DOI: 10.13009/EUCASS2023-834
Aerospace Europe Conference 2023 – 10TH EUCASS – 9TH CEAS

Comparative review of Multidisciplinary Design Analysis and Optimization architectures for the preliminary design of a liquid rocket engine

Lucas Baraton[1,2,†], *Annafederica Urbano*[1], *Loïc Brevault*[2] *and Mathieu Balesdent*[2]
[1]*ISAE SUPAERO, Université de Toulouse, Toulouse, France*
[2]*ONERA DTIS, Université Paris-Saclay, Palaiseau, France*
{lucas.baraton, annafederica.urbano}@isae-supaero.fr · {loic.brevault, mathieu.balesdent}@onera.fr
[†]Corresponding author

Abstract

Multidisciplinary Design Analysis and Optimization (MDAO) problems can involve system architecture choices, represented by design variables (called conditional variables) modifying the optimization problem's structure (*e.g.*, number of design variables and constraints). The associated problem is called Conditional Search-Space Problem (CSSP). This paper proposes a MDAO formulation of the CSSP, and presents a relaxation methodology to solve a certain class of problems. The method is compared to the classical approach consisting in optimizing every possible architecture and illustrated on a liquid rocket engine application. Two MDAO formulations are compared on the relaxed problem: Multidisciplinary Feasible (MDF) and Individual Discipline Feasible (IDF).

1. Introduction

The performances of a launcher are a key driver regarding the success of a space mission. The overall architecture of the launch vehicle and in particular its propulsion system are important elements to be optimized in order to reach an acceptable level of performance, reliability and profitability. The growing complexity of the design process brings the need of novel numerical methods to deal with this highly constrained non-linear optimization problem. For example, the propulsion system design is a typical multidisciplinary problem involving several disciplines (or subsystems) such as the thrust chamber, the feedsystem and the cooling system.

The Multidisciplinary Design Analysis and Optimization (MDAO) framework is a powerful set of tools for the preliminary design of engineering systems[37] because it allows to solve large, highly coupled and multidisciplinary problems. Numerous problem decompositions (MDAO architectures) and optimization algorithms have been developed in the last decades to solve increasingly complex design problems.[36] For instance, MDAO has been used to study the propulsion system design problem: Cai *et al.*[15] used MDAO to optimize a Liquid Rocket Engine (LRE) equipped with a gas-generator cycle propelled by liquid oxygen (LOX) and hydrogen (LH2). Okninski *et al.*[39] also performed MDAO to optimize an upper-stage engine for in-orbit use.

Besides, in the preliminary design phases, architectural and technological choices need to be made for the propulsion system. These choices strongly impact the overall system behavior. Within the mathematical formulation of the design problem, they can be embedded in a particular type of design variables which, depending on their values, modify the structure of the optimization problem (in particular, the number of design variables and constraints can be impacted). When designing the first stage of a launcher, the propulsion type, solid or liquid, is an example of such a variable. In the first case, variables relative to the dimensions of the solid propellant grain geometry must be optimized, and the system is typically constrained by the maximum stress on the booster case. In the second one, the regenerative cooling system and its parameters (*e.g.*, number and geometry of the cooling channels) must be selected in order to keep the thrust chamber wall temperature under a maximum allowable value. As the value of the variable '*propulsion type*' conditions which variables and constraints act on the problem, it is called a conditional variable in this paper. The associated optimization problem is called Conditional Search-Space Problem (CSSP). Other terminologies can be found both for this type of variables (dimensional variables,[34,42] meta variables[6]) and for the resulting optimization problem (Variable Size Design Space Problem[1,42]). Complex design problems may include other types of variables: continuous, integer and categorical variables. Continuous variables are the most used in design processes and refer to standard quantities such as pressure, mass, length, *etc*. Integer variables are related to discrete, quantitative choices

DOI: 10.13009/EUCASS2023-834

that can be ordered (*e.g.*, number of engines). On the contrary, categorical variables are related to discrete choices with an unordered structure (*e.g.*, material choice, for instance composite or aluminum alloy). Note that a conditional variable can be either integer or categorical (cases where conditional variables are continuous exist but they are not treated here).

Several contributions to the field of CSSP[1,34,42,46] have been published, especially in the machine learning community.[6,53] MDAO brings additional challenges (different disciplines may involve different models and codes, interdisciplinary couplings, *etc*). This has been for instance considered in recent works[13,14,42] that have analyzed the impact of conditional technological choices in the context of MDAO. In addition, another important challenge is that the presence of conditional variables may importantly impact the interdisciplinary coupling relationships, that may cause a reformulation of the design problem.

This paper aims to connect both topics by investigating how classical MDAO architectures can be adapted to deal with CSSP. Two MDAO architectures are under focus: *Multidisciplinary Feasible* (MDF) and *Individual Discipline Feasible*[36] (IDF). They belong to the monolithic architecture category that solves the design problem by formulating it as a single optimization problem. The main contributions of this work are to formulate both MDAO architectures in a CSSP context, to propose a relaxation-based methodology to solve a certain class of the resulting problems and to analyze the outcomes. To evaluate their abilities to deal with CSSP, the performances of the MDAO architectures are compared to the classical approach that consists in optimizing all the possible combinatorial choices. The parameters of merit are: computational efficiency, optimality, robustness, convergence properties and number of variables and constraints to handle.

The developed method is implemented for a LRE design problem with three disciplines: thrust chamber, feedsystem and cooling. The engine optimized in this study is a LOX-LH2 LRE where the cycle's choice (expander bleed or gas generator) and the number of turbines (1 or 2) are conditional variables.

In Section 2, a brief review about CSSP and the main existing solving strategies are presented. In Section 3, an extension of the CSSP formulation to MDAO problems is proposed. Moreover, a relaxation-based methodology to solve CSSP within the MDAO framework is developed and the associated mathematical formulation is given. The additional challenges when dealing with CSSP in the MDAO context are also described. In Section 4, the LRE design problem is presented (the main models are introduced and validated in Appendix). Numerical simulations are performed in Section 5 to compare the different techniques and assess the performances of the relaxation-based method. Finally, the results are discussed, as well as the advantages and drawbacks of the relaxation-based methodology.

2. Conditional Search Space Problem

2.1 Introduction to CSSP

Modern engineering design problems require several types of variables to account for different quantities in their mathematical formulation. The classical mixed-variable optimization problem deals for instance with continuous and integer variables. Lucidi *et al.*[34] formalized a more general class of problems by introducing two other types of variables: the categorical variables and the dimensional variables. Dimensional variables are integer variables which induce a structural change in the optimization problem by, depending on their values, modifying the number of design variables or the number of constraints.

In the field of space engineering, similar problems have been treated. In those works, the problem is called Variable-Size Design Space Problem (VSDSP) instead of CSSP to refer to the changing number of variables and constraints. This terminology can be misleading and will be discussed in the following. Abdelkhalik[1] and Nyew *et al.*[38] solved the multiple gravity assist trajectory optimization problem in which the variable structure of the problem comes from the number of the swing-by maneuvers performed. The multi-stage launch vehicle design has also been dealt with by Pelamatti *et al.*[42] in a MDAO context, where the dimensional variables impact the number of stages (2 or 3), the type of propulsion (solid or liquid) for each stage as well as the acting constraints. This last example extends the definition of dimensional variables first proposed by Lucidi *et al.*[34] by allowing them to be categorical.

The Machine Learning (ML) community has faced equivalent problems. The process of choosing a learning algorithm and tuning its associated hyperparameters (*e.g.* number of layers and nodes in a neural network) to better fit a given set of data is usually solved by hand. Therefore, numerous works in the literature proposed automated approaches to perform this task. For instance, once a learning algorithm is chosen, its hyperparameters must be tuned to get an optimal model. Such a process is called hyperparameters optimization (HPO) and reviews exist to summarize the topic.[53] An analogy is made by Yang *et al.*[53] showing that HPO is not different from traditional optimization: the error rate or the root mean squared error are classical objective functions to be minimized in HPO. Usual constraints are also presented. HPO is a typical CSSP because sometimes, hyperparameters are set to a value that activates other hyperparameters which need to be tuned, as illustrated by Audet *et al.*[6] In that example, the optimization algorithm of

DOI: 10.13009/EUCASS2023-834

a neural network is a hyperparameter that can take two distinct values, activating or not two mutually exclusive sets of three hyperparameters. In that work, the algorithm choice is called a meta variable. The HPO literature also refers to conditional variables (or conditional hyperparameters)[53] because they condition the action of other variables and constraints on the problem. Those two terms are analog to what has been called dimensional variable earlier.

Following the previous example, it can be seen that a conditional variable can modify the structure of the problem by changing the acting variables without modifying its size. Thus, Variable-Size Design Space Problem can be a misleading term because the size might not vary while the structure does.

