analysis" button on the right side. The five central graphs then become populated with the preferred final designs from each local optimiser. The top four graphs display the preferred design values from each domain, while the last graph displays the best obtainable drag and mass, and any scaled constraint failures for the region of interest. The two colours distinguish the outputs from the different domains. In many ways and deliberately, this GUI is very similar to the one presented in Section 3.4.

What is notable to see in Figure 6.2 is that the domains disagree on three of the four design variables over successive iterations. This is indicative of the presence of conflicting objectives and is up to the student to manage these manually. Effectively the student replaces the Multidisciplinary Pattern Search, as they set the available search space for automatic domain level optimisation. To comprehend why the Multidisciplinary Pattern Search has been omitted, one must remember that the subject students are second year undergraduates, with little to no prior knowledge of the aircraft design process or MDO. Although this makes them good candidates, as they will likely have no prejudices towards certain design practices, they are largely unfamiliar with basic concepts in design optimisation. So they were given direct control over the bound movements to prevent a "push-button" MDO solver and to make the overall search simpler to grasp. This also

allowed students' design logic to be observed by noting their choices of moves and to focus on establishing how easy it is to find feasible designs using bound movements.

In addition, it was deemed unnecessary to introduce a weighted objective function, because objective weightings have no physical meaning real world design and would further complicate the process. Instead, the compromise between drag and mass was achieved by students visually monitoring changes over consecutive iterations.

6.3 Results and Discussion

The course was composed of a mixture of mechanical and aerospace students altogether numbering 80, which were randomly allocated into 16 teams. In total 11 of the 16 groups used the output from the spreadsheet tool as the starting geometry for their preliminary assessment stage. Three groups found the program unintuitive, one group had decided on a wooden skeleton wing and used other methods to generate an initial geometry and one group gave no reason. Individually, students that attempted to use the tool were asked to complete a focus group type questionnaire, which can be found in Appendix A. The questionnaire obtained feedback on three elements of the tool: how intuitive they perceived the process to be, how easy it was to find a final design and the effectiveness of the GUI and process combination. There were also comment sections that allowed students to provide specific feedback, as to why they did not use tool (if they didn't) and any changes they would propose.

The questionnaires were filled out during a presentation review of each group's progress at the mid-way stage. As not every member in each group attempted to use the tool, only those that did were asked to fill out the questionnaire. 28 anonymous responses were collected, of which 21 had answered that they had successfully used the tool in their design.

Although most students that attempted to use the tool were successful, nearly half of participants noted that the provided guidance notes had insufficient detail. This indirectly contributed to students failing to understand the types of inputs required or failing to enable the "Solver" optimiser add-in in Microsoft Excel. Nevertheless those students that sought help during the weekly sessions and were present at the introductory lecture were able to use the tool to generate candidate wing geometries. Figures 6.3, 6.4 and 6.5 summarise the responses to the three questions from the 21 participants that answered positively.

Overall, Figure 6.4 shows that most students found the process intuitive. This indicates that manually altering the bounds on the shared design variables using the provided spreadsheet tool was concept that easy to grasp. What is interesting however, is that a lot of students did not find it easy to locate a final result. Some of the written feedback

