JOURNAL OF AIRCRAFT
Vol. M, No. N

A Pattern Search Algorithm for Blackboard Based Multidisciplinary Design Optimisation Frameworks

Nickolay Jelev [*] and Andy Keane[*]
University of Southampton
and
Carren Holden[†]
Airbus UK.

Preliminary aircraft design necessitates the use of a range of analysis tools, which are often scattered among many departments in an organisation and require regular tuning from skilled operators. For this reason, a distributed Multidisciplinary Design Optimisation approach that permits individual organisational domains to use their preferred analysis and optimisation tools would be most suitable. This paper revisits a Blackboard framework, which uses simple heuristics to automatically guide organisational design domains to a single optimum by narrowing the bounds on the shared design variables. The authors present a newly developed rule base for this legacy framework, which has been given the title "Multidisciplinary Pattern Search".
Two examples, one of which is for conceptual transonic wing design, demonstrate the merit of the newly developed rule base, database and visualisation modules. They also serve as a means for comparisons with two established Multidisciplinary Design Optimisation architectures. The results indicate that the Blackboard performs better than the distributed Collaborative Optimisation approach, albeit worse than the monolithic Simultaneous Analysis and Design method that tends to be organisationally disruptive to implement.

Nomenclature

a_n	= Weighting for objective function
f_0	= Global objective function
$f_{1,2,...k}$	= Domain local objective functions
lb_i	= Variable lower bound
$\mathbf{lb}$	= Vector of current lower bounds
$\mathbf{lb_{init}}$	= Vector of starting lower bounds
ub_i	= Variable upper bound
$\mathbf{ub}$	= Vector of current upper bounds
$\mathbf{ub_{init}}$	= Vector of starting upper bounds
$\epsilon_{\mathbf{c}}$	= Current convergence factor
$\epsilon_{\mathbf{c_{init}}}$	= Starting value for the convergence factor
$\epsilon_{\mathbf{c_{min}}}$	= Desired final convergence factor
$\epsilon_{\mathbf{cb}}$	= Bound contraction factor
$\epsilon_{\mathbf{mb}}$	= Bound movement factor
$\epsilon_{\mathbf{pat}}$	= Bound movement factor during pattern move

I. Introduction

PRELIMINARY aircraft design problems tend to be split into a number of disciplines, subsystems or domains for organisational reasons. They require a Multidisciplinary Design Optimisation (MDO) architecture to extract a truly multidisciplinary optimal design, rather than one that simply satisfies constraints and performance targets. There exist many methods to solve such problems and they mainly fall into two categories: monolithic or distributed. Monolithic architectures combine all discipline analyses under a single optimiser, which controls the entire design process. Because these architectures are generally able to solve complex problems in fewer analysis evaluations than distributed approaches [1,2], their application in certain aspects of industrial design has seen an increase [3].

There remain a plethora of problems however 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, which are often spread across multiple divisions in the organisation and necessitate regular input from skilled design engineers. In these cases, monolithic architectures would be unsuitable because they can become difficult to operate and maintain. This is especially the case when the analysis programs undergo regular updates [4] 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.

To make MDO applicable to organisationally dispersed problems, academics have developed multi-level (or distributed) MDO architectures. Simply put these permit low level domains to design and optimise an aspect of an aircraft 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. In theory these concepts work well, but have proven difficult to implement in a practical design environment. Which is why, as far as the authors are aware, none of the dozen or so established distributed architectures (and their subsequent modifications) from the literature have seen any extensive use in industry. In fact, the most recent survey on MDO concluded that further work is needed in this area [5], because many of the well-established distributed architectures share the following generic set of drawbacks: slow convergence, complicated formulations, limited real world testing, for some no guarantee of convergence and perhaps most importantly the need for severe organisational restructuring [3].

Current state of the art methods tend to address the practical challenges of implementation by retaining the design processes used by engineering teams. Methods compatible with the industrial settings are more likely to be accepted and applied in industry because they will have lower risks to implementation. One recent example is a version of the Bi-Level Integrated System Synthesis currently under development at IRT Saint Exupery [6]. A top level optimiser controls the shared design variables and domain level optimisers set the values for the local variables. Over successive iterations between the two levels, Gazaix et al. [6] 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.

Previously, other authors have used a similar concept to improve the practicality of their proposed MDO methods. Shahan and Seepersad [7] used a Bayesian Network to capture knowledge from individual domain analyses. The shared design variables were mapped on probability distributions to enable a chief engineer to select designs that

[*]Nickolay Jelev and Prof. Andy Keane are part of the Computational Engineering and Design Research Group, within the Faculty of Engineering and the Environment at the University of Southampton, UK.

[†]Dr. Carren Holden is associated with Airbus Operations Ltd., Pegasus House, Aerospace Avenue, Filton, Bristol BS34 7PA, United Kingdom

meet the preferences of multiple domains. Lewis and Mistree [8] and more recently Xiao et al. [9] demonstrated the merit of Game Theory approaches on multi-objective MDO problems, while others have demonstrated various bound reduction strategies [2, 10–12].

The work in this paper presents a solution that overcomes the limited practicality of distributed MDO, but by combining a legacy Blackboard framework by Price et al. [2] with a newly developed rule base. The rule base, which we have called here the *Multidisciplinary Pattern Search* (MDPS), uses relatively simple if-else-then rules to reduce the separation between the bounds on the shared design variables. Its development and assessment, along with a way of visualising convergence, are the main contributions of this paper. Two sections discussing several pattern search algorithms and an explanation of Multidisciplinary Pattern Search follow after a review of the legacy Blackboard framework. Section IV describes the newly developed Multidisciplinary Pattern Search and Section V validates the approach over two examples. Finally, Sections VI and VII discuss the findings, provide directions for future work and conclude the paper. The Appendix summarises the main analyses functions used in the Unmanned Aerial Vehicle (UAV) wing design problem.

II. The Legacy Blackboard Framework

Blackboard frameworks have been used in the past as a way to address communication problems in collaborative engineering projects [13–15]. Price et al. [2] adapted the generic Blackboard model for MDO problems, by proposing a framework that iteratively reduces the bound spacing on the shared design variables. The framework enables organisational domains to work concurrently and in almost complete isolation from one another, while a rule base within the Blackboard guides the process to a single optimal design. The process is illustrated in Figure 1 and can be described by the following steps:

Phase 1: Chief engineers select a common starting design and define 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 in relative isolation from all other teams.

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 a domain requires coupling information from another, they can simply query the database and obtain it from there.

Phase 4: At the end of the local search (or at a pre-set design gate), each domain highlights their preferred solution in the database.

Phase 5: A rule base then evaluates the preferred solutions and proposes new upper/lower bounds, effectively setting a new search space to the domains for the following iteration. Subsequently, the process is repeated from *Phase 2* until the available search region becomes small enough to be deemed to represent a single design.

From an MDO perspective, this method avoids the need for a specific problem decomposition or internal domain restructuring, which are drawbacks of many already established MDO architectures [3]. This makes the Blackboard framework suitable for problems in the early preliminary design stage where a desired concept is already selected and parametrised, but key geometric variables are not yet fixed, just enclosed by upper and lower bounds. Simply put, the method can be viewed as a distributed MDO architecture where low level domains optimise their specific engineering objectives (such as drag, mass, cost, etc..) and a system level Blackboard coordinates the process. Therefore in theory, a transition from the current sequential design process to the proposed concurrent MDO method should be easy because the internal process within the design teams remains the same, while already existing data communication, storage and management protocols can be easily adapted to suit the Blackboard.

To understand why the Blackboard is attractive from a practical perspective, let's consider a scenario where conflicting designs are proposed by different organisational domains. Current industrial practices deal with these conflicts via regular, usually face to face meetings of chief engineers representing each organisational domain. Representatives then attempt to sway decision makers in their favour by presenting engineering evidence to support their design solution [16]. The final decision is ultimately up to certain engineers that can be influenced by a number of sociological factors [17, 18]. The Blackboard introduces a formal procedure for selecting competing designs, which not only helps to avoid these drawbacks, but also makes a digital environment available for use by geographically separated design teams.

While the original framework (pictured in Figure 1) is sound, it made limited use of data mining, lacked a user interface and can be shown to fail on problems outside those tested in the original publication [19]. So here we propose to use the Multidisciplinary Patten Search instead, which has been explicitly developed to make better use of previously evaluated solutions to speed up convergence. An additional visualisation module is also introduced to help users interact with the process. This short description of the Blackboard search method contains sufficient information to now proceed with a review of the legacy pattern search optimisers that underpin the Multidisciplinary Pattern Search logic.

III. A Survey of Heuristic Optimisers

The Multidisciplinary Pattern Search shares a number of concepts with two existing heuristic optimisation algorithms - The "Hooke and Jeeves" and "Tabu" searches [20, 21]. Here we introduce these two optimisers to help describe the Multidisciplinary Pattern Search later in Section IV. Both the Hooke and Jeeves, and Tabu searches share a common set of moves collectively called the *neighbourhood search*. Given a problem with several optimisation variables, a *neighbourhood search* begins with an initial evaluation of the objective function at a set starting point. New trial designs are selected by increasing and decreasing the magnitude of all the variables one by one. If an improvement on the starting point is observed, the design with the highest improvement is kept and set as the new base point. Otherwise the moves are rejected and the search continues to the next variable. If all moves fail during a neighbourhood search, it is repeated again, but this time with a smaller step magnitude. This continues until a predetermined step size is reached and the algorithm is deemed to have converged.

While this is an effective local search method, it can be very inefficient. Several advancements have improved the convergence of the *neighbourhood search* by exploiting the history of previously evaluated solutions. Bell and Pike [22] proposed that the successful direction from previous iterations in the neighbourhood search should be remembered and exploited. Hence rather than evaluating both upper and lower steps each time, the direction that resulted in a successful move during the previous search should be evaluated first and kept if found to improve the objective. This logic is effectively trying to sidestep moves in the opposite direction that are unlikely to produce an improvement.

In 1961 Hooke and Jeeves [20] proposed an additional *pattern move* to the neighbourhood search. The pattern move stores the magnitude and direction of the successful moves and applies a single move across all variables that yielded an improvement during the neighbourhood search. The move accumulates over successive iterations and is always applied at the end of a neighbourhood search. It acts as a short-cut bypassing what would otherwise be several neighbourhood searches in the same direction.