When the choice of the learning algorithm (model selection problem[43]) is itself a hyperparameter, a new class of problems arises where the goal is to select the algorithm and tune at the same time its hyperparameters. The problem is called Combined Algorithm Selection and Hyperparameter optimization (CASH) and has been reviewed by Elshawi *et al.*[17] The algorithm Auto-WEKA[28] is an example of CASH solver. The challenges associated with CSSP are well illustrated in this field in the sense that hierarchical choices have to be made: choices about some hyperparameters are nonsense if the algorithm they are associated with has not been selected previously. The CSSP covers more general problems as well. For instance, general algorithm configuration[46] keeps the same principle as HPO or CASH but extends the methods to any algorithm (mixed-integer solvers for instance[24]).

Finally, Audet *et al.*[6] proposed a general formulation of the CSSP with a classification of the variables depending on their types, their roles and their conditional dependencies. Their problem statement is based on one hypothesis: the conditional variables are always active. In other words, a high-level conditional variable can not activate or deactivate another one at a lower level in the decision tree. The limits of this hypothesis will be discussed in the next section.

2.2 State-of-the-art of the solution strategies

In the previous subsection, the main aspects of the CSSP have been discussed. This paragraph gives an overview of existing solution strategies gathered in categories. More complete reviews, to which the reader is referred, have been published in the literature.[6,46,53]

2.2.1 Model-free methods

The grid search[2] and random search[10] model-free techniques are commonly used and easy to implement. The first one uses factorial design of experiments to target interesting regions of the search space and then refines the grid around those. It becomes less efficient as the number of optimization variables grows. The second one may perform better but there is no process that drives the method to evaluate the most interesting regions, wasting time in the poor performing areas of the search space.[53] Besides, Lucidi *et al.*[34] developed a procedure to solve the problem without derivatives by alternating a local search for the continuous variables and a local search for the discrete ones. The derivative-free continuous search can be performed by a direct-search as proposed by Audet *et al.*[6] The *MADS* algorithm[5] is a suitable option to tackle this step.

2.2.2 Model-based methods

To improve efficiency, model-based techniques are available to focus on interesting regions of the design space according to the model responses. Probabilistic and deterministic methods have been studied.

Bayesian optimisation (BO): BO is the one of the most used probabilistic techniques to solve CSSP. Its principle is to build a surrogate model of the objective function and of the constraints based on an initial Design of Experiment (DoE). Then, an active learning function is optimized within an auxiliary optimization problem (involving the surrogate models) to identify the most promising solution regarding the CSSP. This solution is evaluated afterward on the exact functions (objective and constraints). The DoE and the surrogate model are finally updated with the exact values and a new iteration is started. BO strategies are well suited to the ML community as the training of a ML model can be computationally intensive: with this approach, only interesting ML model configurations are trained. Several BO approaches exist and their differences come from the surrogate model used. Bayesian optimization employing Gaussian processes (BO-GP)[44] is a well performing method to solve continuous optimization problem with expensive black-box functions. However, adjustments have been made to also deal with integer and categorical variables[41,45] as well as conditional variables for HPO,[48] CASH[32] and engineering problems, especially launchers design.[42] Nevertheless, such approaches become less efficient when dealing with high-dimensional search space (curse of dimensionality) which arise when facing CSSP or when the combinatorial induced by the conditional variables grows. Hence, several works improved this aspect by exploiting the tree-like dependency structure of the CSSP.[25,35] While BO-GP needed adaptations to fit mixed-integer and CSSPs' frameworks, BO with other types of surrogate models have been designed

DOI: 10.13009/EUCASS2023-834

purposely such as random forests[24] and tree-structured parzen estimator,[9] which are particularly adapted to handle conditional variables.

Evolutionary algorithms: In the category of metaheuristics, evolutionary algorithms have been abundantly used to solve CSSPs because they are adapted to deal with discrete variables. Particle Swarm Optimization[18] (PSO) have for instance been implemented for HPO.[16] However, a proper initialization is required and limits the performances of PSO. Genetic Algorithms[40] (GA) are a popular choice and several versions have been developed to allow the exploration of a conditional search space. The hidden-gene GA[1] considers all the possible combinations of optimization variables to build a problem where the conditional dependencies have been removed to use the classical operations in evolutionary programming. Variables are either active or inactive (part of the hidden gene) during the optimization process. The structured chromosome algorithm[38] proposes a framework where the hierarchical dependencies are kept in the evolution process. Finally, the gender-based algorithm[3] implements the nature-inspired concept of genders to adapt the classical evolutionary operations to conditional dependencies. However, evolutionary algorithms require a large number of function evaluations (objective and constraints). Hence, their applicability to computationally expensive problems is limited.

Gradient-descent algorithms: Deterministic model-based methods are also available. Lucidi *et al.*[33] developed a procedure to solve the problem with a gradient-based approach to perform the local search for the continuous variables and applied it to a large-scale unconstrained problem. The discrete ones are tackled by the same process as the derivative-free version.[34] The main limitations of gradient-based approaches are that differentiability and continuity properties are needed. Furthermore, gradient-based approaches provide local optima. These reasons explain the limited applicability of gradient-based approaches to the field of CSSP. One contribution of this paper is to provide a relaxation-based method to use gradient-descent algorithm for solving a certain class of CSSPs in a MDAO context.

3. Conditional Search-Space Problem within the MDAO framework

3.1 Problem description and formulation

The first contribution of this paper is to adapt the MDAO framework to deal with CSSP. This subsection proposes a formulation linking both topics. MDAO shares similarities with classical optimization problems. The goal is to minimize an objective function $f(\cdot)$ by iteratively modifying a vector of n design variables $\mathbf{x} \in \mathcal{X}$, while respecting a set of m design constraints $\mathbf{c}(\cdot)$. Note that in this paper, the design constraints are written as inequalities for conciseness, but they can also represent equalities. In a multidisciplinary framework, several disciplines (or subsystems) are involved in the problem. The design variables and constraints can thus be sorted according to the impacted discipline(s). In the classical MDAO notation,[36] a quantity shared by several disciplines is noted $(\cdot)_0$, whereas $(\cdot)_i$ ($i \in \{1, \dots, N\}$, where N is the number of disciplines) refers to a quantity attached to a single discipline. Mathematically, a discipline is a mapping between inputs and outputs. The latter are called coupling variables $\mathbf{y}$ because they model interdisciplinary relationships and are continuous by definition (see Fig. 1a). Indeed, sometimes, a discipline j needs the output $\mathbf{y}_i$ of a discipline i to perform its disciplinary analysis (meaning to compute its own outputs). One way of providing $\mathbf{y}_i$ to discipline j is via a variable called target variable $\mathbf{y}_i^t$, handled by the optimizer. However, to make sure that $\mathbf{y}_i^t$ and $\mathbf{y}_i$ are equal (in other words, to satisfy the couplings), an additional consistency constraint is imposed $\mathbf{c}_i^c(\cdot)$. This MDAO architecture is called *Individual Discipline Feasible*[36] (IDF) and yields:

$$
\begin{aligned}
\text{minimize: } & f(\mathbf{x},\ \mathbf{y}(\mathbf{x},\ \mathbf{y}^t)) \\
\text{with respect to: } & \mathbf{x},\ \mathbf{y}^t \\
\text{subject to: } & \mathbf{c}_0(\mathbf{x},\ \mathbf{y}(\mathbf{x},\ \mathbf{y}^t)) \leq 0 \\
& \mathbf{c}_i(\mathbf{x}_0,\ \mathbf{x}_i,\ \mathbf{y}_i(\mathbf{x}_0,\ \mathbf{x}_i,\ \mathbf{y}^t_{j\neq i})) \leq 0, \quad i, j \in \{1, \dots, N\} \\
& \mathbf{c}_i^c \ = \ \mathbf{y}_i^t - \mathbf{y}_i(\mathbf{x}_0,\ \mathbf{x}_i,\ \mathbf{y}^t_{j\neq i}) = 0, \quad i, j \in \{1, \dots, N\}
\end{aligned}
\tag{1}
$$

Another way of satisfying the couplings is to estimate them for each value of the design variables (hence, at each iteration of the optimization process) via an auxiliary analysis called multidisciplinary analysis (MDA). MDA consists in solving a non-linear system of equations using dedicated algorithms (*e.g.*, Gauss-Seidel or Newton-Raphson) to determine the value of the coupling variables that ensure the multidisciplinary feasibility. The associated MDAO architecture is called *Multidisciplinary Feasible*[36] (MDF). With respect to the IDF formulation, some simplifications

DOI: 10.13009/EUCASS2023-834

can be made:

$$\begin{aligned} \text{minimize: } & f(\mathbf{x},\ \mathbf{y}(\mathbf{x})) \\ \text{with respect to: } & \mathbf{x} \\ \text{subject to: } & \mathbf{c}_0(\mathbf{x},\ \mathbf{y}(\mathbf{x})) \leq 0 \\ & \mathbf{c}_i(\mathbf{x}_0,\ \mathbf{x}_i,\ \mathbf{y}(\mathbf{x}_0,\ \mathbf{x}_i,\ \mathbf{y}_{j\neq i})) \leq 0,\quad i,j \in \{1,\ldots,N\} \end{aligned} \tag{2}$$

Indeed, as at each iteration of the optimizer, the coupling variables satisfying the interdisciplinary system of equations are determined (denoted by $\mathbf{y}(\mathbf{x})$ in the formulation), thus the consistency constraints $\mathbf{c}_i^c$ and the target variables $\mathbf{y}^t$ are removed from the IDF formulation. The system-level optimizer only controls the design variables. Thus, IDF has a greater dimensionality in terms of optimization variables and constraints to handle, whereas MDF has to solve an iterative process at each optimization step.