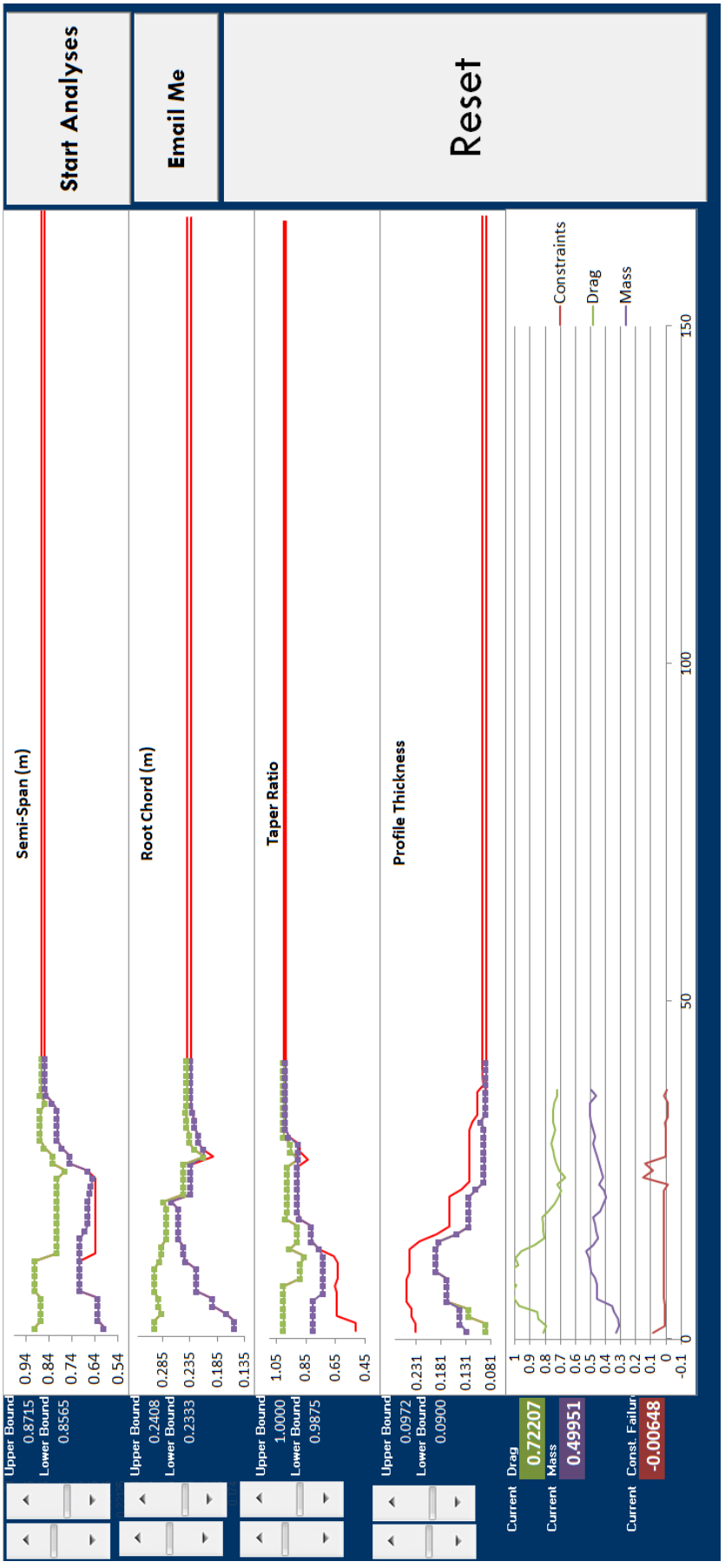

Figure 6.2: Graphic user interface for student design tool

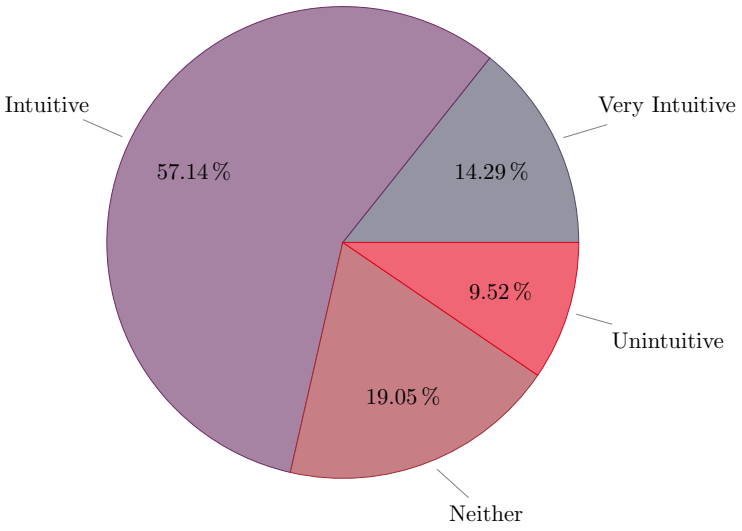

Figure 6.3: Answers to the question: How intuitive did you find bound movements as a design method?

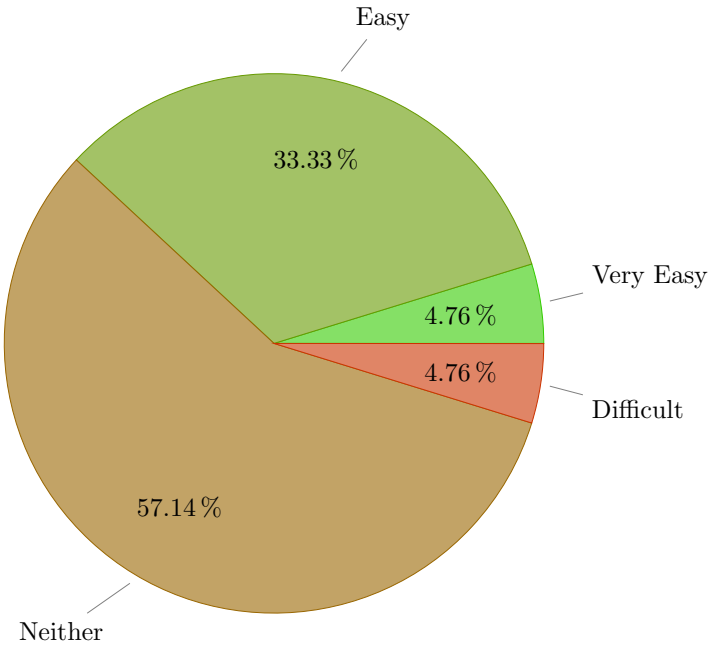

Figure 6.4: Answers to the question: How easy was it to find a final design?

from students indicated that it was not clear why certain bound movements caused constraint failures. Student wished to see not only which constraints were failing, but also what could be done to remedy the failures. This is supported by Figure 6.5 where a third of participants wished to see more information presented by the GUI.

The selected starting inputs could have also introduced difficulties in finding a final answer. As students were free to choose the bound limits and constraints, they may have inadvertently over-constrained the MDO problem, which would make the manual search harder. It is difficult to tell how much this bedevilled the search, however the

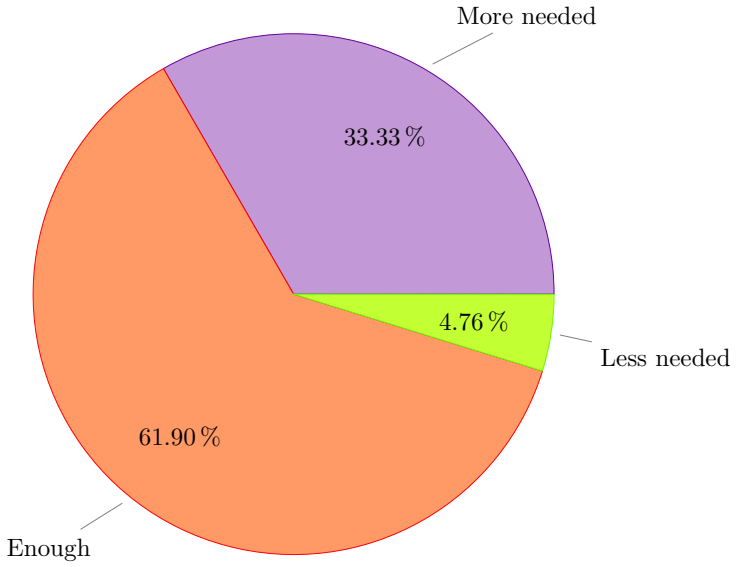

Figure 6.5: Answers to the question: Was there enough information displayed?

feedback suggested that the process was easy to use once the rather steep learning curve in the initial set-up was overcome.

Moving forward, it is worth reiterating that the results presented here are an evaluation of the difficulty of manually altering the bounds to find a multidisciplinary feasible result. The automatic search using the Multidisciplinary Pattern Search has been excluded from the tool, to keep the complexity of the method at a level, which could be easily understood by undergraduate by students. This can makes the process much more intensive for the student designer, than the version presented in Chapter 3, which requires no designer input altogether.

This explains why some students found the GUI inadequate for the purpose and wished to see more information on the causes of constraint failures as well as advisable moves that would improve the objectives. This feedback opens an avenue for future research on a hybrid automated blackboard approach. The rule base in the proposed approach would continue to automatically set new design regions, but the chief designer would maintain the ability to override these decisions. They could manually move the search in directions of their preference and query the rule base for suggested future moves. This hybrid approach which might give the best of both worlds - efficient convergence, while keeping designers engaged. For such a method to work well, the blackboard would require a rule base that can handle human interruption and a more sophisticated GUI. In many ways this would be a return to the concept of computational steering, which has been applied to analysis and optimisation methods in the past [104, 158, 159].