The author of the Tabu search [21], and others after him [23], recognised that the Hooke and Jeeves method is prone to re-evaluating designs that have already been queried. This could be easily avoided by storing all moves in a database, which could be checked before each new evaluation for duplicates. In our description here we call this rule a "duplicate check", even though this is called a "Tabu move" in the scheme of the Tabu search.

The Tabu search makes even more sophisticated use of memory in order to improve the speed of convergence and the likelihood of finding a global minimum. While a full description can be found in a number of references [21, 24, 25], the method can be simply described as follows. Every new design and its outcome are encoded and stored in a database. This allows the search to sometimes accept results that degrade the objective function in order to explore wider areas in the design space. In doing so, this increasing the likelihood of finding

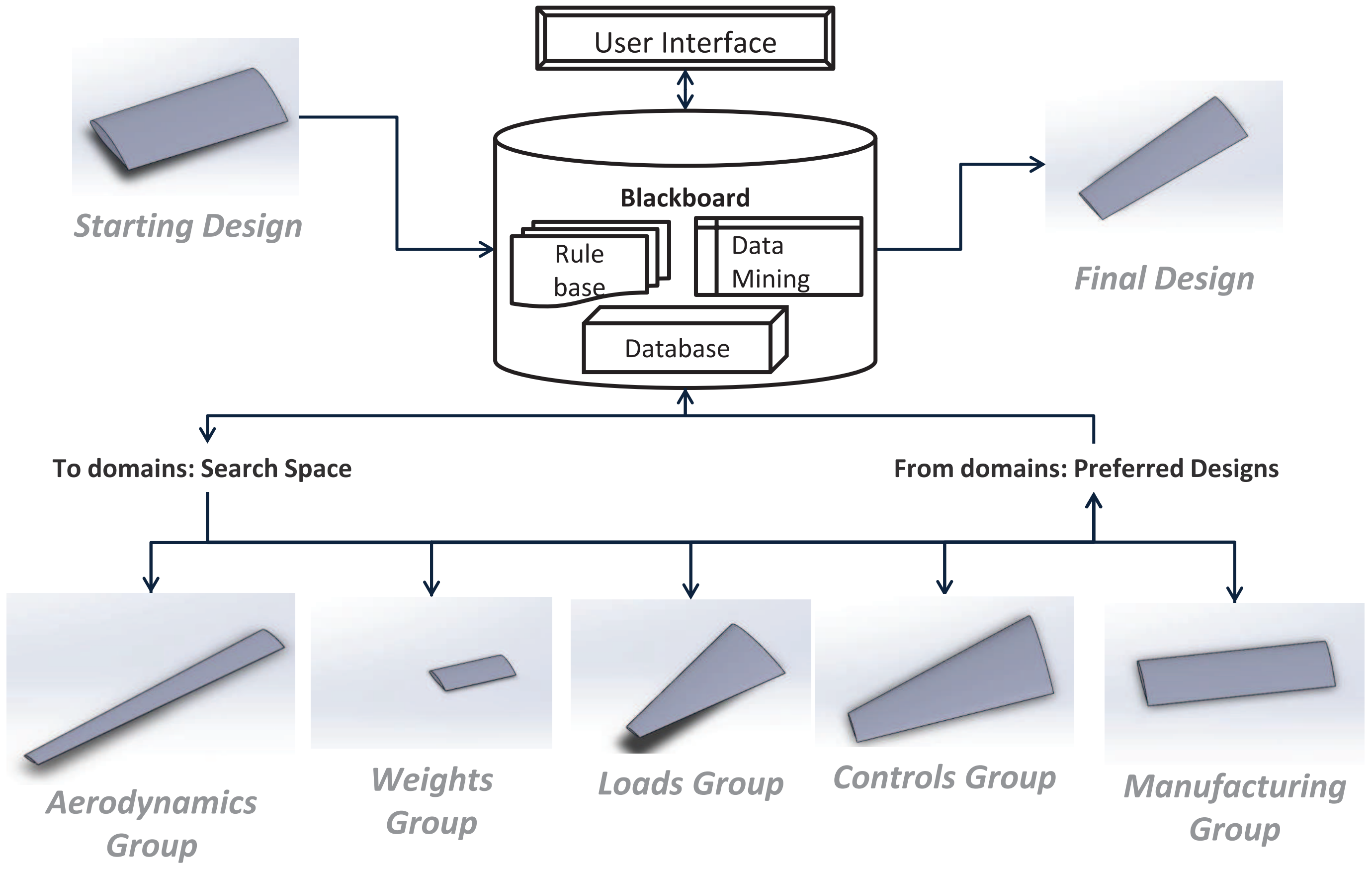


Figure 1. Blackboard framework for MDO (as adapted from Price et. al [2])

a global minimum while avoiding the risk of re-evaluating the same designs. A list of "Tabu" moves are kept in several types of memory. 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. The original method was developed for combinatorial problems, but it has been shown to work on continuous problems as well [26].

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 [27]. Their relative simplicity and robustness makes them particularly attractive for rapid in-house adaptation, which is why their logic underpins the Multidisciplinary Pattern Search.

IV. The Multidisciplinary Pattern Search

The description that follows covers the four main aspects of the Multidisciplinary Pattern Search used here - the rules, control factors, database and visualisation. While some work has been done towards domain local data mining, a separate module is still under development and will be the subject of future research work. Its features are discussed in more detail later in Section VI.

A. The Multidisciplinary Pattern Search Rules

We begin with a description of the Multidisciplinary Pattern Search, whose primary role is to automatically evaluate a number of locally optimal designs and set the search space for the following iteration. In its most basic form, it is a neighbourhood search with several additional rules that help improve the rate of convergence by exploiting the history of already evaluated solutions. We have split the algorithm into four independent sections to help explain the logic and draw parallels with the legacy optimisers described in Section III. Table 1 describes the main rules, Figure 2 shows the process flow and Equations 2 to 10 are the mathematical formulas that control the bounds and factor relaxations.

1. The Neighbourhood Search

The neighbourhood search here is equivalent to the explanation in Section III, with the only difference being that a perturbation is achieved by moving both upper and lower bounds in a given direction. At the start both bounds on each variable are moved together, one by one, 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 meets their local constraints. We define this local domain minima as a *preference*, which in the context of a real world design problem can be obtained from optimisation search, a single domain analysis or even an accurate engineering guess.

The Multidisciplinary Pattern Search then calculates a combined (global) objective from the submitted local preferences. In our formulation we assume the global objective function to be a weighted sum of the domain objectives with a penalty for any the constraint failures. This can be represented by $f_0 = \Sigma a_n Q_n + \Sigma P_n$, where a_n are individual weightings, Q_n are the multiple objectives and P_n are penalties resulting from constraint failures. In addition we have observed a goal based objective function to also work well, provided that the user defined targets are well selected. The function format in this case is $f_0 = \Sigma(Q_n - q_n)^2 + \Sigma P_n$, where q_n are user defined objective targets, while P_n and Q_n hold the same values as noted before. Strictly speaking, the global objective comprises of a set of disjointed locally optimal designs from each domain. It generates an assessment of a design region, rather than an evaluation of any single design in conventional optimisation terminology.

An iteration the multidisciplinary neighbourhood search in this context terminates when all possible bound movements have been exhausted and no improvement was observed. There can be one of two outcomes from then on. A sole pattern move follows in the event that success