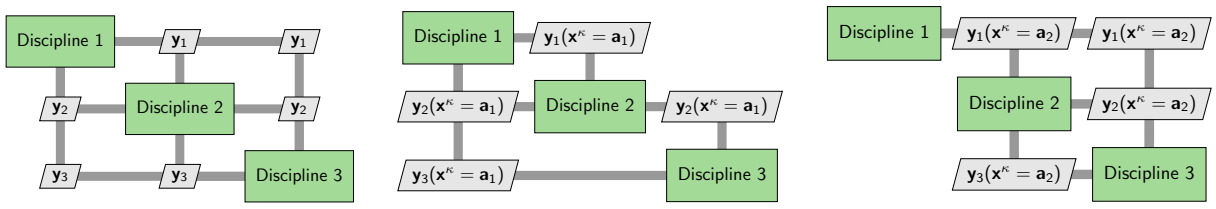
Inspired from the general single-discipline CSSP formulation,[6] the aforementioned architectures can be adapted to the multidisciplinary CSSP by highlighting the conditional dependencies of the problem. The conditional variables are written $\mathbf{x}^\kappa = [\mathbf{x}^{\kappa_1}, \ldots,\ \mathbf{x}^{\kappa_l}]$, where l is the number of hierarchy levels. Indeed, engineering problems often require to model several levels of hierarchy, expressing technological decisions. For instance, in the LRE design problem, let the choice of the feed system type (pressure-fed or turbopump-fed) be a scalar conditional variable x^{κ_1}. Let x^{κ_2} be another scalar conditional variable modeling the number of turbines (relevant in case of a turbopump-fed system exclusively). Thus, the variable x^{κ_1} is higher in the hierarchy because if x^{κ_1} = pressure-fed, x^{κ_2} does not act on the problem. Mathematically, the conditional variable vector at the i^{th} level $\mathbf{x}^{\kappa_i} \in \mathcal{X}^{\kappa_i}$ depends on all the higher-levels $\mathbf{x}^{\kappa_j},\ j \in \{1,\ldots,i-1\}$. In that sense, the adopted formulation differs from the one proposed by Audet *et al.*[6] When defining the conditional variables (called meta variables in their work), the assumption that conditional variables are always active has been made: a high-level conditional variable can not activate or deactivate another one at a lower level in the hierarchy. Nevertheless, the standard design variables $\mathbf{x}^s \in \mathcal{X}^s(\mathbf{x}^\kappa)$ in Eq. (3) are analog to the ones defined by Audet *et al.*[6] : they gather the continuous design variables $\mathbf{x}^c \in \mathbb{R}^{n_c(\mathbf{x}^\kappa)}$, the integer variables $\mathbf{x}^z \in \mathbb{Z}^{n_z(\mathbf{x}^\kappa)}$, but also include the categorical ones $\mathbf{x}^q \in \mathbb{Z}^{n_q(\mathbf{x}^\kappa)}$ (this notation[6] is possible because a bijection exists between the possible categories of $\mathbf{x}^q$ and $\mathbb{Z}^{n_q(\mathbf{x}^\kappa)}$). Their respective numbers are $n_c(\mathbf{x}^\kappa)$, $n_z(\mathbf{x}^\kappa)$ and $n_q(\mathbf{x}^\kappa)$. The design variables definition set is:

$$\mathcal{X} = \{[\mathbf{x}^\kappa,\ \mathbf{x}^s]\ :\ \mathbf{x}^{\kappa_1} \in \mathcal{X}^{\kappa_1};\quad \forall i \in \{2,\ldots,l\},\ \mathbf{x}^{\kappa_i} \in \mathcal{X}^{\kappa_i}(\mathbf{x}^{\kappa_1},\ldots,\mathbf{x}^{\kappa_{i-1}});\quad \mathbf{x}^s \in \mathcal{X}^s(\mathbf{x}^\kappa)\} \tag{3}$$

Let $n_{y,i}$ and $n_{y,i}^t$ be the numbers of coupling and target variables for a discipline i. The conditional dependencies for the coupling and target variables are formalized as:

$$\forall i \in \{1,\ldots,N\},\quad \mathbf{y}_i \in \mathcal{Y}_i(\mathbf{x}^\kappa) \subseteq \mathbb{R}^{n_{y,i}(\mathbf{x}^\kappa)} \text{ and } \mathbf{y}_i^t \in \mathcal{Y}_i^t(\mathbf{x}^\kappa) \subseteq \mathbb{R}^{n_{y,i}^t(\mathbf{x}^\kappa)} \tag{4}$$

This is a novelty brought by the multidisciplinarity of the problem with respect to the single-discipline CSSP: the interdisciplinary couplings structure is a new challenge to handle because, for instance, two disciplines can be coupled in different ways depending on conditional variables. Figs. 1b and 1c illustrate a case where $\mathbf{x}^\kappa \in \{\mathbf{a}_1, \mathbf{a}_2\}$.



(a) Generic couplings structure (b) $\mathbf{x}^\kappa = \mathbf{a}_1$ (c) $\mathbf{x}^\kappa = \mathbf{a}_2$

Figure 1: Generic coupling structure and illustration of conditional dependencies (generated with *pyXDSM*[29])

Let $m_{c,i}$ be the number of consistency constraints for a discipline i (in a IDF formulation). The following relationship yields: $m_{c,i} = n_{y,i}^t \leq n_{y,i}$. Finally, the objective function and the constraints are:

$$f : \mathcal{X} \times \mathcal{Y}_1(\mathbf{x}^\kappa) \times \cdots \times \mathcal{Y}_N(\mathbf{x}^\kappa) \longrightarrow \mathbb{R} \tag{5}$$

$$\mathbf{c}_0 : \mathcal{X} \times \mathcal{Y}_1(\mathbf{x}^\kappa) \times \cdots \times \mathcal{Y}_N(\mathbf{x}^\kappa) \longrightarrow \mathbb{R}^{m_0(\mathbf{x}^\kappa)} \tag{6}$$

$$\mathbf{c}_i : \mathcal{X} \times \mathcal{Y}_i(\mathbf{x}^\kappa) \longrightarrow \mathbb{R}^{m_i(\mathbf{x}^\kappa)},\quad i,j \in \{1,\ldots,N\} \tag{7}$$

$$\mathbf{c}_i^c : \mathcal{Y}_i^t(\mathbf{x}^\kappa) \times \mathcal{Y}_i(\mathbf{x}^\kappa) \longrightarrow \mathbb{R}^{m_{c,i}(\mathbf{x}^\kappa)},\quad i,j \in \{1,\ldots,N\} \tag{8}$$

DOI: 10.13009/EUCASS2023-834

The IDF formulation with conditional dependencies is:

$$\begin{aligned}
\text{minimize: } & f(\mathbf{x}^\kappa,\ \mathbf{x}^s,\ \mathbf{y}(\mathbf{x}^\kappa,\ \mathbf{x}^s,\ \mathbf{y}^t)) \\
\text{with respect to: } & \mathbf{x}^\kappa,\ \mathbf{x}^s,\ \mathbf{y}^t \\
\text{subject to: } & \mathbf{c}_0(\mathbf{x}^\kappa,\ \mathbf{x}^s,\ \mathbf{y}(\mathbf{x}^\kappa,\ \mathbf{x}^s,\ \mathbf{y}^t)) \leq 0 \\
& \mathbf{c}_i(\mathbf{x}_0^\kappa,\ \mathbf{x}_i^\kappa,\ \mathbf{x}_0^s,\ \mathbf{x}_i^s,\ \mathbf{y}_i(\mathbf{x}_0^\kappa,\ \mathbf{x}_i^\kappa,\ \mathbf{x}_0^s,\ \mathbf{x}_i^s,\ \mathbf{y}_{j\neq i}^t)) \leq 0,\ \ i,j \in \{1,\dots,N\} \\
& \mathbf{c}_i^c\ =\ \mathbf{y}_i^t - \mathbf{y}_i(\mathbf{x}_0^\kappa,\ \mathbf{x}_i^\kappa,\ \mathbf{x}_0^s,\ \mathbf{x}_i^s,\ \mathbf{y}_{j\neq i}^t) = 0,\ \ i,j \in \{1,\dots,N\}
\end{aligned} \tag{9}$$