6.4 Concluding Remarks

Perhaps the best result of the student tests is the ability to demonstrate that a MDO method based on simple bound movements can be successfully used in a team-based environment. Such tests are rarely found in industry because engineers have very little bandwidth outside of their daily tasks to undertake academic studies. This is why a student-based research exercise was undertaken to establish how intuitive of the of the blackboard framework when applied to a team-based design problem.

A bespoke version of the blackboard search process was made available to 16 student groups in a standalone spreadsheet-based design optimisation tool. The version excluded the Multidisciplinary Pattern Search and the data mining modules to keep the process complexity to a level where it could be easily understood by relatively inexperienced design students. Instead designers had to manually move the bounds on the shared variables in their search for an optimal design region.

The study provided numerous findings and highlighted several avenues for further research. First, to keep designers engaged with the process, they require considerable information regarding the outcomes of bound alterations. The status of objectives and constraints alone is insufficient to make good design choices. The students, which had the role of the chief engineers, wished to see why certain bound movements resulted in design improvements. More importantly, they required feedback on why certain bound movements resulted in failures and suggestions on future moves that could improve those. The feedback from this study highlighted scope for future research on a mixed human-automated search process, which could offer benefits over the automatic or maul driven bound movements process explored in this and previous chapters.

Chapter 7

Conclusions

7.1 Discussion

The lack of industrial application of distributed and automated MDO architectures, particularly in aircraft design, has been the main motivation of this research work. A legacy blackboard framework was deemed the most promising avenue for further research and provided the foundation for the work covered in this thesis. The core concept centres around the use of a rule base to iteratively reduce the bounds on shared design variables until they converge on a search space representative of a single multidisciplinary optimal design.

This thesis covers two main advancements on the legacy framework. The first is a newly developed rule base has been given the name Multidisciplinary Pattern Search by the author to highlight that it has been inspired by several well established heuristic optimisers. It was developed and tested on two aircraft design problems and its performance was validated against two competing MDO architectures. The results indicated that when combined with a blackboard, it converges faster than the distributed Collaborative Optimisation. Yet it was shown that for a convex MDO problem, it was not able to match both the accuracy and lower convergence rate of the monolithic Simultaneous Analysis and Design architecture. However, the gradient based optimisers often used with such fully integrated architectures were shown to become less effective when faced with problems with internal discontinuities. Simultaneous Analysis and Design was found to be sensitive to starting points and performed less competitively than Multidisciplinary Pattern Search.

Despite the improvements in the rule base used by the blackboard, the process still requires a considerably high number of analysis evaluations to convergence. This is well beyond what is practically achievable in the preliminary aircraft design stage using high fidelity physics-based codes. For this reason a separate data mining module was

developed to reduce the number of analysis evaluations by reusing legacy data stored in the database. More specifically, a set of Kriging regression models were built using legacy data, which were then used instead of the "true" analysis evaluations. Tests on the UAV design problem indicate that 25% reduction in the number of analyses evaluations can be achieved. A sacrifice in robustness was necessary, as the surrogate models can sometimes mislead the search process and divert it towards infeasible design regions.

The blackboard model with manual bound controls was made available to groups of engineering students for a UAV design task. Their feedback suggested that the process altogether was intuitive, but the provided GUI required more work. The students wanted to see specifically which manual bound movements caused constraint failures and what could be done to remedy those.

Finally, it is worth reiterating a fully integrated monolithic architecture works best on MDO problems whose domain analyses are well suited to gradient based optimisation. That way the integration cost and organisational impact can be offset by guarantees and high rates of convergence. If a less intrusive method is needed, the blackboard process offers an intuitive way to control and monitor the activity of organisational domains, which might appeal to chief designers without a specialist MDO background.

7.2 Suggestions for Future Work

Significant progress has been made towards getting the legacy blackboard framework to a stage where it could be applied to industrial design. However, more work still remains to be done as the process is some way away from being ready for industrial trials.

First, the core logic in the Multidisciplinary Pattern Search is robust and has reached a level of maturity where it could be applied to more complex problems. However more work needs to be done on the constraints penalty function. As discussed in Section 5.4, a significant constraint failure was observed in the final design found by the process. This is indicative that the chosen penalty function is sensitive to the absolute value of the global objective. Hence a better penalty function would be the first avenue for future work.

Second, there is much more to be done with the data mining module. The author is well aware the blackboard is unsuitable for some highly non-linear problems, which are often found in academic test suits [113]. This is because it uses a fixed point iteration strategy to communicate state variables across domains and has been implemented to avoid a full multidisciplinary analysis at every iteration. There is scope to apply regression models in the domain local analyses, which should improve the consistency of the individual

domain preferences. This will likely also improve the performance of the blackboard when applied to non-linear MDO problems.

Finally, a practical application on a team-based design problem concluded that more work is needed on the human process interaction in the blackboard. A fully automated approach takes the designer out of the loop, while manually controlling the bounds can be inefficient, especially when performing routine exploratory bound movements. Perhaps a hybrid human/automated blackboard would offer the best of both worlds - efficient use of designer time, while keeping chief designers engaged with the process. This could be as simple as pre-determined breaks in the Multidisciplinary Pattern Search to enable human input or a complete re-design of the rules to allow for unpredictable human interruption of the search process. This avenue offers the broadest scope for work and future research in this area can be tailored to specific applications.

Appendix A

Student Questionnaire

Did you use the UAV Design Spreadsheet in your wing design?

- ☐ Yes
- ☐ No

If no, why not?

How intuitive did you find bound movements as a design method?

- ☐ Very Intuitive
- ☐ Intuitive
- ☐ Neither
- ☐ Unintuitive
- ☐ Very unintuitive

How easy was it to find a final design?

- ☐ Very easy
- ☐ Easy
- ☐ Neither easy
- ☐ Difficult
- ☐ Very difficult

Was there enough information in the "User Control" to help you make decisions about the bounds?

- ☐ More information was needed
- ☐ Information displayed was enough
- ☐ There was too much information

Is there anything that you would change?

Figure A.1: Student feedback questionnaire for the Blackboard analysis tool

Appendix B

List of Publications

This research project has resulted in the following publications:

1. N. Jelev, A. Keane, and C. Holden. A pattern search algorithm for blackboard based multidisciplinary design optimisation frameworks. *Journal of Aircraft*, 0(0):1–16, 2018.