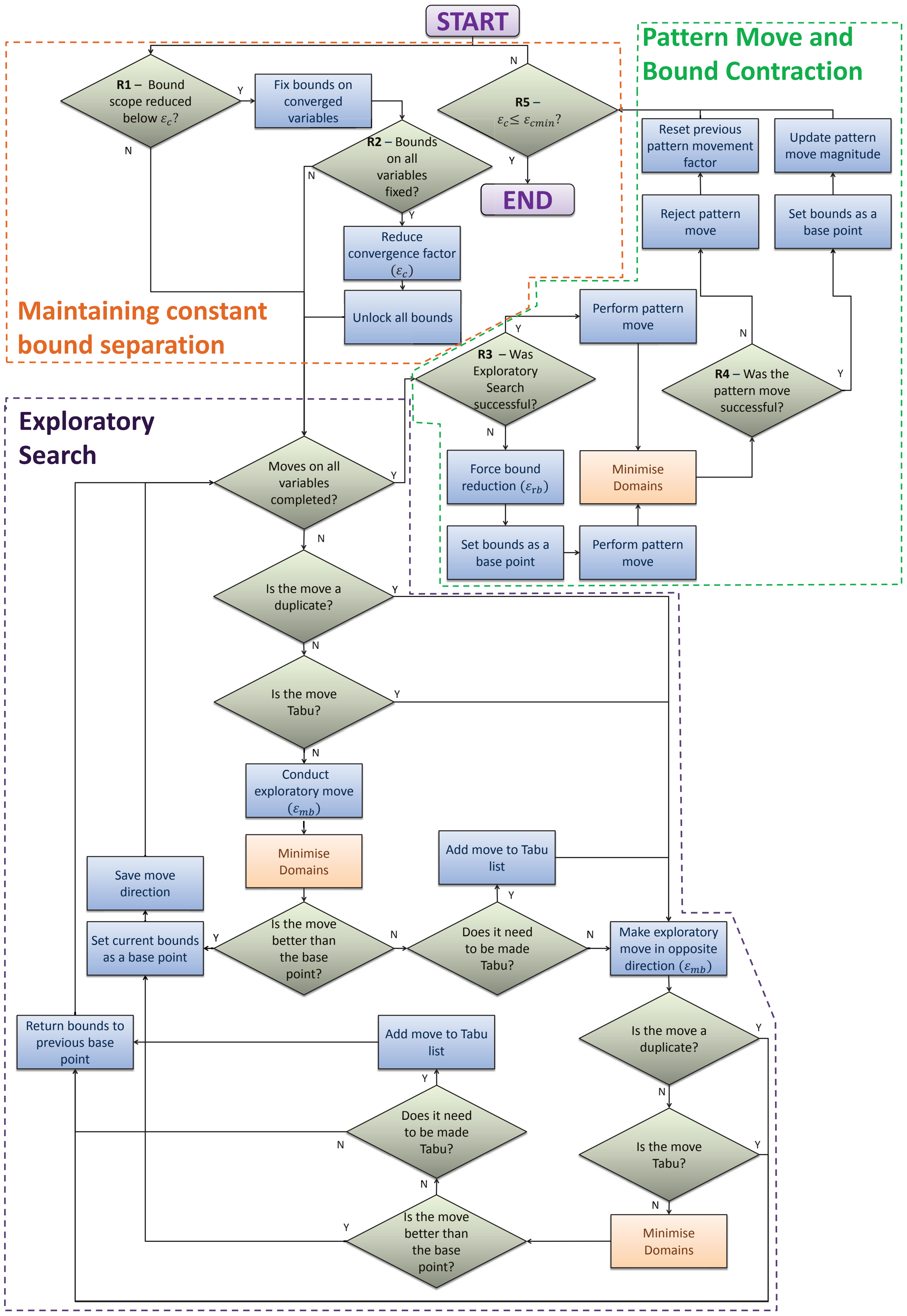

START
Pattern Move and Bound Contraction
N
R1 – Bound scope reduced below ε_c?
Y
Fix bounds on converged variables
R5 – $\varepsilon_c \leq \varepsilon_{cmin}$?
Y
END
N
R2 – Bounds on all variables fixed?
Y
Reduce convergence factor (ε_c)
Unlock all bounds
Reset previous pattern movement factor
Update pattern move magnitude
Reject pattern move
Set bounds as a base point
Maintaining constant bound separation
Perform pattern move
R3 – Was Exploratory Search successful?
R4 – Was the pattern move successful?
Exploratory Search
Moves on all variables completed?
Force bound reduction (ε_{rb})
Minimise Domains
Set bounds as a base point
Perform pattern move
Is the move a duplicate?
Is the move Tabu?
Conduct exploratory move (ε_{mb})
Minimise Domains
Add move to Tabu list
Save move direction
Set current bounds as a base point
Is the move better than the base point?
Does it need to be made Tabu?
Make exploratory move in opposite direction (ε_{mb})
Return bounds to previous base point
Add move to Tabu list
Is the move a duplicate?
Does it need to be made Tabu?
Is the move Tabu?
Is the move better than the base point?
Minimise Domains

Figure 2. Process flow for the Multidisciplinary Pattern Search

Table 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, force bound contraction for all unlocked variables, apply pattern move and proceed to **R4**.
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 and/or reduces any local constraints violations.

was observed. Otherwise a combination of both a bound contraction succeeded by a pattern move follows after an unsuccessful neighbourhood search. The bound contraction forces both bounds to reduce the searchable design region. This ensures convergence, but often causes short term deterioration in the objective function that can be corrected by the successive pattern move or exploratory search. Alone these moves can force multidisciplinary design problems to converge, but much like the single disciplinary case, they are inefficient without the additional pattern move, Tabu and duplicate checks.

2. *The Pattern Move*

The pattern move is triggered directly after a forced bound contraction or when success occurs following an exploratory search. For both cases, it is a bound movement across the variables that generated an improvement during the previous exploratory search. The initial magnitude of the pattern move equals the current bound movement factor and increases as the pattern moves are accepted. If the outcome of the pattern move is rejected, the pattern bound movement factor is reset. See rules R3 and R4 in Table 1.

3. *The Duplicate Check*

The notion of using long term memory to prevent certain moves from firing has been applied in the Multidisciplinary Pattern Search. Duplicate evaluations of the same design region were often observed after two successive neighbourhood searches. To prevent this, every design region is encoded and stored in the database for later use by the duplicate check rule. Before each new region is evaluated, the duplicate check queries the database and bypasses a move if it has already been evaluated. In addition, a form of the Bell and Pike [22] improvement has also been employed here, whereby the direction of previously successful moves is kept and exploited during the search.

4. *The Tabu Rule*

Towards the later stages of the neighbourhood search, we observed that the same bound movements break the same constraints over consecutive iterations. A rule similar to one 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 check identifies moves that caused two consecutive constraint failures from previous iterations and stores them in the database. These "Tabu" moves are then skipped 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, to our knowledge this opportunity to avoid moves that are likely to produce high constraint violations has not been implemented in a single disciplinary search algorithm. We therefore note that this simple rule can be applied 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.

5. *The Constant Bound Separation*

The combination of bound movements and contractions over successive iterations can cause the bounds on some variables to prematurely converge. This often occurs because the bounds are forced towards the upper or lower limit of a variable by the bound movement rules. Ultimately this restricts the search region, which can trap the algorithm in an infeasible search space. This was overcome by iteratively reducing the minimum bound separation threshold. The bounds on a variable are locked when the separation falls below the current threshold value. This rule prevents any further bound movements/contractions until all other variables reach that threshold. This strategy is described in more detail in Section IV.B and by R1 in Table 1.

B. Control Factors

The original Hooke and Jeeves search had one control factor - the step size. The additional complexity associated with controlling bound movements, brings about an increase in the number of user set control factors. Here we describe the five factors that can be set by the user and control the algorithm convergence ($\epsilon_{c_{init}}$, $\epsilon_{c_{min}}$, r_1, ϵ_{rb}, ϵ_{mb}). To help with the explanation, let us define the bound separation to be the difference between the upper and lower bounds. The convergence factor (ϵ_c) 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 1, where ub and lb are the current ub_{init} and lb_{init} denote the starting bounds.

$$\epsilon_c = \frac{ub - lb}{ub_{init} - lb_{init}}. \quad (1)$$

Earlier in Section IV.5, we explained why all variables need to maintain a constant bound spacing. The convergence factor (ϵ_c) acts as a threshold for maintaining a constant bound separation. In practice, this threshold is iteratively reduced by a relaxation value (r_1) in an outer loop over successive iterations. Hence why the algorithm needs the additions starting convergence factor $\epsilon_{c_{init}}$ and the relaxation value r_1 to reach the final convergence factor. 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 from tests we observed 0.5 to produced good results for both these factors. The bound movement ϵ_{mb} and contraction ϵ_{cb} factors, as their names suggest, 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%. From the tests in Section V.A.3, we observed good values for the contraction factor to be between 0.3-0.5, and 0.5 for the movement factor. The mathematical control for all these is defined by:

```
Convergence Check (R1):
    if $(\mathbf{ub} - \mathbf{lb}) < \epsilon_c(\mathbf{ub_{init}} - \mathbf{lb_{init}})$:
        lock converged variables and move to R2
    else
        continue to Exploratory Search
    end

All Bounds Locked Check (R2):
    if all variables are locked:
        unlock all variables and relax convergence factor by:
            $\epsilon_c = \epsilon_c r_1$    (2)
    end

Bound Movement:
    for all shared variables that have not converged
        Use previously successful direction,
            $ub_i = ub_i \pm \epsilon_{mb}(ub_i - lb_i)$    (3)
            $lb_i = lb_i \pm \epsilon_{mb}(ub_i - lb_i)$    (4)
        Check if move is Duplicate and/or Tabu.
        Optimise Domain Local Functions, $f_{1,2...k}$.
        Evaluate Global Objective Function, $f_0$.
        if outcome is an improvement:
            keep bounds, save direction and move to next variable.
        else:
            $ub_i = ub_i \mp \epsilon_{mb}(ub_i - lb_i)$    (5)
            $lb_i = lb_i \mp \epsilon_{mb}(ub_i - lb_i)$    (6)
        Check if move is Duplicate and/or Tabu.
        Optimise Local Functions, $f_{1,2...k}$.
        Evaluate Global Objective Function, $f_0$.
            if outcome is better:
                keep bounds, save direction and move to next variable.
            else:
                return bounds to previous base location.
            else:
        end
    end

Forced Bound Reduction:
    if outcome from Exploratory Search is an improvement:
        keep bounds, save direction and perform Pattern Move.
    else:
        force bound contraction only on unlocked shared variables.
            $\mathbf{ub} = \mathbf{ub} - \frac{\epsilon_{cb}}{2}(\mathbf{ub} - \mathbf{lb})$    (7)
            $\mathbf{lb} = \mathbf{lb} + \frac{\epsilon_{cb}}{2}(\mathbf{ub} - \mathbf{lb})$    (8)
        and then perform Pattern Move.
    end

Pattern Movement:
Try Pattern Move on shared variables.
    $\mathbf{ub} = \mathbf{ub} \pm \epsilon_{pat}(\mathbf{ub} - \mathbf{lb})$    (9)
    $\mathbf{lb} = \mathbf{lb} \pm \epsilon_{pat}(\mathbf{ub} - \mathbf{lb})$    (10)
Optimise Domain Local Functions, $f_{1,2...k}$.
Evaluate Global Objective Function, $f_0$.
    if pattern outcome is an improvement:
        keep bounds and proceed to R5.
    else:
        return to previous base point and proceed to R5.
    end
```

Here **ub** and **lb** are the vectors of shared variables bounds and ϵ_{pat} denotes the bound movement factor for the pattern move.

C. The Database and State Variables Strategy

The Multidisciplinary Pattern Search alone can solve decoupled problems with no interconnecting state variables, such as the conceptual aircraft test case in Section V.B. 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 V.A, 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 of information, while also storing bound movements in an encoded list for use in the duplicate check and Tabu rules.

In the sequential design process, organisational domains generally wait for others to finish their analyses before they start theirs. This ensures that the design being evaluated is consistent with respect to the state variables. Because this can take a long time, this 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 we don't! We simply assume 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, we accept that the proposed designs from each domain will be inconsistent and globally infeasible. However as the bounds start to reduce to a small 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 we expect this assumption to 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 Section VI.

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 we 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) solved these issues.

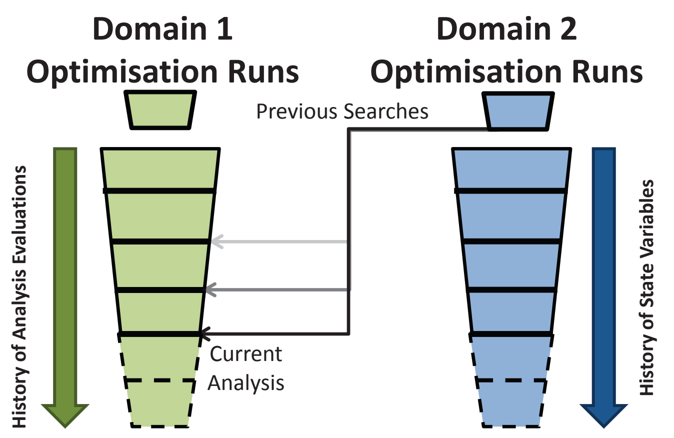


Figure 3. Domains loads the latest state variables and keep them constant for the duration of the local search

D. The System Interface

Next we propose an interface as a means for top level designers to monitor the process in real time. It shows the status of shared variables, objectives and constraints as well as an overview of the domain local searches. Figure 4 illustrates the convergence history of the UAV wing problem described in Section V.A. The interface uses the database to populate the eleven windows in real-time. The first four figures illustrate how the bounds on each variable move and reduce over successive top level iterations, with individual domain level preferences plotted between the two bounds. The following three figures show how each bound movement affects the final local objective and constraints from each domain preference.