The MDF is analog:

$$\begin{aligned}
\text{minimize: } & f(\mathbf{x}^\kappa,\ \mathbf{x}^s,\ \mathbf{y}(\mathbf{x}^\kappa,\ \mathbf{x}^s)) \\
\text{with respect to: } & \mathbf{x}^\kappa,\ \mathbf{x}^s \\
\text{subject to: } & \mathbf{c}_0(\mathbf{x}^\kappa,\ \mathbf{x}^s,\ \mathbf{y}(\mathbf{x}^\kappa,\ \mathbf{x}^s)) \leq 0 \\
& \mathbf{c}_i(\mathbf{x}_0^\kappa,\ \mathbf{x}_i^\kappa,\ \mathbf{x}_0^s,\ \mathbf{x}_i^s,\ \mathbf{y}_i(\mathbf{x}_0^\kappa,\ \mathbf{x}_i^\kappa,\ \mathbf{x}_0^s,\ \mathbf{x}_i^s,\ \mathbf{y}_{j\neq i})) \leq 0,\ \ i,j \in \{1,\dots,N\}
\end{aligned} \tag{10}$$

3.2 Relaxation-based solution strategy

The second contribution of this paper is to propose a relaxation-based framework to convert a certain class of multidisciplinary CSSP into a regular MDAO problem which can be formulated with classical methods such as MDF and IDF and solved with gradient-based algorithms. In order to use such algorithms, several conditions are necessary. First, design variables must be continuous. Then, the disciplinary models shall have continuity and differentiabilty properties because the aim is to use the partial derivatives of their outputs with respect to their inputs to compute the necessary gradients for the optimization algorithm. Several methods exist to build those gradients from the partial derivatives[20] (the chain rule is one of the most used). The idea that is presented in this subsection is to relax the discrete conditional variables with continuous ones ranging from 0 to 1. After relaxation, if the standards variables $\mathbf{x}^s$ are all continuous (and the necessary conditions are met), gradient-based optimizers are usable. If discrete standard variables are present (integer or categorical), mixed-integer programming techniques[37] must be employed (this case is not treated here).

In this subsection, the non-hierarchical case where conditional variables are always active is treated. In the following paragraphs, the consequences of the relaxation for the optimization variables, the constraint functions, the coupling variables and the objective function are introduced. Note that $\hat{\cdot}$ means that a quantity defined in the previous subsection has been modified to fit the relaxation-based method.

Conditional variables: To relax a scalar conditional variable x^κ with p possible values $a_i,\ i \in \{1,\dots,p\}$, a method to convert one discrete variable with a finite number of values into several binary ones[37] can be adapted: p continuous scalar variables ω_{a_i} varying in $[0,1]$ are defined (instead of p binary ones). They will play the role of the conditional variable x^κ in the relaxed problem. At the convergence, to retrieve a solution with a physical meaning, they shall verify:

$$\forall i \in \{1,\dots,p\},\ \omega_{a_i} = \delta_{a_i x^\kappa} \tag{11}$$

where $\delta_{a_i x^\kappa}$ is the Kronecker's symbol ($\delta_{a_i x^\kappa} = 1$ if $x^\kappa = a_i$, and is null otherwise). To verify this relationship at convergence, the following constraints are imposed:

$$c^a(\boldsymbol{\omega}) = \sum_{i=1}^{p} \omega_{a_i} - 1 = 0 \ \text{ and } \ \boldsymbol{\omega} = \left[\omega_{a_1},\dots,\omega_{a_p}\right] \tag{12}$$

$$\forall i \in \{1,\dots,p\},\ \ c_i^\omega(\omega_{a_i}) = 1 + \cos(\pi(2\omega_{a_i}+1)) = 0 \tag{13}$$

The constraint (12) prevents the method to select several values of the same conditional variable. The constraint (13) allows to retrieve binary values for the ω_{a_i} to get a physical solution. The advantage of formulating constraint (13) with a cosine is to stay in the continuous domain where an analytic gradient can be defined. The main drawback is that the sinusoidal behavior makes it difficult to satisfy and can lead to numerical instabilities.

The case where x^κ has two possible values a_1 and a_2 is a particular case where the constraint (12) is imposed in the definition of the relaxation variable: $\omega = 0$ if $x^\kappa = a_1$ and $\omega = 1$ if $x^\kappa = a_2$ (one variable and one constraint (13) are added to the problem instead of two variables $\omega_{a_1},\ \omega_{a_2}$, one constraint (12) and two constraints (13)).

This continuous encoding of the conditional variables can be applied to all the component of $\mathbf{x}^\kappa$ and allows to deal with continuous quantities instead of discrete ones. Hence, the vector $\mathbf{x}^\kappa$ is removed from the problem and is replaced by a vector $\boldsymbol{\omega}$ that gathers all the variables ω defined before. The size of $\boldsymbol{\omega}$ is fixed and is named n_ω. Let

DOI: 10.13009/EUCASS2023-834

m_a and m_ω be the numbers of additional constraints (12) and (13) that have to be imposed. Thus, m_a is the number of conditional variables with strictly more than two possible values and $m_\omega = n_\omega$ is the number of relaxed variables ω that are added to the problem to replace conditional variables. It can be seen that the relaxed problem will become quickly over-constrained as the number of conditional variables grows.

Standard variables and targets: The new standard variables vector $\hat{\mathbf{x}}^s$ is the concatenated vector of all the possible standard design variables. For example, assume that $\mathbf{x}^\kappa$ is one-dimensional (and $x^\kappa \in \{a_1, a_2, a_3\}$). Consider the following situation:

$$\begin{cases} \mathbf{x}^s(x^\kappa = a_1) = [x_1,\ x_2]\,, & n_s(x^\kappa = a_1) = 2 \\ \mathbf{x}^s(x^\kappa = a_2) = [x_2,\ x_4]\,, & n_s(x^\kappa = a_2) = 2 \\ \mathbf{x}^s(x^\kappa = a_3) = [x_3,\ x_4,\ x_5]\,, & n_s(x^\kappa = a_3) = 3 \end{cases} \tag{14}$$

where $n_s(x^\kappa) = n_c(x^\kappa) + n_z(x^\kappa) + n_q(x^\kappa)$ is the number of standard variables. By applying the aforementioned method:

$$\hat{\mathbf{x}}^s = [x_1,\ x_2,\ x_3,\ x_4,\ x_5]\,,\ \hat{n}_s = 5 \text{ and } \boldsymbol{\omega} = \left[\omega_{a_1},\ \omega_{a_2},\ \omega_{a_3}\right],\ n_\omega = 3\,. \tag{15}$$

In the case of the IDF architecture, the same process is applied to target coupling variables that may depend on conditional variables in case the couplings are different from one architecture to another (see Figs. 1 and 3).

Constraints: Let $c : C \longrightarrow \mathbb{R}$ be a generic scalar constraint function. Note that C can be any input sets defined in Eqs. (6) to (8). For conciseness, the potential dependencies on the standard design variables, the target and the coupling variables are not shown. Let $c(\cdot)$ depend on a conditional variable $\mathbf{x}^\kappa$. Assuming again that $\mathbf{x}^\kappa$ is scalar (and $x^\kappa \in \{a_i,\ i \in \{1, \dots, p\}\}$), the new constraint is defined as the weighted average of all the possible constraints $c(\cdot)$ with respect to the weights ω_{a_i}:

$$\hat{c}(\boldsymbol{\omega}) = \sum_{i=1}^{p} \omega_{a_i}\, c(x^\kappa = a_i) \leq 0 \text{ and } \boldsymbol{\omega} = \left[\omega_{a_1}, \dots,\ \omega_{a_p}\right] \tag{16}$$

Several particular cases have to be discussed. In the case where a constraint $c(\cdot)$ is not impacted by any conditional variable (in the work of Audet *et al.*,[6] those constraints are called "global"), the previous process is not necessary, the constraint $c(\cdot)$ can be left unmodified. Furthermore, the constraint $c(\cdot)$ may not be active for certain values of x^κ. If this is the case, for instance for $x^\kappa = a_p$, the new constraint is constructed to be always satisfied when $x^\kappa = a_p$ by imposing $c(x^\kappa = a_p) = 0$:

$$\hat{c}(\boldsymbol{\omega}) = \sum_{i=1}^{p-1} \omega_{a_i}\, c(x^\kappa = a_i) \leq 0 \text{ and } \boldsymbol{\omega} = \left[\omega_{a_1}, \dots,\ \omega_{a_p}\right] \tag{17}$$

Then, if x^κ has only two possible values a_1 or a_2, the constraint becomes:

$$\hat{c}(\omega) = (1 - \omega)\, c(x^\kappa = a_1) + \omega\, c(x^\kappa = a_2) \leq 0 \tag{18}$$

Let $c(\cdot)$ depends a two-dimensional conditional variable $\mathbf{x}^\kappa = (x_1^\kappa,\ x_2^\kappa)$, where x_1^κ and x_2^κ have p_1 and p_2 possible values noted $\{a_i,\ i \in \{1, \dots, p_1\}\}$ and $\{b_i,\ i \in \{1, \dots, p_2\}\}$ respectively. The process is repeated:

$$\hat{c}(\boldsymbol{\omega}) = \boldsymbol{\omega}_1^\top C \boldsymbol{\omega}_2 \tag{19}$$

where: $\boldsymbol{\omega}_1 = [\omega_{a_1}, \dots, \omega_{a_{p_1}}]$, $\boldsymbol{\omega}_2 = [\omega_{b_1}, \dots, \omega_{b_{p_2}}]$ and $\forall i \in \{1, \dots, p_1\}, \forall j \in \{1, \dots, p_2\},\ C_{i,j} = c(\mathbf{x}^\kappa = [a_i, b_j])$.