2. N. Jelev, A. Keane, C. Holden, and A. Sóbester. Rule based architecture for collaborative multidisciplinary aircraft design optimisation. *International Journal of Aerospace and Mechanical Engineering*, 11(5):1006–1015, 2017.

References

[1] ESDU Aerodynamics Committee, Method for the rapid estimation of spanwise loading of wings with camber and twist in subsonic attached flow, ESDU 83040 *ESDU*, Jan. 2013.

[2] J. Agte, O. De Weck, J. Sobieszczanski-Sobieski, P. Arendsen, A. Morris, and M. Spieck. MDO: assessment and direction for advancements - an opinion of one international group. *Structural and Multidisciplinary Optimization*, 40(1-6):17–33, 2010.

[3] N. M. Alexandrov and R. M. Lewis. Comparative properties of collaborative optimization and other approaches to MDO. NASA CR - 1999 - 209354, 1999.

[4] N. M. Alexandrov and R. M. Lewis. Analytical and computational aspects of collaborative optimization for multidisciplinary design. *AIAA Journal*, 40(2):301–309, 2002.

[5] D. L. Allaire, K. E. Willcox, and O. Toupet. A bayesian-based approach to multifidelity multidisciplinary design optimization. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference, AIAA 2010-9183, Fort Worth, Texas, USA*, 2010.

[6] C. Allen and T. Rendall. Computational-fluid-dynamics-based optimisation of hovering rotors using radial basis functions for shape parameterisation and mesh deformation. *Optimization and Engineering*, 14(1):97–118, 2013.

[7] C. B. Allen, D. J. Poole, and T. C. Rendall. Wing aerodynamic optimization using efficient mathematically-extracted modal design variables. *Optimization and Engineering*, 19(2):453–477, 2018.

[8] J. Allison, M. Kokkolaras, M. Zawislak, and P. Y. Papalambros. On the use of analytical target cascading and collaborative optimization for complex system design. In *6th world congress on structural and multidisciplinary optimization, Rio de Janeiro, Brazil*, pages 1–10, 2005.

[9] J. Allison, D. Walsh, M. Kokkolaras, P. Y. Papalambros, and M. Cartmell. Analytical target cascading in aircraft design. In *Proceedings of the 44th AIAA aerospace sciences meeting and exhibit, AIAA-2006-1326, Reno, Nevada, USA*, 2006.

[10] J. T. Allison, M. Kokkolaras, and P. Y. Papalambros. On selecting single-level formulations for complex system design optimization. *Journal of Mechanical Design*, 129(9):898–906, 2007.

[11] J. Alsina. *Development of an aircraft design expert system.* PhD thesis, Cranfied University, Cranfield, UK, 1988.

[12] J. Arora. *Introduction to optimum design.* Academic Press, 3rd edition, 2012.

[13] C. J. Atman, R. S. Adams, M. E. Cardella, J. Turns, S. Mosborg, and J. Saleem. Engineering design processes: A comparison of students and expert practitioners. *Journal of Engineering Education*, 96(4):359–379, 2007.

[14] J. Austin-Breneman, T. Honda, and M. C. Yang. A study of student design team behaviors in complex system design. *Journal of Mechanical Design*, 134(12):124504, 2012.

[15] M. Bell and M. C. Pike. Remark on algorithm 178 [e4] direct search. *Commun. ACM*, 9(9):684–685, Sept. 1966.

[16] S. Bharti, M. Frecker, and G. Lesieutre. Optimal morphing-wing design using parallel nondominated sorting genetic algorithm II. *AIAA Journal*, 47(7):1627–1634, 2009.

[17] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer-Verlag, Berlin, Heidelberg, 2006.

[18] F. Bittner and I. Hahn. Kriging-assisted multi-objective particle swarm optimization of permanent magnet synchronous machine for hybrid and electric cars. In the *International Electric Machines & Drives Conference (IEMDC), Chicago, Illinois, USA*, pages 15–22. IEEE, 2013.

[19] P. T. Boggs and J. W. Tolle. Sequential quadratic programming for large-scale nonlinear optimization. *Journal of Computational and Applied Mathematics*, 124(1-2):123–137, 2000.

[20] A. J. Booker, J. E. Dennis, P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17(1):1–13, 1999.

[21] R. D. Braun and I. Kroo. Development and application of the collaborative optimization architecture in a multidisciplinary design environment. *Multidisciplinary Design Optimization: State of the Art*, edited by Alexandrov, N., and Hussaini,

M. Y., Society for Industrial and Applied Mathematics, Philadelphia, 1997, pages 98–116.

[22] R. D. Braun, A. A. Moore, and I. M. Kroo. Collaborative approach to launch vehicle design. *Journal of Spacecraft and Rockets*, 34(4):478–486, 1997.

[23] N. F. Brown and J. R. Olds. Evaluation of multidisciplinary optimization techniques applied to a reusable launch vehicle. *Journal of Spacecraft and Rockets*, 43(6):1289–1300, 2006.

[24] S. E. Carlson. Genetic algorithm attributes for component selection. *Research in Engineering Design*, 8(1):33–51, 1996.

[25] H. Chagraoui and M. Soula. Multidisciplinary design optimization of stiffened panels using collaborative optimization and artificial neural network. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 0(0):1–17, 2017.

[26] S. Chen, F. Zhang, and M. Khalid. Evaluation of three decomposition mdo algorithms. In *Proceedings of 23rd International Congress of Aerospace Sciences, Toronto, Canada*, 2002.

[27] I. R. Chittick and J. R. R. A. Martins. An asymmetric suboptimization approach to aerostructural optimization. *Optimization and Engineering*, 10(1):133–152, 2009.

[28] S. Choi, J. J. Alonso, and I. M. Kroo. Two-level multifidelity design optimization studies for supersonic jets. *Journal of Aircraft*, 46(3):776–790, 2009.

[29] C. A. C. Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245–1287, 2002.

[30] A. M. Connor and D. G. Tilley. A tabu search method for the optimization of fluid power circuits. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 212(5):373–381, 1998.

[31] J. Cousin and M. Metcalf. The bae (commercial aircraft) ltd transport aircraft synthesis and optimisation program (tasop). In *Aircraft Design, Systems and Operations Conference, Dayton, Ohio, USA*, page 3295, 1990.

[32] E. J. Cramer, J. Dennis, Jr, P. D. Frank, R. M. Lewis, and G. R. Shubin. Problem formulation for multidisciplinary optimization. *SIAM Journal on Optimization*, 4(4):754–776, 1994.