The group of five smaller graphs at the bottom of the interface 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 ability to terminate 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.

V. Performance Validation

We applied the Blackboard search scheme on two deterministic test cases, which aim to capture aspects of the commercial aircraft design process, but at a level that could be handled by a single designer. The first is a UAV design problem developed in-house using empirical and physics based formulas, while the second uses a decommissioned industrial conceptual design tool provided to us by Airbus UK. We are aware that there exist numerous test cases used by the academic community specifically for benchmarking of MDO architectures [28], but we have found that some are highly non-linear and artificially too complex for the type of problems faced at the preliminary aircraft design stage. We have therefore chosen these two cases that we consider to be representative of the preliminary design problem, albeit simple enough to be implemented in-house. The problems are also solved here using the Simultaneous Analysis and Design [29] (SAND) and Collaborative Optimisation [30] (CO) architectures, whose solutions and convergence rates are used for comparison. We have specifically chosen these two architectures as they have few internally tuneable parameters, making them less susceptible to unintentional biasing. In the absence of a well-known global optimum as is the case for both problems, the results from the Simultaneous Analysis and Design architecture are assumed to produce the best obtainable result. The results from the Collaborative Optimisation architecture provide a guideline for the likely number of domain evaluations that might be needed by an established distributed MDO architecture.

A. The Wing Design Problem

The analyses within the UAV wing design problem comprise of empirical and physics based formulas, which are presented in the Appendix. These have been put together specifically to keep the domains free from black box codes, in order to ease later replication studies. The various analyses functions are grouped under two main domains: aerodynamics and structure. The aerodynamics domain seeks to reduce the wing drag while satisfying a wing loading constraint, whereas the structures domain tries to minimise the wing mass while satisfying maximum stress and manufacturing constraints. The global objective is simple weighted sum of each of the local objectives, i.e. drag and mass. Of course one can always combine these into the Breguet range equation and use that as an objective function as demonstrated by [7]. But in order to keep the objective function in the format given in Section IV.A.1, we deemed the weighted function to be sufficient in capturing the compromise between the two domains. In addition there are two state (coupling) variables. Wing mass is an output of the structures domain and is needed by aerodynamics domain and vice versa, the root bending moment is an output from the aerodynamics domain and needed in structures calculations.

Figure 5 shows the landscape of the problem as illustrated by a hierarchical axes plot. 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 7.

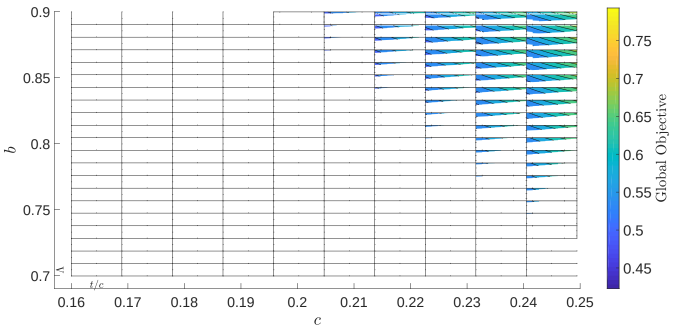


Figure 5. Hierarchical axes plot in four dimensions showing the global objective and constraints

1. *Implementation*

The primary role of the UAV problem is to illustrate the main features of the Blackboard search scheme. The following section explores how varying the rule base complexity, bound control factors and starting bound separation affected the outcome and speed of convergence. These were performed in addition to a comparison study examining the differences between the Blackboard search method and two established MDO architectures. Figure 6 shows the exact decomposition used by the Blackboard method. At the start of a top level iteration, the Multidisciplinary Pattern Search sets the design space and starting point for shared variables. Before the start of the local searches, both domains download the latest state variables (M_{root} and m_{wing}) from the database. These remain fixed for the duration of a search, while the internal domain optimisers control the remaining design variables. At the end of each search, the Multidisciplinary Pattern Search downloads the constraints ($\frac{W}{S}, t_{buff}, \sigma$) and the objectives ($m_{wing}, D$) and sets a new search space and starting point for the following iteration.

Unless otherwise stated, the default settings for the Blackboard are as follows:

- the bound control factors are $\epsilon_{c_{init}} = 0.5$, $\epsilon_{mb} = 0.5$, $\epsilon_{cb} = 0.5$ and $\epsilon_{c_{min}} = 0.01$;

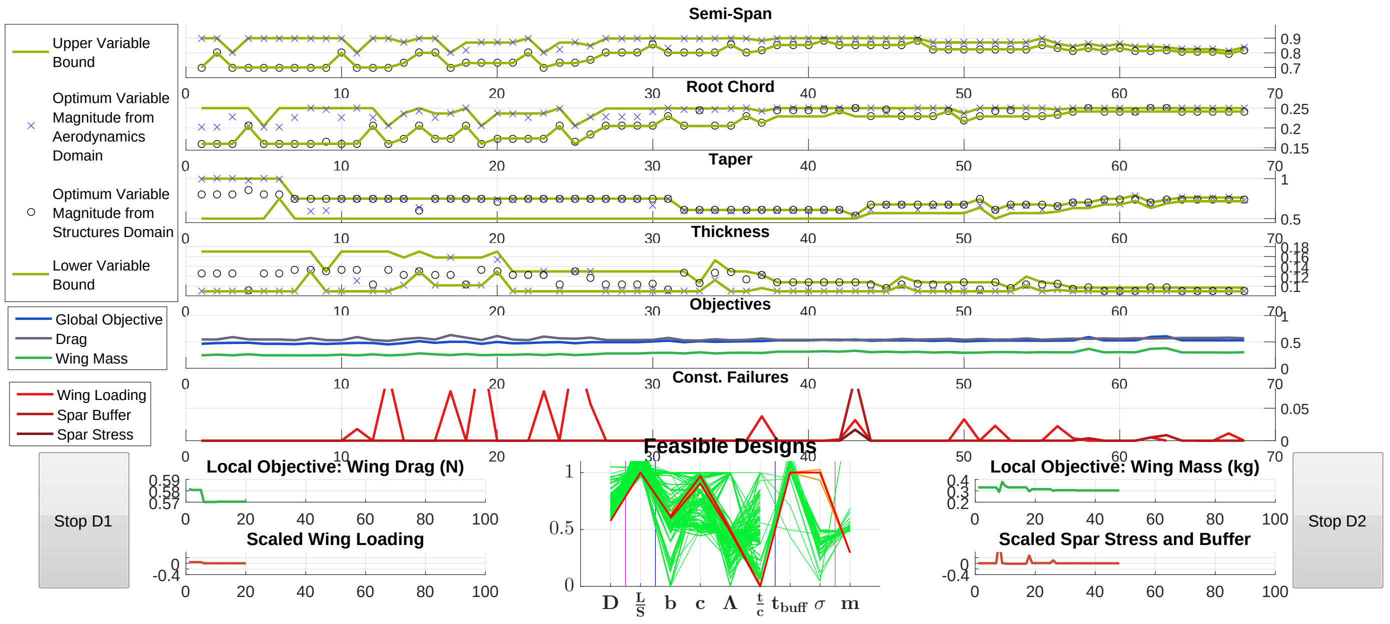
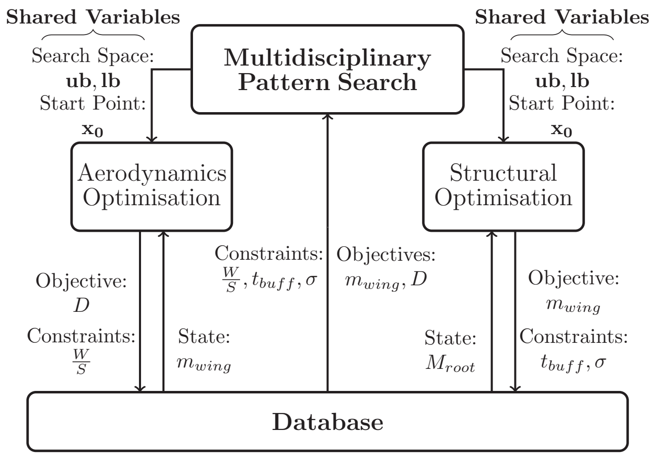


Figure 4. Top level user interface. The six main graphs show the effect of each bound movements. The smaller corner graphs illustrate the domain level iteration count. The lower graph is a parallel axis plot combining the feasible designs from each domain

- the computational steering element disabled and the rule base was allowed to run its course, in order to exclude any human factors 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;
- each new local search used 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.



Shared Variables
Search Space:
ub, lb
Start Point:
$\mathbf{x_0}$
Multidisciplinary Pattern Search
Aerodynamics Optimisation
Structural Optimisation
Objective: D
Constraints: $\frac{W}{S}$
State: m_{wing}
Constraints: $\frac{W}{S}, t_{buff}, \sigma$
Objectives: m_{wing}, D
State: M_{root}
Objective: m_{wing}
Constraints: t_{buff}, σ
Database

Figure 6. Blackboard data transfer for UAV wing design problem

2. *Additional Rules*

We have added significant complexity to the rule set beyond the original neighbourhood search. Here we wanted to investigate to what extent the additional rules/moves benefited the final outcome and the speed of convergence. Starting from the basic multidisciplinary neighbourhood search (NS), we then enabled the pattern move (PM), followed by the duplicate check (DC) and finally the Tabu move (TM). Table 2 shows the mean objectives, constraint failures and analysis calls from the 150 independent runs.