Finally, the method can be generalized to $\mathbf{x}^\kappa$ with any dimension.

Objective function and coupling variables: The coupling variables are treated exactly the same way as the constraints: all their possible values (with respect to the conditional variables) are averaged and weighted by the corresponding relaxed variables ω to define $\hat{\mathbf{y}}$. However, this part of the method might be problematic in a MDF frame on particular applications. Indeed, in the case where a discipline uses a specific code (a computational fluid dynamics solver for instance), the latter might be unable to deal with averaged coupling variables coming from the other disciplines, causing the MDA to fail. If this is the case, one MDA per conditional variable combination has to be performed, limiting greatly the applicability of the method.

Concerning the objective function $f(\cdot)$, two cases are possible. If $f(\cdot)$ depends only on coupling variables $\mathbf{y}$, no further transformations are needed: the conditional dependencies have already been relaxed by definition of $\hat{\mathbf{y}}$. If conditional dependencies remain, meaning that $f(\cdot)$ still depends on $\mathbf{x}^\kappa$ even after the relaxation of the coupling variables, the same process explained earlier can be applied.

DOI: 10.13009/EUCASS2023-834

Final formulation: The relaxed IDF formulation becomes:

$$\begin{aligned}
\text{minimize: } & \hat{f}(\boldsymbol{\omega},\ \hat{\mathbf{x}}^s,\ \hat{\mathbf{y}}(\boldsymbol{\omega},\ \hat{\mathbf{x}}^s,\ \hat{\mathbf{y}}^t)) \\
\text{with respect to: } & \boldsymbol{\omega},\ \hat{\mathbf{x}}^s,\ \hat{\mathbf{y}}^t \\
\text{subject to: } & \hat{\mathbf{c}}_0(\boldsymbol{\omega},\ \hat{\mathbf{x}}^s,\ \hat{\mathbf{y}}(\boldsymbol{\omega},\ \hat{\mathbf{x}}^s,\ \hat{\mathbf{y}}^t)) \leq 0 \\
& \hat{\mathbf{c}}_i(\boldsymbol{\omega}_0,\ \boldsymbol{\omega}_i,\ \hat{\mathbf{x}}_0^s,\ \hat{\mathbf{x}}_i^s,\ \hat{\mathbf{y}}(\boldsymbol{\omega}_0,\ \boldsymbol{\omega}_i,\ \hat{\mathbf{x}}_0^s,\ \hat{\mathbf{x}}_i^s,\ \hat{\mathbf{y}}_{j\neq i}^t)) \leq 0,\ \ i,j \in \{1,\dots,N\} \\
& \hat{\mathbf{c}}_i^c\ =\ \hat{\mathbf{y}}_i^t - \hat{\mathbf{y}}_i(\boldsymbol{\omega}_0,\ \boldsymbol{\omega}_i,\ \hat{\mathbf{x}}_0^s,\ \hat{\mathbf{x}}_i^s,\ \hat{\mathbf{y}}_{j\neq i}^t,) = 0,\ \ i,j \in \{1,\dots,N\} \\
& \mathbf{c}_k^a = 0,\ \ k \in \{1,\dots,m_a\} \\
& \mathbf{c}_k^\omega = 0,\ \ k \in \{1,\dots,m_\omega\}
\end{aligned} \tag{20}$$

The relaxed MDF formulation is not reported here but is analog to the IDF, where the target variables and the consistency constraints are removed. Using this method, the CSSP is relaxed into a mixed-variable nonlinear programming problem. If the standard variables $\mathbf{x}^s$ are continuous, the CSSP becomes a nonlinear programming problem that can be solved by a gradient descent algorithm (if the models are differentiable), as illustrated in the following sections.

4. Application case: optimization of a liquid rocket engine

To evaluate the method previously described, a LRE optimization with two conditional variables and three disciplines/subsystems (1: thrust chamber, 2: feed system and 3: regenerative cooling) is presented. The design problem is the following: maximizing the specific impulse of an upper-stage turbopump-fed engine in vacuum for a given thrust (the LE-5B engine[26] is selected as baseline: T_{vac} = 137.3 kN). Two open cycles are considered: Expander Bleed (EB) and Gas Generator (GG). The objective function is the specific impulse I_{sp} which, for an open cycle system, yields:

$$f(\mathbf{y}_1, \mathbf{y}_2) = I_{sp} = (1 - X)I_{sp,main} + XI_{sp,sec} \tag{21}$$

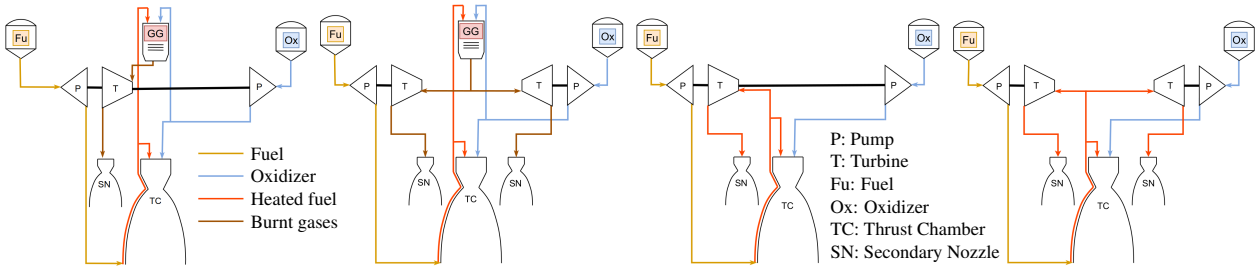
where $I_{sp,main}$ and $I_{sp,sec}$ are respectively the specific impulses of the main thrust chamber and of the secondary system (the flow that goes through the turbines and is expanded in a secondary nozzle). X is the ratio between the secondary mass flow rate and the total one.

The engine can have 4 different architectures that are illustrated on Fig. 2. Those engine configurations differ by their cycle, EB or GG, and their number of turbines, 1 or 2. Hence, two conditional variables $\mathbf{x}^\kappa = [x_1^\kappa, x_2^\kappa]$ encapsulate the choice of architecture among the four available. The first one x_1^κ is the cycle's choice and can take two values: GG or EB. The second one x_2^κ is the number of turbines and also have two possible values: 1 or 2. Additionally three physical constraints are considered: a 2000 kg maximum engine mass, a 1500 K maximum nozzle wall temperature at the throat and a 800 K maximum temperature at the turbine inlet. This last constraint concerns only the GG because the typical temperatures T_{GG} reached at the turbine inlet for those cycles can be problematic for the turbine material. The engine mass M_{en} is defined as the sum of the following masses: combustion chamber M_{CC}, nozzle M_N, turbopump(s) M_{tp}, fuel tank $M_{t,f}$ and oxidizer tank $M_{t,o}$. In the case of a GG architecture, the gas generator mass M_{GG} is added. To test efficiently the method, simplified models are considered (see appendix). In particular, in the regenerative cooling system model, only the behavior at the throat (where the heat flux is the highest) is modeled. This is the reason why the constraint on the nozzle wall temperature $T_{wg,t}$ is imposed at the throat. Furthermore, a friction model has not been developed for the flow in the cooling channels: a constraint of 100 m/s on the coolant flow speed u_l is added considering that in realistic configurations, the speed of the coolant will be constrained by pressure losses in the channels. All the constraints are gathered in Table 1. The problem also counts eight standard design variables $\mathbf{x}^s$ (that are continuous), and eight target variables (hence eight consistency constraints) in case of an IDF formulation. All the variables are described in Table 1. One of the coupling variable (that becomes a target in IDF) is the fuel total temperature at the turbine inlet $T_{0,f}$ computed by the cooling discipline. This coupling is only active when an EB is selected. In the case of a GG, the turbine inlet temperature is the combustion temperature in the gas generator T_{GG} and is directly computed within the discipline feed system.