[33] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2-4):311–338, 2000.

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

[35] N. R. Deepak, T. Ray, and R. R. Boyce. Evolutionary algorithm shape optimization of a hypersonic flight experiment nose cone. *Journal of Spacecraft and Rockets*, 45(3):428–437, 2008.

[36] A. DeMiguel. *Two decomposition algorithms for nonconvex optimization problems with global variables*. PhD thesis, Stanford University, Stanford, California, USA, 2001.

[37] A. DeMiguel and W. Murray. An analysis of collaborative optimization methods. In *Proceedings of the 8th AIAA/USAF/NASA/ISSMO symposium on multidisciplinary analysis and optimization, Long Beach, California, USA.*, 2000.

[38] J. DeYoung. Minimization theory of induced drag subject to constraint conditions. NASA CR3140, 1979.

[39] A. Díaz-Manríquez, G. Toscano, J. H. Barron-Zambrano, and E. Tello-Leal. A review of surrogate assisted multiobjective evolutionary algorithms. *Computational Intelligence and Neuroscience*, 2016(19):14, 2016.

[40] I. Doerfler and O. Baumann. Learning from a drastic failure: The case of the airbus a380 program. *Industry and Innovation*, 21(3):197–214, 2014.

[41] H. El-Bibany and B. C. Paulson. Concord: A framework for design, management and coordination in a collaborative AEC environment. Technical Report 62, Stanford University, Stanford, California, USA, 1992.

[42] A. Forrester and A. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1):50–79, 2009.

[43] A. Forrester, A. Sobester, and A. Keane. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[44] D. Gates. Boeing celebrates 787 delivery as program's costs top $32 billion. *The Seattle Times*, 2011, retrieved 16/09/2018, from https://www.seattletimes.com/business/boeing-celebrates-787-delivery-as-programs-costs-top-32-billion/.

[45] A. Gazaix, F. Gallard, V. Gachelin, T. Druot, S. Grihon, V. Ambert, D. Guénot, R. Lafage, C. Vanaret, B. Pauwels, et al. Towards the industrialization of new mdo methodologies and tools for aircraft design. In *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Denver, Colorado, USA*, page 3149, 2017.

[46] F. Glover. Tabu search - part I. *Operations Research Society of America Journal on Computing*, 1(3):190–206, 1989.

[47] F. Glover and E. Taillard. A user's guide to tabu search. *Annals of Operations Research*, 41(1):1–28, 1993.

[48] T. Goel, R. Vaidyanathan, R. T. Haftka, W. Shyy, N. V. Queipo, and K. Tucker. Response surface approximation of pareto optimal front in multi-objective optimization. *Computer Methods in Applied Mechanics and Engineering*, 196(4-6):879–893, 2007.

[49] S. Grihon, L. Krog, A. Tucker, and K. Hertel. A380 weight savings using numerical structural optimization. In *20th AAAF colloquium on material for aerospace applications, Paris, France*, pages 763–766, 2004.

[50] S. Gudmundsson. *General aviation aircraft design: applied methods and procedures*. Butterworth-Heinemann, 2013.

[51] R. T. Haftka. Simultaneous analysis and design. *AIAA journal*, 23(7):1099–1103, 1985.

[52] R. T. Haftka and L. T. Watson. Multidisciplinary design optimization with quasiseparable subsystems. *Optimization and Engineering*, 6(1):9–20, 2005.

[53] S. Hannapel, N. Vlahopoulos, and D. Singer. Including principles of set-based design in multidisciplinary design optimization. In *12th AIAA Aviation Technology, Integration and Operations (ATIO) Conference, Indianapolis, Indiana, USA*, 2012.

[54] S. E. Hannapel. Development of multidisciplinary design optimization algorithms for ship design under uncertainty. *University of Michigan, Ann Arbor, Michigan, USA*, 2012.

[55] C. Heath and J. Gray. Openmdao: Framework for flexible multidisciplinary design, analysis and optimization methods. In *Proceedings of the 53rd AIAA Structures, Structural Dynamics and Materials Conference, Honolulu, Hawaii, USA*, 2012.

[56] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[57] R. Hooke and T. A. Jeeves. "direct search" solution of numerical and statistical problems. *Journal of the Association for Computing Machinery*, 8(2):212–229, 1961.

[58] H. C. Howard and D. R. Rehak. Kadbase: interfacing expert systems with databases. *IEEE Intelligent Systems*, 4(3):65–76, 1989.

[59] J. Hwang, G. Kenway, and J. R. R. A. Martins. Geometry and structural modeling for high-fidelity aircraft conceptual design optimization. In *15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Atlanta, Georgia, USA*, page 2041, 2014.

[60] A. Isaacs, T. Ray, and W. Smith. An evolutionary algorithm with spatially distributed surrogates for multiobjective optimization. In M. Randall, H. A. Abbass, and J. Wiles, editors, *Progress in Artificial Life*, pages 257–268, Berlin, Heidelberg, 2007.

[61] D. 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.

[62] A. Jameson. Re-engineering the design process through computation. *Journal of Aircraft*, 36(1):36–50, 1999.

[63] N. Jelev, A. Keane, and C. Holden. A pattern search algorithm for blackboard based multidisciplinary design optimisation frameworks. *Journal of Aircraft*, 0(0):1–16, 2018.

[64] N. Jelev, A. Keane, C. Holden, and A. Sóbester. Rule based architecture for collaborative multidisciplinary aircraft design optimisation. *International Journal of Aerospace and Mechanical Engineering*, 11(5):1006–1015, 2017.

[65] Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(1):3–12, 2005.

[66] Y. Jin and M. Geslin. A study of argumentation-based negotiation in collaborative design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 24(1):35–48, 2010.

[67] Y. Jin and B. Sendhoff. Reducing fitness evaluations using clustering techniques and neural network ensembles. In K. Deb, editor, *Genetic and Evolutionary Computation – GECCO 2004, Seattle, Washington, USA*, pages 688–699, Berlin, Heidelberg, 2004.

[68] D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21(4):345–383, 2001.

[69] R. T. Jones. The spanwise distribution of lift for minimum induced drag of wings having a given lift and a given bending moment. NACA Technical Report 2249, 1950.

[70] S. Jun, Y. Jeon, J. Rho, and D. Lee. Application of collaborative optimization using genetic algorithm and response surface method to an aircraft wing design. *Journal of Mechanical Science and Technology*, 20(1):133, 2006.