Table 2. Effect on additional rules on final outcome

	MDPS (NS)	MDPS (NS+PM)	MDPS (NS+PM+DC)	MDPS (NS+PM+DC+TM)
Mean for Magnitudes for the Objectives:				
Obj	**0.5204**	**0.5226**	**0.5231**	**0.5249**
D	0.5363	0.5404	0.5419	0.5445
m_{wing}	0.3059	0.3064	0.3064	0.3071
Mean Scaled Constraints Failure:				
	0.49%	0.23%	0.40%	0.21%
Mean Number of Bound Moves and Analysis Calls:				
Moves:	132	135	116	113
Aero. Calls:	1780	1849	1612	1600
Strc. Calls:	2468	2587	2228	2152
Total Calls:	**4248**	**4436**	**3840**	**3752**

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

We observe a general decrease in the number of analysis evaluations with increased complexity. 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 of a neighbourhood search, which in some cases may be discarded, thus increasing the number of analysis evaluations.

Secondly because the current preference value depends on the outcome of the previous search (see description on state variable strategy in Section IV.C), 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. So when we ran the test without the pattern move (but including the duplicate check and the Tabu move) we observed a slower overall performance. Furthermore when we started the local searches from a random point, rather than the previous optimum value, the pattern move alone reduced the total number of analysis evaluations by 13%.

Ultimately, we can represent the bound convergence for one optimisation run using Figures 7 and 8. Figure 7 is similar to Figure 4, but shows all the all function design inputs, bounds locations and analysis evaluations on a set of linear axes. Whereas Figure 8 overlays the changes in search space on the hierarchical axis plot, whereby the darkest region show the area where the bounds have converged.

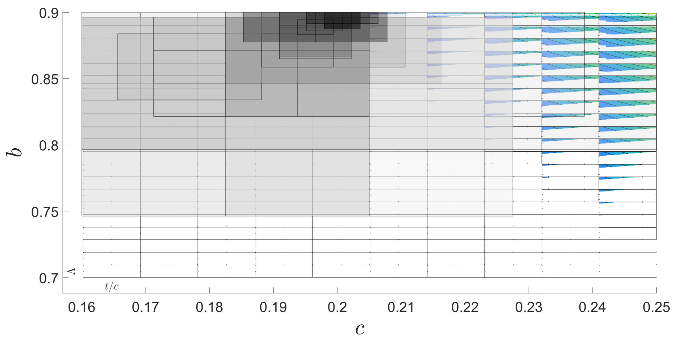


Figure 7. Bound movements superimposed on surface plot

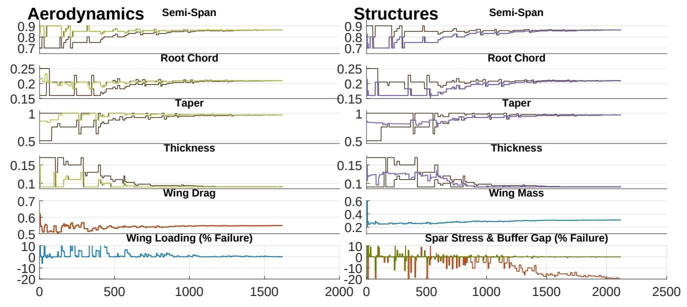


Figure 8. Bound movements superimposed on linear axis plot

3. *Bound Control Strategy*

This section looks at how the size of bound movement and contraction factors affected the performance of the rule base. The problem was started from the fixed starting point given in Table 5 (in the Appendix) and solved for an exhaustive range of bound movement and contraction factors, while keeping all others fixed at their default values.

Figure 9 and 10 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. We can observe 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. Now on the other hand the choice of bound reduction factor has a more pronounced effect, as can be seen by colour changes along the x-axes in both figures. We observe, as one would expect, that the algorithm converges much faster as we increase the bound contraction factor. 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 why we see a general increase in the objective value. Ultimately the outcomes of these tests motivated the selection of the previously detailed default bound control factors.

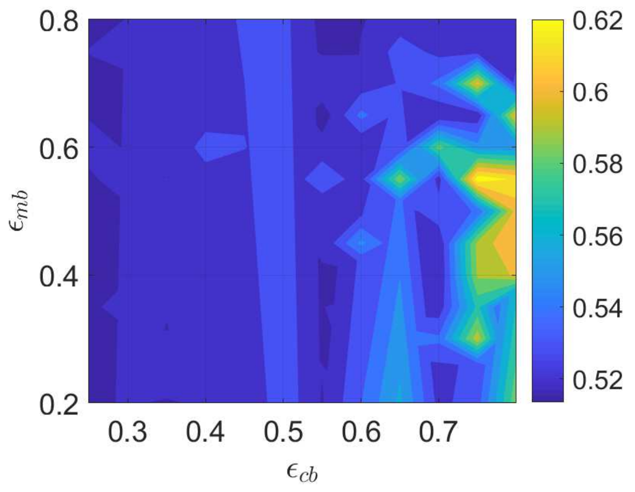


Figure 9. Effect of bound movement and contraction magnitude on the final design

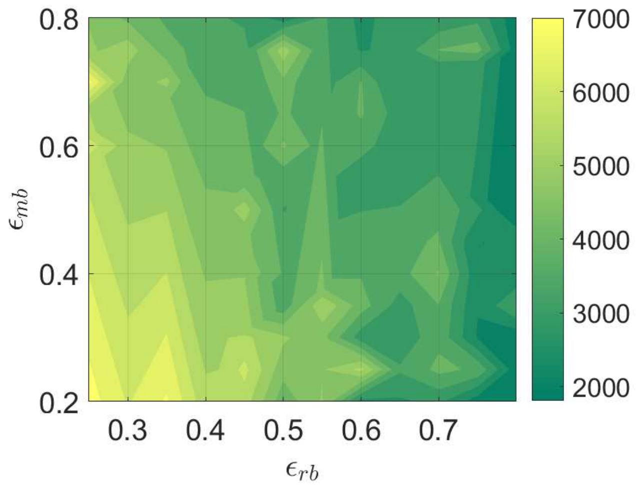


Figure 10. Effect of bound movement and contraction magnitude on the number of analysis evaluations

4. *Starting Bound Separation*

The Multidisciplinary Pattern Search has two tasks: to find the optimal design region and to reduce the design space. We wanted to determine if there was any benefit to start the process from a smaller bound spacing (still within the original bounds) and let the rule base focus on the searching task. Figures 11 to 15 show the results from 150 starting runs where the initial bound spacing was focussed around the starting point.

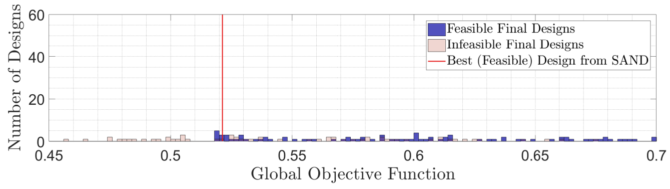


Figure 11. Starting Bound Separation 5%

The results presented in Figures 11 to 15 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 we have used

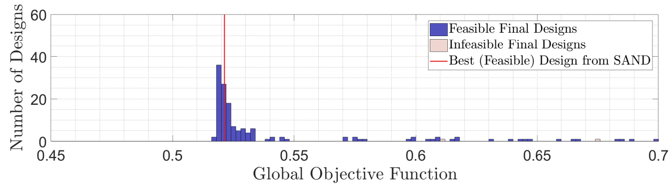


Figure 12. Starting Bound Separation 25%

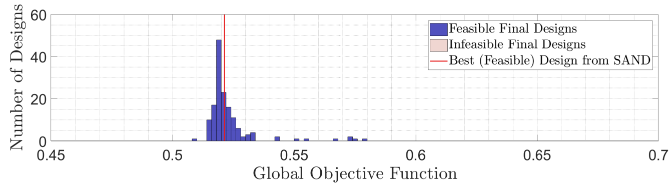


Figure 13. Starting Bound Separation 50%

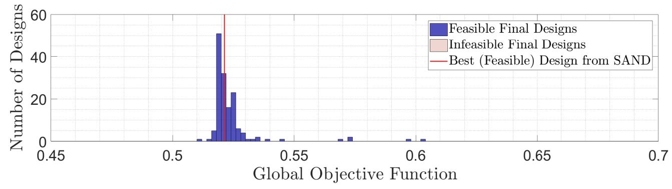


Figure 14. Starting Bound Separation 75%

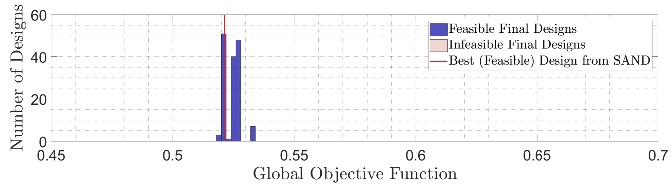


Figure 15. Starting Bound Separation 100%

in Figures 11 to 15. More importantly Figures 11 to 15 show that with increasingly smaller starting bounds, the process was more likely to get trapped in a local minimum. This is why we observe the different amount scatter between the five figures. Overall although we observed that the process converged slightly faster when started from narrower bounds, we see little benefit to starting the search from the narrower bound spacing.

5. *Comparison with Existing MDO Architectures*

These 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 optimise a single objective, which means that the problem structure has to be changed. Its optimal problem decomposition requires all domain objectives to be computed at the system level, with only certain selected analyses performed at domain level. If the aerodynamics and structures modules were modified to accommodate this decomposition, the UAV problem would fail to emulate the assumed industrial design 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) [31] strategy decomposition, 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 3 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 we have made every effort to ensure a direct and fair comparison between the architectures, the results presented in Table 3 should be viewed with a degree of caution. Both comparison architectures here have some (though few) internally tuneable parameters, which makes them susceptible to unintentional biasing [19]. This can of course mislead some the conclusions drawn from the results, hence why the results presented in Table 3 are viewed as a broad comparison rather than a critique of any given architecture.

Table 3. Comparison between three MDO architectures

MDO Architecture	SAND	MOCO	MDPS
Mean Objectives Value:			
Obj	**0.5225**	**0.5559**	**0.5231**
D	0.5360	0.5709	0.5445
m_{wing}	0.3081	0.3276	0.3071
Maximum Scaled Constraint Failure:			
	0%	0%	0.21%
Mean Number of Analysis Calls:			
Drag Module:	**80**	**123811**	**1600**
Wing Mass Module:	**80**	**122135**	**2152**

*MOCO and SAND are abbreviations for Multiobjective Collaborative Optimisation and Simultaneous Analysis and Design.