By applying the method described in the last section, the problem can be relaxed and two relaxation variables are added: ω_1 relaxes the cycle choice x_1^κ ($\omega_1 = 1$ if an Expander Bleed is selected, $\omega_1 = 0$ otherwise), ω_2 relaxes the number of turbines x_2^κ ($\omega_2 = 1$ if 2 turbines are selected, $\omega_2 = 0$ otherwise). Note that the constraint Eq. (12) is already applied in the definition of the relaxation variables because both conditional variables have two possible values. However, two additional constraints Eq. (13) are imposed:

$$c_k^\omega = 1 + \cos(\pi(2\omega_i + 1)) = 0,\ \ k \in \{1, 2\} \tag{22}$$

DOI: 10.13009/EUCASS2023-834



(a) Gas generator, 1 turbine (b) Gas generator, 2 turbines (c) Expander bleed, 1 turbine (d) Expander bleed, 2 turbines

Figure 2: Engine architectures depending on the 4 possible combinations of conditional variables

Table 1: Composition of the standard design variables, target variables and design constraints vectors, their conditional dependencies (the engine architecture on which they act) and their related discipline (1: thrust chamber, 2: feed system, 3: cooling) are highlighted

Design variables		**Description**	**GG1**[1]	**GG2**[1]	**EB1**[1]	**EB2**[1]
$\mathbf{x}_0^s =$	$\begin{pmatrix} p_{CC} \\ OF_{CC} \end{pmatrix}$	pressure in the combustion chamber	◯	◯	◯	◯
		oxidizer/fuel ratio in the combustion chamber	◯	◯	◯	◯
$\mathbf{x}_2^s(\mathbf{x}^\kappa) =$	$\begin{pmatrix} OF_{GG} \\ PR \\ PR_f \\ PR_o \end{pmatrix}$	oxidizer/fuel ratio in the gas generator	◯	◯	×	×
		pressure ratio in the turbine	◯	×	◯	×
		pressure ratio in the fuel turbine	×	◯	×	◯
		pressure ratio in the oxidizer turbine	×	◯	×	◯
$\mathbf{x}_3^s =$	$\begin{pmatrix} h_t \\ w_t \end{pmatrix}$	height of the cooling channels at the throat	◯	◯	◯	◯
		width of the cooling channels at the throat	◯	◯	◯	◯
Target variables[2]		**Description**	**GG1**	**GG2**	**EB1**	**EB2**
$\mathbf{y}_1^t =$	$\begin{pmatrix} I^t_{sp,main} \\ T^t_{CC} \\ D^t_t \\ \epsilon^t \\ c^{*t} \end{pmatrix}$	specific impulse of the main thrust chamber	◯	◯	◯	◯
		combustion chamber temperature	◯	◯	◯	◯
		throat diameter	◯	◯	◯	◯
		area ratio	◯	◯	◯	◯
		characteristic velocity	◯	◯	◯	◯
$\mathbf{y}_2^t =$	$\begin{pmatrix} \dot{m}^t_{CC} \\ \dot{m}^t_{tot,f} \end{pmatrix}$	mass flow rate in the combustion chamber	◯	◯	◯	◯
		total fuel mass flow rate	◯	◯	◯	◯
$y_3^t =$	$T^t_{0,f}$	fuel total temperature at the turbine inlet	×	×	◯	◯
Design constraints		**Description**	**GG1**	**GG2**	**EB1**	**EB2**
$c_0(x_1^\kappa) =$	$M_{en} - M_{en}^{max}$	maximum engine mass[3]	◯	◯	◯	◯
$c_2(x_1^\kappa) =$	$T_{GG} - T_{GG}^{max}$	maximum temperature at the turbine inlet	◯	◯	×	×
$\mathbf{c}_3 =$	$\begin{pmatrix} T_{wg,t} - T^{max}_{wg,t} \\ u_l - u_{l,max} \end{pmatrix}$	maximum nozzle wall temperature at the throat	◯	◯	◯	◯
		maximum flow speed in the cooling channels	◯	◯	◯	◯

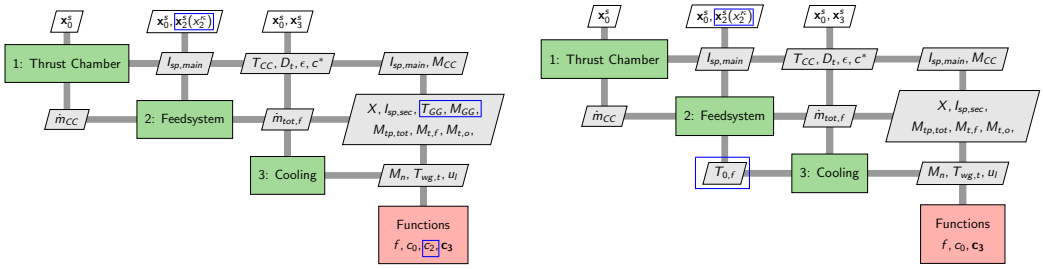
[1] GGn: gas generator with n turbines, EBn: expander bleed with n turbines, ◯: acting, ×: non-acting
[2] relevant for the IDF formulation exclusively, one consistency constraint is associated to each target variable
[3] the conditional dependency here shows that M_{GG} has to be added or not to M_{en}, depending on x_1^κ

5. Numerical results

This section presents the numerical results obtained for the LRE design problem with the four possible engine architectures (without conditional variables) in MDF, gas generator with one and two turbines (GG1 and GG2), expander bleed with one and two turbines (EB1 and EB2) and the Relaxed CSSP (RCSSP) with MDF and IDF formulations. The number of variables and constraints for each problem are summarised in Table 2.

The optimization algorithm used is the gradient-based algorithm *PSQP* from *pyOptSparse*.[52] The models have been built using the framework *openMDAO*[21] which has been purposely designed to support gradient-based optimization by efficiently computing the total derivatives of a multidisciplinary system. As gradient-descent methods provide local optima, a multi-start method is employed to generate initial guesses that cover the design space. The initial points are generated by performing a Design of Experiments (DoE) on the search domain defined by the design, the target

DOI: 10.13009/EUCASS2023-834



(a) $x_1^\kappa = GG$ (b) $x_1^\kappa = EB$

Figure 3: Variable structure of the problem with two possible values of the cycle's choice, the impacted component are highlighted with the blue boxes (generated with *pyXDSM*[29])

Table 2: Number of optimization variables (standard, target, relaxation) and constraints (coupling, relaxation, design)

Problem		Number of design variables ($n_s/n_y^t/n_\omega$)	Number of equality constraints (m_c/m_b)	Number of inequality constraints (m)
EB1	**MDF**	5 (5/0/0)	0 (0/0)	3
EB2	**MDF**	6 (6/0/0)	0 (0/0)	3
GG1	**MDF**	6 (6/0/0)	0 (0/0)	4
GG2	**MDF**	7 (7/0/0)	0 (0/0)	4
RCSSP	**MDF**	10 (8/0/2)	2 (0/2)	4
	IDF	18 (8/8/2)	10 (8/2)	4

and the relaxation variables (18 parameters in total, see Table 2). In order to compare between the methods, the same starting points are taken for each problem. The DoE is done with the latin hypercube sampling method, from the toolbox *SMT*.[12] Sometimes, a combination of initial values for the set of design variables leads to non-physical values for some coupling variables $\mathbf{y}$, crashing the numerical models. To ensure that the starting points are adapted to the models, a Multidisciplinary Analysis (MDA) is performed on each start for the RCSSP. If the MDA fails to converge, the associated point is removed from the batch of initial guesses.

The tolerance on the design and relaxation constraints' satisfaction is set to 10^{-3} for all cases. The couplings' satisfaction absolute tolerance is set to 10^{-8} for the MDF cases because in that formulation, the couplings are supposed to be satisfied at each optimization step to apply the chain rule.[20] However, the consistency constraint tolerance is set to 10^{-3} for IDF because the number of optimization variables and constraints is higher, resulting in a more complex optimization problem and thus is prone to a lower probability to find feasible points than MDF. Note that those tolerances are not ideal to fairly compare MDF and IDF, the aim is rather to show that both coupled and decoupled monolithic MDAO formulations can be employed.

Note that in the following, an iteration is an optimization step, and a disciplinary call is a step where the disciplines are given a set of inputs and return a set of outputs. Those two notions are equivalent in IDF because no MDA is performed. In Table 3, the average performances of the six cases over 100 initial guesses are presented, as well as their standard deviation. Solving the RCSSP requires more computational effort on average in terms of running time, iterations (and disciplinary calls for the MDF case) with respect to each non-conditional problem. However, using the relaxation method allows to solve them all at once instead of handling the four problems sequentially, saving computational resources. Furthermore, Tables 4 and 5 show that solving the RCSSP allows to retrieve an interesting solution in terms of specific impulse, and more importantly, the best engine architecture is obtained: a gas generator with two turbines ($\omega_1 = 0$, $\omega_2 = 1$, the additional constraints Eq. (22) are hence respected). Table 3 also presents the feasibility rate of each problem. A feasible point is defined as a point where the optimizer converges locally and satisfies the constraints. The feasibility rate shows that a GG architecture is better suited for the given mission whereas the EB has difficulties verifying the constraints (especially in the one turbine case).

Table 5 also shows that a slightly better solution could have been found (representing a gain of 0.1 s of specific impulse). This is confirmed by Table 4: GG2 and RCSSP in MDF and IDF do not converge on the same optimum. Indeed, both MDAO formulations of the RCSSP have more optimization variables and constraints to handle (see Table 2). Therefore, the design space of both RCSSPs are more complex to explore than GG2. Besides, the two sinusoidal relaxation equality constraints (22) may generate numerous local optima. Increasing the number of initial guesses (and

DOI: 10.13009/EUCASS2023-834

thus the computational effort) to better explore the design space could have solved this issue, but considering the slight difference between the optimal values of the objectives, this solution has been discarded.