[71] N. Kang, M. Kokkolaras, and P. Y. Papalambros. Solving multiobjective optimization problems using quasi-separable mdo formulations and analytical target cascading. *Structural and Multidisciplinary Optimization*, 50(5):849–859, 2014.

[72] N. Kang, M. Kokkolaras, P. Y. Papalambros, S. Yoo, W. Na, J. Park, and D. Featherman. Optimal design of commercial vehicle systems using analytical target cascading. *Structural and Multidisciplinary Optimization*, 50(6):1103–1114, 2014.

[73] A. Keane and P. Nair. *Computational approaches for aerospace design: the pursuit of excellence*. John Wiley & Sons, 2005.

[74] A. Keane and J. Scanlan. Design search and optimization in aerospace engineering. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 365(1859):2501–2529, 2007.

[75] A. Keane, A. Sóbester, and J. Scanlan. *Small unmanned fixed-wing aircraft design: a practical approach*. John Wiley & Sons, 2017.

[76] G. K. Kenway and J. R. R. A. Martins. Multipoint high-fidelity aerostructural optimization of a transport aircraft configuration. *Journal of Aircraft*, 51(1):144–160, 2014.

[77] H. Kim, S. Ragon, G. Soremekun, B. Malone, and J. Sobieszczanski-Sobieski. Flexible approximation model approach for bi-level integrated system synthesis. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York, USA*, page 4545, 2004.

[78] H. M. Kim, W. Chen, and M. M. Wiecek. Lagrangian coordination for enhancing the convergence of analytical target cascading. *AIAA Journal*, 44(10):2197–2207, 2006.

[79] H. M. Kim, N. F. Michelena, P. Y. Papalambros, and T. Jiang. Target cascading in optimal system design. *Journal of Mechanical Design*, 125(3):474–480, 2003.

[80] H. M. Kim, D. G. Rideout, P. Y. Papalambros, and J. L. Stein. Analytical target cascading in automotive vehicle design. *Journal of Mechanical Design*, 125(3):481–489, 2003.

[81] S. Kodiyalam. Evaluation of methods for multidisciplinary design optimization (MDO) part 1. NASA CR-1998-208716, 1998.

[82] S. Kodiyalam and J. Sobieszczanski-Sobieski. Bilevel integrated system synthesis with response surfaces. *AIAA Journal*, 38(8):1479–1485, 2000.

[83] S. Kodiyalam, C. Yuan, and J. Sobieski. Evaluation of methods for multidisciplinary design optimization (mdo). part 2. Technical report, NASA CR-2000-210313, 2000.

[84] R. M. Kolonay and M. H. Kobayashi. Optimization of aircraft lifting surfaces using a cellular division method. *Journal of Aircraft*, 52(6):2051–2063, 2015.

[85] L. Krog, A. Tucker, and G. Rollema. Application of topology, sizing and shape optimization methods to optimal design of aircraft components. In *Procceedings of 3rd Altair UK HyperWorks Users Conference*, 2002.

[86] I. Kroo, S. Altus, R. Braun, P. Gage, and I. Sobieski. Multidisciplinary optimization methods for aircraft preliminary design. In *5th Symposium on Multidisciplinary Analysis and Optimization, Panama City Beach, Florida, USA*, volume 4325, page 1994, 1994.

[87] I. Kroo and V. Manning. Collaborative optimization-status and directions. In *8th Symposium on Multidisciplinary Analysis and Optimization, Long Beach, California, USA*, page 4721, 2000.

[88] A. B. Lambe and J. R. R. A. Martins. Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes. *Structural and Multidisciplinary Optimization*, 46(2):273–284, 2012.

[89] T. Legget. A350: The aircraft that airbus did not want to build. *BBC News*, 2013, retrieved 16/09/2018, from https://www.bbc.co.uk/news/business-22803218.

[90] A. S. Lewis and M. L. Overton. Nonsmooth optimization via quasi-newton methods. *Mathematical Programming*, 141(1):135–163, 2013.

[91] K. Lewis and F. Mistree. Modeling interactions in multidisciplinary design: A game theoretic approach. *AIAA Journal*, 35(8):1387–1392, 1997.

[92] R. M. Lewis, V. Torczon, and M. W. Trosset. Direct search methods: then and now. *Journal of Computational and Applied Mathematics*, 124(1):191–207, 2000.

[93] H. Liu, W. Chen, M. Kokkolaras, P. Y. Papalambros, and H. M. Kim. Probabilistic analytical target cascading: a moment matching formulation for multilevel optimization under uncertainty. *Journal of Mechanical Design*, 128(4):991–1000, 2006.

[94] M. A. Lobbia. Multidisciplinary design optimization of waverider-derived crew reentry vehicles. *Journal of Spacecraft and Rockets*, 54(1):233–245, 2016.

[95] H. Lu and W. Chen. Dynamic-objective particle swarm optimization for constrained optimization problems. *Journal of combinatorial optimization*, 12(4):409–419, 2006.

[96] K. Madhavan, D. Shahan, C. C. Seepersad, D. A. Hlavinka, and W. Benson. An industrial trial of a set-based approach to collaborative design. In *Proceedings of*

ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Brooklyn, New York, USA, 2008.

[97] J. R. R. A. Martins. *A coupled-adjoint method for high-fidelity aero-structural optimization.* PhD thesis, Stanford University, Stanford, California, USA, 2002.

[98] J. R. R. A. Martins and A. B. Lambe. Multidisciplinary design optimization: A survey of architectures. *AIAA Journal*, 51(9):2049–2075, 2013.

[99] J. R. R. A. Martins, C. Marriage, and N. Tedford. pymdo: an object-oriented framework for multidisciplinary design optimization. *ACM Transactions on Mathematical Software (TOMS)*, 36(4):20, 2009.

[100] C. D. McAllister, T. W. Simpson, K. Hacker, K. Lewis, and A. Messac. Integrating linear physical programming within collaborative optimization for multiobjective multidisciplinary design optimization. *Structural and Multidisciplinary Optimization*, 29(3):178–189, 2005.

[101] T. McGeer. Wing design for minimum drag with practical constraints. *Journal of Aircraft*, 21(11):879–886, 1984.

[102] E. Mezura-Montes and C. A. C. Coello. Constraint-handling in nature-inspired numerical optimization: past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.

[103] N. Michelena, H. M. Kim, and P. Papalambros. A system partitioning and optimization approach to target cascading. In *Proceedings of the 12th International Conference on Engineering Design, Munich, Germany*, volume 2, pages 1109–1112, 1999.