B. Industry Standard Tool

We have also been given access to a decommissioned engineering tool for conceptual transonic wing design [32]. It takes 10 wing variables and predicts the main performance metrics (wing drag and weight), as well as five aerodynamic and structural constraints. The drag estimation uses an analytical method that has been calibrated using commercial aerofoil sections. The wing mass estimation uses a stress based method to individually size the multiple structural elements making up an aluminium wing (ribs, stiffeners, skin thickness, etc.), A separate calibrated empirical method is used to size the secondary wing components (such as flaps, ailerons, spoilers, slats etc.). The aerodynamic loads are calculated using the ESDU 83040 [33] method and directly impact the primary wing structure, while the secondary wing structure changes to remain consistent with the load bearing elements.

The tool is treated as a black box evaluator and has been artificially decomposed in two domains. One domain has been set up to minimise wing drag, while satisfying a pitching moment constraint and the other minimises wing weight, while satisfying wing volume and undercarriage bay length constraints. Here we have also chosen Matlab's SQP optimiser within the function *fmincon* to perform the domain local optimisation. In addition an objective function (Equation 11), similar to the previous problem, was devised to collectively minimise the wing drag and mass at the system level. In some respects this decomposition makes the problem significantly less complex than the previous one, because the domains are standalone and have no interconnecting state variables as shown by Figure 16. The difficulty however comes from the higher number design variables, as well as the internal domain non-linearities, which for this problem are unknown. The problem decomposition for the Blackboard is illustrated in Figure 16 and the global objective function is given as follows.

$$Obj. = 4.7 \times D/q + 1.05 \times 10^{-4} W_{wg}. \tag{11}$$

Here D/q and W_{wg} represent the ratio of wing drag to dynamic pressure and wing weight, while the two weighing factors for the objectives were also chosen arbitrarily. The problem was solved using the Blackboard as well as the two classical MDO architectures that were used in the previous comparison, with the results of the study presented in Table 4.

We know from the findings in the report by Price et. al [2] 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 problem variant by Price et. al [2]. Failures with the architectures have been noted by Alexandrov and

Table 4. Aircraft design problem variables and results

	L. Limit	U. Limit	Units	Starting	SAND	CO	MDPS
Find Variables:							
Aspect Ratio	6.0	12.0	-	10.000	10.982	fail	8.6602
Leading Edge Sweep	25	45.0	deg	26.000	27.012		26.172
Spanwise Kink Position	0.2	0.45	-	0.3000	0.2000		0.4332
Inner Taper Ratio	0.4	0.7	-	0.7000	0.6581		0.6350
Outer Taper Ratio	0.2	0.6	-	0.2000	0.2000		0.2281
Root Thickness to Chord Ratio	0.1	0.18	-	0.1700	0.1000		0.1129
Kink Thickness to Chord Ratio	0.06	0.14	-	0.1400	0.1350		0.0745
Tip Thickness to Chord Ratio	0.06	0.14	-	0.0900	0.0600		0.0745
Wing Washout	2.0	5.0	deg	3.0000	3.0779		4.4822
Fraction Tip Washout at Kink	0.65	0.85	-	0.6500	0.6500		0.7518
That Minimise:							
Obj.			-	**27.650**	**25.200**		**25.938**
Drag, D/q			m^2	3.8816	3.1933		3.4738
Wing Weight, W_{wg}			N	89589	97061		90765
Subject To:							
Pitching Moment		5.4	-	4.1323	4.8719		3.5498
Wing Volume	23.0		m^3	33.328	25.708		22.682
Undercarriage Bay Length	2.1		m	2.2986	2.4999		2.3698
Aero. Analysis Calls					223		16576
Strc. Analysis Calls					223		7094

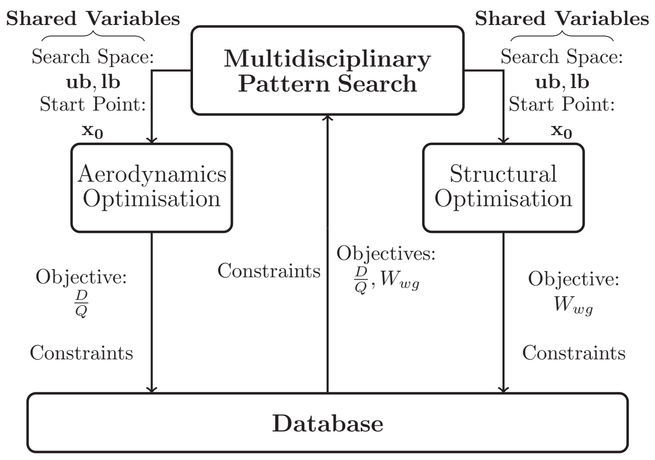


Figure 16. Blackboard data transfer for transonic wing design problem

Lewis [34] 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 [31] for the Multiobjective version of Collaborative Optimisation that was used here. This therefore hinders the application of most gradient based optimisers, which explains why the solution failed under Matlab's SQP optimiser.

Simultaneous Analysis and Design finds what we consider to be the best answer. It is noteworthy that the outcome from this architecture obtained 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 optimal in real life. This outcome is therefore 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 [35].

The Blackboard search method, with the Multidisciplinary Pattern Search, has no formal guarantees of global convergence. This is why it converged to a local minimum with a lower objective value than the Simultaneous Analysis and Design method. Nevertheless the obtained final result is still significantly better than the starting point, which is after all a positive outcome when compared to the failure of Collaborative Optimisation.

VI. Discussion and Future Work

The Blackboard based approach has the advantage that it is relatively straightforward to implement in existing organisational structures. As stated by Agte et al. [3], it is often easy for researches to focus on the technical worth of a distributed MDO architecture and overlook practical aspects regarding its application in a real world design environment. We have therefore focused to develop an MDO method that requires minimal internal domain restructuring, but instead sacrifices certain other features that some existing MDO architectures claim to possess.

For example, we are aware that the Blackboard is unsuitable for some highly non-linear problems, which are often found in academic test suits [28]. This is because coupling variables remain fixed during domain local optimisation searches, which can result in severe oscillations and ultimately convergence to infeasible designs. Even though many of the problems found in academic test suits can be solved by existing distributed MDO architectures, we have deemed them to be artificially too complex for the type of problems facing aircraft designers.

Instead we have chosen two aircraft design test cases that emulate the difficulties in the early preliminary design stage. We observed that in some rare cases, the assumed final outcome from the Blackboard failed the constraints by a very small margin, even though the constraints in the individual domain were satisfied. This is because each domain outputs a slightly different final result within the final 1% bound spacing. These differences manifest as constraint failures when the final design is assumed to be a combination of the final outputs from each domain. Ultimately the constraints failures were well below the accuracy of chosen analytical methods and in any case could be absorbed in the factor of safety. While these could be eliminated by further tightening the convergence tolerance, we found it inefficient to reduce the bound separation below 1% when the answer could be found in fewer iterations using a serial design method that uses the output of the Blackboard as a starting point.

The convergence of the Blackboard method still remains slow, even though the newly developed Multidisciplinary Pattern Search has a number of data mining elements, which its predecessor rule base lacked [2]. Although the results are on par with what can be expected from a distributed MDO architecture, a further reduction in the number of function calls is needed to make this a viable industrial design method. We see the use of surrogates as a way to further reduce the number of analysis evaluations without affecting the practicality of the method. In the current framework, domains populate the database with design inputs and analysis outputs. This information remains largely unused over the course of a top level search. Previous works [36, 37] have demonstrated that one can build accurate surrogate models using sufficiently well populated legacy datasets. It is therefore the aim of future work to use the available data and build surrogate models that could eventually be used in place of the "true" domain analyses. Because this would require nothing more than access to the database, a separate Data Mining module could be used to build and tune these models in parallel to the domain level searches and therefore still preserve the internal domain structure. Furthermore these models could also be used by other domains to provide better estimates of state variables, rather than the current strategy in use, which only extracts the latest state variable in the database.

Finally, it is worth reiterating that by adopting a rule based approach, it is difficult to carry out any formal global convergence analysis. The aim instead has been to provide an intuitive way to control and monitor the activity of organisational domains, which might appeal to chief designers without a specialist MDO background.

VII. Conclusion

Monolithic MDO architectures performed much better than the distributed alternatives, because when used with a suitable optimiser they can be both robust and converge quickly, for a wide variety of problems. However, because they require all organisational domains to be grouped under a single optimiser, they are unsuitable for the preliminary stage of industrial aircraft design. Classical distributed MDO architectures are in theory more suitable, but have thus far seen very little take up in industry. For this reason we revisited a legacy Blackboard framework [2], which sacrifices formal guarantees of global convergences in order to maintain the current organisational domain structure.

Our work has focussed on the development of a new rule base and visualisation modules to improve robustness, human-process interaction and speed up convergence of the Blackboard search method. We demonstrated the merit of a newly developed rule base and observed that in general it converged to a better result than the starting point for the two aircraft design problems it was tested on. The Blackboard was shown to perform better when compared against the alternative Collaborative Optimisation architecture. However, it was not able to match both speed and accuracy of the chosen monolithic architecture, which in practice tends to be organisationally unsuitable.

We believe that the work on the rule base has matured sufficiently to allow several avenues for further research to be explored. Firstly we aim to introduce domain level data mining using surrogates. The data obtained from multiple domain level searches will be used to build a surrogate, which can then be evaluated instead of the individual domain level analysis. In this way we hope to further reduce the number of analysis evaluations used by the Blackboard in order to make it more competitive against the monolithic optimisation architectures. Secondly our tests so far have demonstrated the merit of the method on idealised, deterministic test cases with few shared and state variables. Future work will focus to extend the complexity of the problems in numbers of shared design variables, number of domains and higher analyses fidelity. Finally, it is our ultimate goal to apply and test the applicability of the Blackboard to a team based design problem where human factors will also play a role in the process.

Appendix

The UAV wing design problem consists of two domains, each tasked with optimising a simple wing geometry such that its mass and drag are collectively reduced for a fixed payload, fuselage and empennage mass. The four shared variables that describe the geometry are the semi-span b, chord c, taper Λ and maximum aerofoil thickness to chord ratio $\left(\frac{t}{c}\right)_{max}$.