Table 3: Average algorithmic performances and standard deviations over 100 initial guesses

Problem		**Running time** [s]	**Iterations** [-]	**Disciplinary calls** [-]	**Feasibility rate*** [%]
EB1	**MDF**	54.78 (± 25.70)	139 (± 76)	987 (± 477)	20
EB2	**MDF**	67.84 (± 28.64)	173 (± 89)	1074 (± 484)	38
GG1	**MDF**	55.94 (± 21.32)	157 (± 55)	888 (± 299)	80
GG2	**MDF**	69.82 (± 31.15)	200 (± 85)	1101 (± 449)	81
RCSSP	**MDF**	93.91 (± 90.76)	221 (± 196)	1380 (± 1261)	47
	IDF	75.95 (± 58.59)	452 (± 377)	452 (± 377)	25

*The feasibility rate expresses the convergence rate towards feasible points

Table 4: Optimal solution in terms of design and relaxation variables among 100 initial guesses

Problem		p_{CC} [bar]	OF_{CC} [-]	OF_{GG} [-]	PR [-]	PR_f [-]	PR_o [-]	h_t [mm]	w_t [mm]	ω_1 [-]	ω_2 [-]
EB1	**MDF**	15.000	4.097	×	50.000	×	×	60.000	1.000	×	×
EB2	**MDF**	17.106	4.090	×	×	50.000	49.187	60.000	1.148	×	×
GG1	**MDF**	51.685	3.908	0.811	50.000	×	×	33.080	1.000	×	×
GG2	**MDF**	56.599	3.882	0.811	×	50.000	44.748	31.270	1.000	×	×
RCSSP	**MDF**	54.542	3.891	0.811	19.690	50.000	48.756	31.947	1.000	0.000	1.000
	IDF	52.893	3.897	0.811	18.342	50.000	50.000	32.500	1.000	0.000	1.000

Table 5: Optimal objective and constraints among 100 initial guesses

Problem		I_{sp} [s]	M_{en} [kg]	$T_{wg,t}$ [K]	T_{GG} [K]	u_{chan} [m/s]
EB1	**MDF**	432.636	1646.863	1499.898	×	99.997
EB2	**MDF**	439.754	1587.192	1499.997	×	99.987
GG1	**MDF**	461.284	1385.414	1499.683	800.000	99.983
GG2	**MDF**	462.443	1361.211	1498.564	800.000	99.998
RCSSP	**MDF**	462.356	1364.327	1499.999	799.999	99.999
	IDF	462.263	1367.022	1499.636	799.999	99.998

On Fig. 4 and Fig. 5, the best solution of MDF for the RCSSP is taken, and IDF is ran on the same initial point. The relaxed variables with respect to the iterations are represented on Fig. 4, showing that they need few iterations to reach a integer value, hence confirming that the constraints (22) are satisfied. The objective function I_{sp} is represented with respect to the disciplinary calls on Fig. 5. These figures (in parallel with Table 3) illustrate the advantages and drawbacks of both MDAO formulations: in terms of disciplinary evaluations and running time, and considering the tolerances on the coupling satisfaction, IDF is more efficient than MDF. Indeed, the MDA needs several disciplinary calls before each iteration, which is not the case for IDF. Advanced techniques exist to increase the MDA's performances (for instance the Aitken's relaxation[27] for a Gauss-Seidel algorithm or the recycling of the previous coupling variables' values to start the new MDA iterations), but have not been used in this study. However, IDF needs more iterations because the method must satisfy eight additional equality constraints (the consistency constraints). Those additional constraints explain as well the lower feasibility rate in IDF.

On that case study, the relaxation-based approach performs well. Indeed, instead of solving four combinatorial problems, the MDF formulation of the method allows to retrieve the best architecture with a smaller computational cost: on average, a total of 669 iterations (and 4050 disciplinary calls) per starting point are required in the first case, whereas 221 iterations (and 1380 calls) per starting point are required for the second one. The performances of IDF show that a decoupled approach is also suitable to solve the design problem efficiently.

DOI: 10.13009/EUCASS2023-834

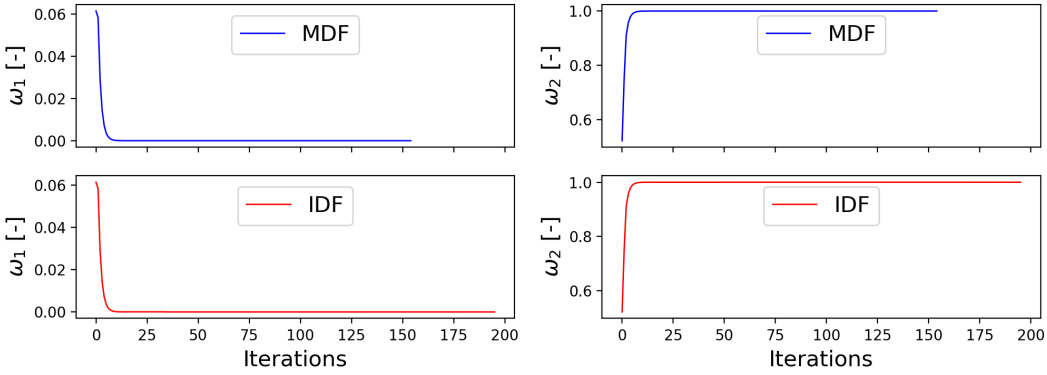


Figure 4: Relaxed variables with respect to the iterations, MDF and IDF starting from the same point

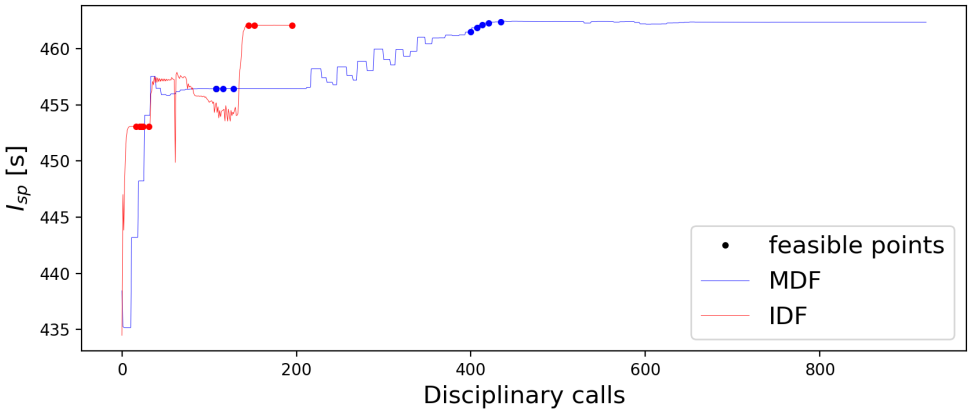


Figure 5: Specific impulse with respect to the disciplinary calls, MDF and IDF starting from the same point

6. Conclusion

In this paper, a mathematical formulation which links Multidisciplinary Design Analysis and Optimization (MDAO) framework to Conditional Search-Space Problems (CSSP) is proposed. Moreover, a relaxation-based method has been presented. It allows to convert a certain class of CSSP into a classical MDAO problem.

The relaxation-based method has several advantages. If the standard variables are continuous, the problem can be solved with gradient-based algorithms under the necessary hypotheses of continuity and differentiability, as shown by the application case in Section 5. If discrete standard variables (integer or categorical) are also part of the problem, the method relaxes the CSSP into a mixed-integer nonlinear programming problem (MINLP). To push the analysis further, it would be interesting to evaluate the performances of classical MINLP techniques on the relaxed problem. The combinatorial of the problem has also been reduced: in the presented application case, a single problem is solved instead of one per engine architecture, resulting in an increased computational efficiency.

However, the main drawback of the method is its scalability. Indeed, adding additional constraints to ensure the consistency of the solution at convergence and handling all the possible variables and constraints at once is a burden. Firstly, if the number of conditional variables increases (or their number of possible values), the relaxed version of the problem may quickly become over-constrained, complexifying the solution process. Secondly, the additional equality constraints may generates a high-number of local minima: finding a global solution becomes hence more difficult. Then, the method might be inapplicable to particular cases where the codes are not flexible enough to handle averaged coupling variables. Another potential drawback that has not been studied in this paper is the impact of introducing a metric and an order over the values of the conditional variables. Indeed, the two categories Expander Bleed (EB) and Gas Generator (GG) of the cycle's choice variable are not ordered, and no distance between the two can be defined. The choice has been made to associate $\omega_1 = 0$ to GG and $\omega_1 = 1$ to EB, imposing implicitly that $EB > GG$ and making the definition of a relative distance possible, but the impact of this choice has not been evaluated. Finally, the method does not allow to eliminate the poorly-performing combinations of conditional variables. For instance, an expander bleed engine is not physically adapted to provide a high thrust at sea level. Running the same method on this application

DOI: 10.13009/EUCASS2023-834

case leads to numerical issues. Indeed, the expander bleed models are still used to compute outputs even if they are not selected by the relaxed variables, causing, for instance, the multidisciplinary analysis algorithm to crash in the MDF formulation.