[104] J. D. Mulder, J. J. Van Wijk, and R. Van Liere. A survey of computational steering environments. *Future Generation Computer Systems*, 15(1):119–129, 1999.

[105] R. Nazzeri, M. Haupt, F. Lange, and C. Sebastien. *Selection of Critical Load Cases Using an Artificial Neural Network Approach for Reserve Factor Estimation.* Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth eV, 2015.

[106] L. M. Nicolai and G. Carichner. *Fundamentals of Aircraft and Airship Design: Volume 1, Aircraft Design.* AIAA 2010.

[107] H. P. Nii. The blackboard model of problem solving and the evolution of blackboard architectures. *AI Magazine*, 7(2):38, 1986.

[108] J. Nocedal and S. Wright. *Numerical Optimization.* Springer-Verlag New York, 2006.

[109] G. Norris and M. Wagner. *Airbus A380: superjumbo of the 21st century.* Zenith Imprint, 2005.

[110] J. Ollar, V. Toropov, and R. Jones. Sub-space approximations for mdo problems with disparate disciplinary variable dependence. *Structural and Multidisciplinary Optimization*, 55(1):279–288, 2017.

[111] Y. Ong, A. Keane, and P. B. Nair. Surrogate-assisted co-evolutionary search. In *At Proceedings of the 4th Asia Pacific Conference on Simulated Evolution and Learning, Singapore*, pages 1140-1145. IEEE, 2002.

[112] Y. S. Ong, P. B. Nair, A. Keane, and K. W. Wong. *Surrogate-Assisted Evolutionary Optimization Frameworks for High-Fidelity Engineering Design Problems*, Studies in Fuzziness and Soft Computing, pages 307–331. Springer Berlin Heidelberg, 2005.

[113] S. L. Padula, N. Alexandrov, and L. L. Green. Mdo test suite at nasa langley research center. In *6th AIAA/SASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Bellevue, Washington, USA*, pages 410–420, 1996.

[114] D. J. Pate, M. D. Patterson, and B. J. German. Optimizing families of reconfigurable aircraft for multiple missions. *Journal of Aircraft*, 49(6):1988–2000, 2012.

[115] R. E. Perez, H. H. Liu, and K. Behdinan. Evaluation of multidisciplinary optimization approaches for aircraft conceptual design. In *AIAA/ISSMO multidisciplinary analysis and optimization conference, Albany, New York, USA*, 2004.

[116] P. Piperni, M. Abdo, F. Kafyeke, and A. T. Isikveren. Preliminary aerostructural optimization of a large business jet. *Journal of Aircraft*, 44(5):1422–1438, 2007.

[117] D. J. Poole, C. B. Allen, and T. C. Rendall. A generic framework for handling constraints with agent-based optimization algorithms and application to aerodynamic design. *Optimization and Engineering*, 18(3):659–691, 2017.

[118] N. V. Praet. Bombardier hands control of c series airliner to airbus, October 2017, retrieved 16/09/2018, from https://www.theglobeandmail.com/report-on-business/bombardier-sells-majority-stake-in-c-series-to-airbus/article36610340/.

[119] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, et al. *Numerical recipes The Art of Scientific Computing.* Cambridge University Press, Cambridge, 3rd edition, 1986.

[120] A. R. Price, A. Keane, and C. M. Holden. On the coordination of multidisciplinary design optimization using expert systems. *AIAA Journal*, 49(8):1778–1794, 2011.

[121] A. O. Pugachev, A. V. Sheremetyev, V. V. Tykhomirov, and O. I. Shpilenko. Structural dynamics optimization of rotor systems for a small-size turboprop engine. *Journal of Propulsion and Power*, 31(4):1083–1093, 2015.

[122] G. T. Pulido and C. A. Coello Coello. Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer. In *Proceedings of Genetic and Evolutionary Computation Conference GECCO 2004, Seattle, Washington, USA*, pages 225–237, 2004.

[123] T. Qu, G. Q. Huang, V.-D. Cung, and F. Mangione. Optimal configuration of assembly supply chains using analytical target cascading. *International Journal of Production Research*, 48(23):6883–6907, 2010.

[124] S. S. Rao. *Engineering Optimization: Theory and Practice.* John Wiley & Sons, 2009.

[125] D. P. Raymer. *Aircraft Design: A Conceptual Approach.* American Institute of Aeronautics and Astronautics, 2006.

[126] G. Rennen. Subset selection from large datasets for kriging modeling. *Structural and Multidisciplinary Optimization*, 38(6):545, 2009.

[127] B. Roth and I. Kroo. Enhanced collaborative optimization: Application to an analytic test problem and aircraft design, (2008). In *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, Canada*, 2008.

[128] B. D. Roth. *Aircraft family design using enhanced collaborative optimization.* PhD thesis, Stanford University, Stanford, California, USA, 2008.

[129] B. D. Roth and I. M. Kroo. Enhanced collaborative optimization: a decomposition-based method for multidisciplinary design. In *Proceedings of the ASME design engineering technical conferences, Brooklyn, New York, USA*, pages 3–6, 2008.

[130] K. Sabbagh. *Twenty First Century Jet: Making and Marketing the Boeing 777.* Scribner Book Company, 1996.

[131] M. Sadoff. Pitch-up problem: A criterion and method of evaluation. NASA-MEMO-3-7-59A, 1959.

[132] O. Schrenk. A simple approximation method for obtaining the spanwise lift distribution. *The Aeronautical Journal*, 45(370):331–336, 1941.

[133] R. Sellar, S. Batill, and J. Renaud. Response surface based, concurrent subspace optimization for multidisciplinary system design. In *34th Aerospace Sciences Meeting and Exhibit, Reno, Nevada, USA*, number 96-0714, 1996.

[134] D. Shahan and C. C. Seepersad. Bayesian networks for set-based collaborative design. In *Proceedings of the ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, San Diego, California, USA*, American Society of Mechanical Engineers, pages 303–313, 2009.

[135] D. W. Shahan and C. C. Seepersad. Bayesian network classifiers for set-based collaborative design. *Journal of Mechanical Design*, 134(7):071001, 2012.

[136] L. Shi and K. Rasheed. A survey of fitness approximation methods applied in evolutionary algorithms. In *Computational intelligence in expensive optimization problems*, pages 3–28. Springer, 2010.