In the aerodynamics domain, we aim to minimise total wing drag and satisfy a limit on the wing loading. The stall speed is generally the leading constraint in industrial aerodynamic design. However, because of the absence of simple empirical or physics based methods for estimating stall speeds accurately, we have used wing loading as constraint at cruise, as it indirectly affects the stall speed.

The wing consists of a circular main spar encased in a foam outer section. The structural domain aims to minimise wing mass, while satisfying a maximum stress constraint using a cantilever beam model. In the analysis we neglect wing torque as well as any effects from the drag loads. For simplicity the global objective (Equation 12) is constructed as a weighted sum of the conflicting wing drag and mass objectives with the weightings ($\alpha_1 = 0.4, \alpha_2 = 1$) selected here arbitrarily,

$$Obj. = \alpha_1 D + \alpha_2 m_{wing}. \tag{12}$$

The total wing drag (D) in Newtons is obtained from the aerodynamics domain (Equation 13) and the wing mass (m_{wing}) in kilograms is taken from the structures domains (Equation 30).

The aerodynamics objective is to minimise drag as:

$$D = \frac{1}{2}\rho V^2 S \left(C_{Di} + C_{Dp}\right) \tag{13}$$

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.

Lift induced drag is obtained from:

$$C_{Di} = \frac{C_L^2}{\pi ARe} \tag{14}$$

with C_L as the coefficient of lift, AR as the aspect ratio and e the span efficiency.

The profile drag coefficient (C_{Dp}) is obtained from 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:

$$C_{Dp} = C_f f_{tc} \frac{S_{wet}}{S}. \tag{15}$$

The Blasius relationships for 100% laminar flow [38] is used for the mean skin friction coefficient calculation as:

$$C_f = \frac{1.328}{\sqrt{Re}} \tag{16}$$

where Re is the Reynolds number based on the mean aerodynamic chord.

The form factor is obtained from Torenbeek's [38] empirical formula:

$$f_{tc} = 1 + 2.7\left(\frac{t}{c}\right)_{max} + 100\left(\frac{t}{c}\right)_{max}^4 \tag{17}$$

with $\left(\frac{t}{c}\right)_{max}$ denoting the aerofoil maximum thickness to chord ratio. The wetted wing [39] area is estimated as follows:

$$S_{wet} = 2\left(1 + 0.5\left(\frac{t}{c}\right)_{max}\right)S. \tag{18}$$

The wing loading constraint is computed as follows.

$$\left(\frac{W}{S}\right)_{min} \leq \left(\frac{m_{tot} g}{S}\right). \tag{19}$$

Modern Computation Fluid Dynamics (CFD) packages compute the loads and subsequent span-wise bending moments, which feature in the structural design. The following section uses a model loosely based on lecture notes from the Massachusetts Institute of Technology Open Course (see note [a]) to calculate the span-wise loads and subsequent bending moments resulting from flight at given load factor. We note that there are also other analytically derived methods for generating a loading distribution, such as those proposed by Schrenk [40] or ESDU 83040 [33], although these require separate wing inertial loads estimates from the wing mass.

The span-wise loading here is assumed to be proportional to the local chord length as given by:

$$q(y) = \frac{NW_{excl.wing}}{b(1 + \Lambda)}(1 + (\Lambda - 1)\frac{y}{b}) \tag{20}$$

[a]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

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 as:

$$M(y) = \int_0^b \int_0^b q(y)dydy. \tag{21}$$

We have used the trapezium numerical integration method with 50 span-wise points. The structural domain in commercial aircraft design is largely concerned with designing and optimising a structure that can satisfying the multitude of load cases arising from the different operating conditions. For simplicity, the structures domain here aims to minimise the wing mass by satisfying a load case for a 7g vertical gust. The structural domain assumes that the wing comprises of a circular main spar of diameter d_{spar} and thickness t_{spar}, which runs the length of the span. The spar is constrained to fit inside a foam section as shown in Figure 17 [41].

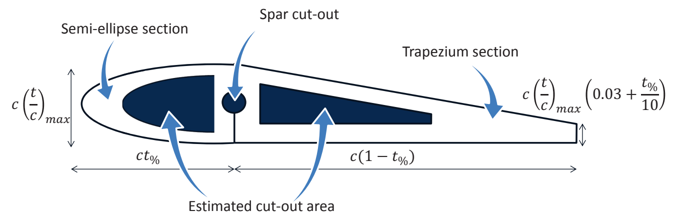

Semi-ellipse section
Spar cut-out
Trapezium section
$c\left(\frac{t}{c}\right)_{max}$
$c\left(\frac{t}{c}\right)_{max}\left(0.03 + \frac{t_{\%}}{10}\right)$
$ct_{\%}$
$c(1 - t_{\%})$
Estimated cut-out area

Figure 17. Cross-section of the wing profile and internal structure

The main spar provides the stiffness for the wing, with any additional stiffness from the foam assumed to be negligible. The structural domain optimises an estimated wing mass (Equation 30), while satisfying the maximum bending stress constraint resulting from the gust load. Additional constraints ensure the spar fits inside the foam geometry with sufficient buffer spacing from the edges and that the thickness does not exceed the radius of the spar as given by:

$$t_{spar} + 0.01 \geq \left(\frac{t}{c}\right)_{max} c \tag{22}$$

and

$$t_{spar} \geq 0.001. \tag{23}$$

The stress on the main spar is assumed to act on a single point (at the root where the bending moment is the largest). It is calculated using the formula:

$$\sigma = \frac{M_{root} z}{I_y} \tag{24}$$

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 total wing mass is calculated from the sum of the foam section and spar masses. A simplified mass model estimates the mass of the foam section based on the volume of a frustum as given by Equation 28.

The root profile is estimated on the combined areas of a semi-ellipse and trapezium, shown in Figure 17 by:

$$A_{root} = p_{cut}\left[\frac{1}{4}\pi c^2 t_{\%}\left(\frac{t}{c}\right)_{max} + \frac{1}{2}c^2(1 - t_{\%})\left(\frac{t}{c}\right)_{max}\left(1 + 0.03 + \frac{t_{\%}}{10}\right)\right] \tag{25}$$

where $t_{\%}$ is the chord-wise location of maximum thickness and p_{cut} is the cut-out fraction calculated from Equation 26.

The wing cut-out fraction is estimated by:

$$p_{cut} = 1 - \left(\frac{t}{c}\right)_{max}. \tag{26}$$

The cross-sectional area at the tip is:

$$A_{tip} = \Lambda^2 A_{root}. \tag{27}$$

The total foam section is hence calculated from the volume of a frustum:

$$m_{foam} = \frac{2}{3}b(A_{root} + \sqrt{A_{root}A_{tip}} + A_{tip})\rho_{foam}. \tag{28}$$

The mass of the spar is simply:

$$m_{spar} = 2A_{spar}b\rho_{spar} \tag{29}$$

where A_{spar} is the cross-sectional area of the spar and ρ_{spar} is density of the spar material.

The total mass is then:

$$m_{wing} = m_{foam} + m_{spar} + m_{aux} \tag{30}$$

where m_{aux} is the mass of servos and other wing elements that are assumed fixed. The manufacturing buffer constraint is given as

$$t_{buff} = 0.01 \geq c\left(\frac{t}{c}\right)_{max} - t_{spar}, \tag{31}$$

while the stress constraint is

$$FOS\,\sigma_{UTS} \geq \sigma. \tag{32}$$

Here t_{buff} is a buffer section between the spar and the skins of the profile, FOS is the factor of safety, σ_{UTS} is the ultimate tensile strength of the material and σ is the stress resulting from bending.

Table 5 summarises all the variables used in the problem and sets a fixed starting point for the bound control factor tests. Highlighted within are also certain values, such as the Oswald's efficiency, that are fixed or assumed to be approximately constant.

VIII. Acknowledgements

This work was jointly funded by Airbus UK. and the University of Southampton, whose support is gratefully acknowledged.

References

[1] Tedford, N. P., and Martins, J. R. R. A., "Benchmarking multidisciplinary design optimization algorithms," *Optimization and Engineering*, Vol. 11, No. 1, 2010 pp. 159–183.
doi: 10.1007/s11081-009-9082-6

[2] Price, A. R., Keane, A. J., and Holden, C. M. E., "On the coordination of multidisciplinary design optimization using expert systems," *AIAA journal*, Vol. 49, No. 8, 2011 pp. 1778–1794.
doi: 10.2514/1.J050928

[3] Agte, J., De Weck, O., Sobieszczanski-Sobieski, J., Arendsen, P., Morris, A., and Spieck, Martin. "MDO: assessment and direction for advancements and opinion of one international group," *Structural and Multidisciplinary Optimization*, Vol. 40, No. 1-6, 2010 pp. 17–33.
doi: 10.1007/s00158-009-0381-5

[4] Braun, R. D., and Kroo, I., "Development and application of the collaborative optimization architecture in a multidisciplinary design environment," *Multidisciplinary Design Optimization: State of the Art*, NASA Technical Report, 1995.

[5] Martins, J. R. R. A., and Lambe, A. B., "Multidisciplinary Design Optimization: A Survey of Architectures," *AIAA Journal*, Vol. 51, No. 9, 2013 pp. 2049–2075.
doi: 10.2514/1.J051895

[6] Gazaix, A., Gallard, F., Gachelin, V., Druot, T., Grihon, S., Ambert, V., Guénot, D., Lafage, R., Vanaret, C., Pauwels, B., Bartoli, N., Lefèbvre., T., Sarouille, P., Desfachelles, N., Brézillon, J., Hamadi, M., and Gürol, S., "Towards the Industrialization of New MDO Methodologies and Tools for Aircraft Design," *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2017 p. 3149.
doi: 10.2514/6.2017-3149

Table 5. UAV wing design objectives, variables and constraints

Objectives	L. Bound	Symbol	U. Bound	Units	Starting Point
Minimise:					
Global Obj.	-	Obj	-	-	0.5667
Wing drag	-	D	-	N	0.5861
Wing mass	-	m_{wing}	-	kg	0.3321
Shared Optimisation Design Variables					
Semi-span	0.7	b	0.9	m	0.8000
Root chord	0.16	c	0.25	m	0.2000
Taper	0.5	Λ	1.0	-	1.0000
Thick. to chord ratio	0.09	$\frac{t}{c}$	0.17	-	0.1200
Domain Local Optimisation Design Variables					
Spar:					
Wall thickness	0.0010	t_{spar}	0.0080	m	0.0010
Outer diameter	0.0020	d_{spar}	0.0160	m	0.0080
State Variables					
Wing mass	-	m_{wing}	-	kg	0.3321
Max. bend. mom.	-	M_{root}	-	-	7.5843
Fixed Variables					
Chord-wise location of max. thickness	-	$t_{\%}$	-	-	0.4000
Cruise speed	-	V	-	m/s	13
Oswald efficiency	-	e	-	-	0.84
UAV mass excluding wing	-	$m_{excl.wing}$	-	kg	1.150
Aux. masses on wing	-	m_{aux}	-	kg	0.150
Factor of safety	-	FOS	-	-	1.4
Spar density	-	ρ_{spar}	-	kg/m^3	1550
Foam density	-	ρ_{foam}	-	kg/m^3	30
Max. allowable stress	-	σ_{UTS}	-	MPA	590
Load factor	-	N	-	-	7
Domain Local Constraints					
Spar buffer	0.01	t_{buff}	-	m	0.0160
Wing loading	-	$\frac{W}{S}$	40	N/m^2	44.45
Tensile stress	-	σ	421.4	MPa	220.7

*All atmospheric properties are fixed at International Standard Atmosphere (ISA) sea level conditions.