[137] M.-K. Shin and G.-J. Park. Multidisciplinary design optimization based on independent subspaces. *International Journal for Numerical Methods in Engineering*, 64(5):599–617, 2005.

[138] J. N. Siddall. *Optimal engineering design: principles and applications*. CRC Press, 1982.

[139] I. Sobieski, V. Manning, and I. Kroo. Response surface estimation and refinement in collaborative optimization. In *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, Missouri, USA*, page 4753, 1998.

[140] I. P. Sobieski and I. M. Kroo. Collaborative optimization using response surface estimation. *AIAA Journal*, 38(10):1931–1938, 2000.

[141] J. Sobieszczanski-Sobieski. Optimization by decomposition: a step from hierarchic to non-hierarchic systems. *Recent Advances in Multidisciplinary Analysis and Optimization*, page 51, 1988.

[142] J. Sobieszczanski-Sobieski, J. S. Agte, and R. R. Sandusky. Bilevel integrated system synthesis. *AIAA Journal*, 38(1):164–172, 1998.

[143] N. Srinivasan and K. Deb. Multi-objective function optimisation using non-dominated sorting genetic algorithm. *Evolutionary Computation*, 2(3):221–248, 1994.

[144] A. Srivastava, K. Hacker, K. Lewis, and T. Simpson. A method for using legacy data for metamodel-based design of large-scale systems. *Structural and Multidisciplinary Optimization*, 28(2-3):146–155, 2004.

[145] R. Tappeta and J. Renaud. Multiobjective collaborative optimization. *Journal of Mechanical Design*, 119(3):403–411, 1997.

[146] N. P. Tedford and J. R. R. A. Martins. Benchmarking multidisciplinary design optimization algorithms. *Optimization and Engineering*, 11(1):159–183, 2010.

[147] E. Torenbeek. *Synthesis of subsonic airplane design*. Springer Science & Business Media, 1982.

[148] V. Toropov and L. Alvarez. Development of mars–multipoint approximation method based on the response surface fitting. *AIAA Journal*, 98:4769, 1998.

[149] V. Toropov, A. Filatov, and A. Polynkin. Multiparameter structural optimization using fem and multipoint explicit approximations. *Structural Optimization*, 6(1):7–14, 1993.

[150] S. Tosserams, L. Etman, P. Papalambros, and J. Rooda. An augmented lagrangian relaxation for analytical target cascading using the alternating direction method of multipliers. *Structural and Multidisciplinary Optimization*, 31(3):176–189, 2006.

[151] S. Tosserams, L. Etman, and J. Rooda. Augmented lagrangian coordination for distributed optimal design in mdo. *International Journal for Numerical Methods in Engineering*, 73(13):1885–1910, 2008.

[152] S. Tosserams, M. Kokkolaras, L. Etman, and J. Rooda. A nonhierarchical formulation of analytical target cascading. *Journal of Mechanical Design*, 132(5):051002, 2010.

[153] Unified Teaching Staff. Aeronautics and astronautics: Wing structural analysis and bending test (s/l10). *MIT OpenCourseWare*, 2005, retrieved 16/09/2018, https://ocw.mit.edu/courses/aeronautics-and-astronautics/16-01-unified-engineering-i-ii-iii-iv-fall-2005-spring-2006/systems-labs-06/spl10.pdf.

[154] S. Vázquez, M. J. Martín, B. B. Fraguela, A. Gómez, A. Rodriguez, and B. Elvarsson. Novel parallelization of simulated annealing and hooke & jeeves search algorithms for multicore systems with application to complex fisheries stock assessment models. *Journal of Computational Science*, 17:599–608, 2016.

[155] B. Walther and S. Nadarajah. Adjoint-based constrained aerodynamic shape optimization for multistage turbomachines. *Journal of Propulsion and Power*, 31(5):1298–1319, 2015.

[156] A. Ward, J. K. Liker, J. J. Cristiano, and D. K. Sobek. The second toyota paradox: How delaying decisions can make better cars faster. *Sloan Management Review*, 36(3):43, 1995.

[157] B. J. Wielinga, A. T. Schreiber, and J. A. Breuker. Kads: A modelling approach to knowledge engineering. *Knowledge Acquisition*, 4(1):5–53, 1992.

[158] E. Winer and C. Bloebaum. Development of visual design steering as an aid in large-scale multidisciplinary design optimization. part i: method development. *Structural and Multidisciplinary Optimization*, 23(6):412–424, 2002.

[159] E. Winer and C. Bloebaum. Development of visual design steering as an aid in large-scale multidisciplinary design optimization. part ii: method validation. *Structural and Multidisciplinary Optimization*, 23(6):425–435, 2002.

[160] B. Wujek and J. Renaud. New adaptive move-limit management strategy for approximate optimization, part 1. *AIAA Journal*, 36(10):1911–1921, 1998.

[161] B. Wujek and J. Renaud. New adaptive move-limit management strategy for approximate optimization, part 2. *AIAA Journal*, 36(10):1922–1934, 1998.

[162] M. Xiao, L. Gao, H. B. Qiu, X. Y. Shao, and X. Z. Chu. An approach based on enhanced collaborative optimization and kriging approximation in multidisciplinary design optimization. *Advance Material Research*, 118:399–403, 2010.

[163] M. Xiao, L. Gao, X. Shao, H. Qiu, and P. Jiang. A generalised collaborative optimisation method and its combination with kriging metamodels for engineering design. *Journal of Engineering Design*, 23(5):379–399, 2012.

[164] M. Xiao, H. Qiu, L. Gao, X. Shao, and X. Chu. An enhanced collaborative optimization methodology for multidisciplinary design optimization. In *2010 International Conference on Mechanical, Industrial, and Manufacturing Technologies, Sanya, China*, 2010.

[165] M. Xiao, X. Shao, L. Gao, and Z. Luo. A new methodology for multi-objective multidisciplinary design optimization problems based on game theory. *Expert Systems with Applications*, 42(3):1602–1612, 2015.

[166] H. Xu, C.-H. Chuang, and R.-J. Yang. Improving multiobjective multidisciplinary optimization with a data mining-based hybrid method. In *ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Boston, Massachusetts, USA*, 2015.

[167] M. C. Yang and Y. Jin. An examination of team effectiveness in distributed and co-located engineering teams. *International Journal of Engineering Education*, 24(2):400–408, 2008.

[168] S.-I. Yi, J.-K. Shin, and G. Park. Comparison of mdo methods with mathematical examples. *Structural and Multidisciplinary Optimization*, 35(5):391–402, 2008.