[7] Shahan, D. and Seepersad, C. C., "Bayesian networks for set-based collaborative design," *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, American Society of Mechanical Engineers, 2009, pp. 303–313.
doi: 10.1115/detc2010-28724

[8] Lewis, K. and Mistree, F., "Modeling interactions in multidisciplinary design: A game theoretic approach," *AIAA journal*, Vol. 35, No. 8, 1997, pp. 1387-1392.
doi: 10.2514/2.248

[9] Xiao, M., Shao, X., Gao, L., and Luo, Z., "A new methodology for multi-objective multidisciplinary design optimization problems based on game theory," *Expert Systems with Applications*, Vol. 42, No. 3, 2015, pp. 1602–1612.
doi: 10.1016/j.eswa.2014.09.047

[10] Wang, G.G. and Shan, S., "Design space reduction for multi-objective optimization and robust design optimization problems," Tech. rep., SAE Technical Paper, 2004.
doi: 10.4271/2004-01-0240

[11] Hannapel, S.E., "Development of Multidisciplinary Design Optimization Algorithms for Ship Design Under Uncertainty," Ph.D. Thesis, The University of Michigan, Michigan, 2012.

[12] Ollar, J., Toropov, V., and Jones, R., "Sub-space approximations for MDO problems with disparate disciplinary variable dependence," *Structural and Multidisciplinary Optimization*, Vol. 55, No. 1, 2017, pp. 279–288.
doi: 10.1007/s00158-016-1496-0

[13] Nii, H. P., "The blackboard model of problem solving and the evolution of blackboard architectures," *AI magazine*, Vol. 7, No. 2, 1986, pp. 38.
doi: 10.1609/aimag.v7i2.537

[14] Khedro, T., Genesereth, M. R., and Teicholz, P. M., "Agent-based framework for integrated facility engineering," *Engineering with computers*, Vol. 9, No. 2, 1993, pp. 94–107.
doi: 10.1007/BF01199048

[15] Szczerbicki, C Reidsema, E., "A blackboard database model of the design planning process in concurrent engineering," *Cybernetics & Systems,* Vol. 32, No. 7, 2001, pp. 755–774.
doi: 10.1080/019697201317080952

[16] Nicolai, L. M., Carichner, G., and Malcolm, L., *Fundamentals of Aircraft and Airship Design: Volume 1, Aircraft Design*, American Institute of Aeronautics and Astronautics, 2010.
doi: 10.2514/4.867538

[17] Yang, M. C. and Jin, Y., "An examination of team effectiveness in distributed and co-located engineering teams," *International Journal of Engineering Education*, Vol. 24, No. 2, 2008, pp. 400.

[18] Yu, B. Y., Honda, T., Sharqawy, M., and Yang, M., "Human behavior and domain knowledge in parameter design of complex systems," *Design Studies*, Vol. 45, 2016, pp. 242–267.
doi: 10.1016/j.destud.2016.04.005

[19] Jelev, N., Keane, A., Holden, C., and Sóbester, A., "Rule Based Architecture for Collaborative Multidisciplinary Aircraft Design Optimisation," *International Journal of Aerospace and Mechanical Engineering*, Vol. 11, No. 5, 2017, pp. 1006–1015.
doi: 10.1999/1307-6892/10007081

[20] Hooke, R. and Jeeves, T. A., "Direct Search Solution of Numerical and Statistical Problems," *Journal of the ACM (JACM)*, Vol. 8, No. 2, 1961, pp. 212–229.
doi: 10.1145/321062.321069

[21] Glover, F., "Tabu Search - Part I," *ORSA Journal on computing*, Vol. 1, No. 3, 1989, pp. 190–206.
doi: 10.1287/ijoc.1.3.190

[22] Bell, M. and Pike, M. C., "Remark on Algorithm 178 [E4] Direct Search," *Commun. ACM*, Vol. 9, No. 9, Sept. 1966, pp. 684–685.
doi: 10.1145/365813.365825

[23] Vázquez, S., Martín, M. J., Fraguela, B. B., Gómez, A., Rodriguez, A., and Elvarsson, B., "Novel parallelization of simulated annealing and Hooke & Jeeves search algorithms for multicore systems with application to complex fisheries stock assessment models," *Journal of AAPOS Computational Science*, Vol. 17, 2016, pp. 599–608.
doi: 0.1016/j.jocs.2016.07.003

[24] Glover, F. and Taillard, E., "A user's guide to tabu search," *Annals of operations research*, Vol. 41, No. 1, 1993, pp. 1–28.
doi: 10.1007/BF02078647

[25] Connor, A. M. and Tilley, D. G., "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*, Vol. 212, No. 5, 1998, pp. 373–381.
doi: 10.1243/0959651981539541

[26] Jaeggi, D., Parks, G. T., Kipouros, T., and Clarkson, P. J., "The development of a multi-objective Tabu Search algorithm for continuous optimisation problems," *European Journal of Operational Research*, Vol. 185, No. 3, 2008, pp. 1192–1212.
doi: 10.1016/j.ejor.2006.06.048

[27] Lewis, R. M., Torczon, V., and Trosset, M W., “Direct search methods: then and now,” *Journal of computational and Applied Mathematics*, Vol. 124, No. 1, 2000, pp. 191–207.
doi: 10.1016/S0377-0427(00)00423-4

[28] Padula, S. L., Alexandrov, N., and Green, L. L., “MDO test suite at NASA Langley Research Center,” *Sixth AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, AIAA Paper No. 96-4028, 1996.
doi: 10.2514/6.1996-4028

[29] Haftka, R. T., “Simultaneous analysis and design,” *AIAA Journal*, Vol. 23, No. 7, 1985, pp. 1099–1103.
doi: 10.2514/3.9043

[30] Kroo, I., Altus, S., Braun, R., Gage, P., and Sobieski., I., “Multidisciplinary optimization methods for aircraft preliminary design,” *In 5th Symposium on Multidisciplinary Analysis and Optimization,* p. 4325. 1994.
doi: 10.2514/6.1994-4325

[31] Tappeta, R. and Renaud, J., “Multiobjective collaborative optimization,” *Journal of Mechanical Design*, Vol. 119, No. 3, 1997, pp. 403–411.
doi: 10.1115/1.2826362

[32] Cousin, J. and Metcalf, M., “The BAe (commercial aircraft) LTD transport aircraft synthesis and optimisation program (TASOP),” *Aircraft Design, Systems and Operations Conference*, 1990, p. 3295.
doi: 10.2514/6.1990-3295

[33] Catalogue : ESDU validated engineering data, ESDU London, 1983.

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

[35] Keane, A. and Nair, P., *Computational approaches for aerospace design: the pursuit of excellence*, John Wiley & Sons, 2005. doi: 10.1002/0470855487

[36] Srivastava, A., Hacker, K., Lewis, K., and Simpson, T., “A method for using legacy data for metamodel-based design of large-scale systems,” *Structural and Multidisciplinary Optimization*, Vol. 28, No. 2-3, 2004, pp. 146–155.
doi: 10.1007/s00158-004-0438-4

[37] Rennen, G., “Subset selection from large datasets for kriging modeling,” *Structural and Multidisciplinary Optimization*, Vol. 38, No. 6, 2009, pp. 545.
doi: 10.1007/s00158-008-0306-8

[38] Torenbeek, E., *Synthesis of subsonic airplane design: an introduction to the preliminary design of subsonic general aviation and transport aircraft, with emphasis on layout, aerodynamic design, propulsion and performance*, Springer Science & Business Media, 1982. doi: 10.1007/978-94-017-3202-4

[39] Gudmundsson, S., *General aviation aircraft design: applied methods and procedures*, Butterworth-Heinemann, 2013. doi: 10.1016/c2011-0-06824-2

[40] Schrenk, O., “A simple approximation method for obtaining the spanwise lift distribution,” *The Aeronautical Journal*, Vol. 45, No. 370, 1941,pp. 331–336.
doi: 10.1017/S0368393100101075

[41] Keane, A., Sóbester, A., and Scanlan, J., *Small Unmanned Fixed-wing Aircraft Design: A Practical Approach*, John Wiley & Sons, 2017. doi: 10.1002/9781119406303.app1

[42] Choi, S., Alonso, J. J., and Kroo, I. M., “Two-level multifidelity design optimization studies for supersonic jets,” *Journal of Aircraft*, Vol. 46, No. 3, 2009, pp. 776–790.
doi: 10.2514/1.34362

[43] Jameson, A., “Re-engineering the design process through computation,” *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 36–50.
doi: 10.2514/2.2412

[44] Chittick, I. R. and Martins, J. R. R. A., “An asymmetric suboptimization approach to aerostructural optimization,” Optimization and Engineering, Vol. 10, No. 1, 2009, pp. 133–152.
doi: 10.1007/s11081-008-9046-2

[45] Roth, B. D. and Kroo, I. M., “Enhanced collaborative optimization: a decomposition-based method for multidisciplinary design,” *Proceedings of the ASME design engineering technical conferences, Brooklyn, New York*, 2008, pp. 3–6.
doi:10.1115/DETC2008-50038