Finally, both the coupled MDAO formulation (Multidisciplinary Feasible or MDF) and the decoupled one (Individual Discipline Feasible or IDF) have advantages and drawbacks with respect to each other, which does not allow to highlight a better suited formulation for CSSP in the frame of the analysis performed in this paper.

In following works, alternative approaches will be investigated. The most promising option is to build a decomposition based methodology, where the optimization effort is partitioned among several coordinated optimization subproblems. It already exists in the field of MDAO: those formulations are called distributed architectures[36] and the subproblems are generally defined by the disciplines (or subsystems) of the global design problem. The goal would be to adapt it to a CSSP framework to understand if a disciplinary-based decomposition is the most suitable choice, or if another decomposition shall be considered (for instance with respect to conditional variables).

7. Acknowledgments

The authors acknowledge ONERA and ISAE-SUPAERO for funding the PhD thesis of Lucas Baraton. This work was supported by the Chair for Advanced Space Concepts (SaCLab) resulting from the partnership between Airbus Defence and Space, Ariane Group, and ISAE-SUPAERO.

References

[1] Ossama Abdelkhalik. Hidden genes genetic optimization for variable-size design space problems. *Journal of Optimization Theory and Applications*, 156:450–468, 2013.

[2] Belarmino Adenso-Diaz and Manuel Laguna. Fine-tuning of algorithms using fractional experimental designs and local search. *Operations research*, 54(1):99–114, 2006.

[3] Carlos Ansótegui, Meinolf Sellmann, and Kevin Tierney. A gender-based genetic algorithm for the automatic configuration of algorithms. In *Principles and Practice of Constraint Programming-CP 2009: 15th International Conference, CP 2009 Lisbon, Portugal, September 20-24, 2009 Proceedings 15*, pages 142–157. Springer, 2009.

[4] Astrium. Vulcain 2 rocket engine. Technical report, 2011.

[5] Charles Audet and John E Dennis Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on optimization*, 17(1):188–217, 2006.

[6] Charles Audet, Edward Hallé-Hannan, and Sébastien Le Digabel. A general mathematical framework for constrained mixed-variable blackbox optimization problems with meta and categorical variables. In *Operations Research Forum*, volume 4, page 12. Springer, 2023.

[7] J Barton, G Turin, and N Girard. Development status of the vulcain 2 engine. In *35th Joint Propulsion Conference and Exhibit*, page 2616, 1999.

[8] Ian H. Bell, Jorrit Wronski, Sylvain Quoilin, and Vincent Lemort. Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical property library coolprop. *Industrial & Engineering Chemistry Research*, 53(6):2498–2508, 2014.

[9] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[10] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

[11] Rene Bosson, Patrick Sabin, and Gilles Turin. Improvements of the hydrogen turbopump for the vulcain 2 engine. In *35th Joint Propulsion Conference and Exhibit*, page 2344, 1999.

[12] Mohamed Amine Bouhlel, John T. Hwang, Nathalie Bartoli, Rémi Lafage, Joseph Morlier, and Joaquim R. R. A. Martins. A python surrogate modeling framework with derivatives. *Advances in Engineering Software*, page 102662, 2019.

[13] Jasper Bussemaker, Luca Boggero, and Pier Davide Ciampa. From system architecting to system design and optimization: A link between mbse and mdao. In *INCOSE International Symposium*, volume 32, pages 343–359. Wiley Online Library, 2022.

[14] Jasper H Bussemaker, Pier Davide Ciampa, and Bjoern Nagel. System architecture design space exploration: An approach to modeling and optimization. In *AIAA Aviation 2020 Forum*, page 3172, 2020.

[15] Guobiao Cai, Jie Fang, Yuntao Zheng, Xiaoyan Tong, Jun Chen, and Jue Wang. Optimization of system parameters for liquid rocket engines with gas-generator cycles. *Journal of Propulsion and Power*, 26(1):113–119, 2010.

[16] Marc Claesen, Jaak Simm, Dusan Popovic, Yves Moreau, and Bart De Moor. Easy hyperparameter search using optunity. *arXiv preprint arXiv:1412.1114*, 2014.

[17] Radwa Elshawi, Mohamed Maher, and Sherif Sakr. Automated machine learning: State-of-the-art and open challenges. *arXiv preprint arXiv:1906.02287*, 2019.

[18] Hugo Jair Escalante, Manuel Montes, and Luis Enrique Sucar. Particle swarm model selection. *Journal of Machine Learning Research*, 10(2), 2009.

[19] Sanford Gordon and Bonnie J McBride. Computer program for calculation of complex chemical equilibrium compositions and applications. part 1: Analysis. Technical report, 1994.

DOI: 10.13009/EUCASS2023-834

[20] Justin S. Gray, John T. Hwang, Joaquim R. R. A. Martins, Kenneth T. Moore, and Bret A. Naylor. OpenMDAO: An Open-Source Framework for Multidisciplinary Design, Analysis, and Optimization. *Structural and Multidisciplinary Optimization*, 59:1075–1104, 2019.

[21] Justin S. Gray, John T. Hwang, Joaquim R. R. A. Martins, Kenneth T. Moore, and Bret A. Naylor. OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization. *Structural and Multidisciplinary Optimization*, 59(4):1075–1104, April 2019.

[22] Oskar J Haidn. Advanced rocket engines. *Advances on propulsion technology for high-speed aircraft*, 1:6–1, 2008.

[23] Armin Herbertz. Component modeling for rocket engine cycle analysis. *Transactions of the Japan Society for Aeronautical and Space Sciences, Aerospace Technology Japan*, 14(ists30):119–127, 2016.

[24] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration (extended version). *Technical Report TR-2010–10, University of British Columbia, Computer Science, Tech. Rep.*, 2010.

[25] Rodolphe Jenatton, Cedric Archambeau, Javier González, and Matthias Seeger. Bayesian optimization with tree-structured dependencies. In *International Conference on Machine Learning*, pages 1655–1664. PMLR, 2017.

[26] Yojiro Kakuma, Masaaki Yasui, Tadaoki Onga, Ryuichi Sekita, and Shogo Warashina. Le-5b engine development. In *36th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, page 3775, 2000.

[27] Gaetan KW Kenway, Graeme J Kennedy, and Joaquim RRA Martins. Scalable parallel approach for high-fidelity steady-state aeroelastic analysis and adjoint derivative computations. *AIAA journal*, 52(5):935–951, 2014.

[28] Lars Kotthoff, Chris Thornton, Holger H Hoos, Frank Hutter, and Kevin Leyton-Brown. Auto-weka: Automatic model selection and hyperparameter optimization in weka. *Automated machine learning: methods, systems, challenges*, pages 81–95, 2019.

[29] Andrew B Lambe and Joaquim RRA Martins. Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes. *Structural and Multidisciplinary Optimization*, 46:273–284, 2012.

[30] M Leonardi, F Di Matteo, and F Nasutia. Parametric study on the performance of an expander bleed engine. *Aerotecnica Missili & Spazio*, 96:32–43, 2017.

[31] Marco Leonardi, Marco Pizzarelli, and Francesco Nasuti. Analysis of thermal stratification impact on the design of cooling channels for liquid rocket engines. *International Journal of Heat and Mass Transfer*, 135:811–821, 2019.

[32] Julien-Charles Lévesque, Audrey Durand, Christian Gagné, and Robert Sabourin. Bayesian optimization for conditional hyperparameter spaces. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 286–293. IEEE, 2017.

[33] Stefano Lucidi and Veronica Piccialli. A derivative-based algorithm for a particular class of mixed variable optimization problems. *Optimization Methods and Software*, 19(3-4):371–387, 2004.

[34] Stefano Lucidi, Veronica Piccialli, and Marco Sciandrone. An algorithm model for mixed variable programming. *SIAM Journal on Optimization*, 15(4):1057–1084, 2005.

[35] Xingchen Ma and Matthew B Blaschko. Additive tree-structured conditional parameter spaces in bayesian optimization: A novel covariance function and a fast implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):3024–3036, 2020.

[36] Joaquim RRA Martins and Andrew B Lambe. Multidisciplinary design optimization: a survey of architectures. *AIAA journal*, 51(9):2049–2075, 2013.

[37] Joaquim RRA Martins and Andrew Ning. *Engineering design optimization*. Cambridge University Press, 2021.

[38] Hui Meen Nyew, Ossama Abdelkhalik, and Nilufer Onder. Structured-chromosome evolutionary algorithms for variable-size autonomous interplanetary trajectory planning optimization. *Journal of Aerospace Information Systems*, 12(3):314–328, 2015.

[39] Adam Okninski, Jan Kindracki, and Piotr Wolanski. Multidisciplinary optimisation of bipropellant rocket engines using h2o2 as oxidiser. *Aerospace Science and Technology*, 82:284–293, 2018.

[40] Randal S Olson, Nathan Bartley, Ryan J Urbanowicz, and Jason H Moore. Evaluation of a tree-based pipeline optimization tool for automating data science. In *Proceedings of the genetic and evolutionary computation conference 2016*, pages 485–492, 2016.

[41] Julien Pelamatti, Loïc Brevault, Mathieu Balesdent, El-Ghazali Talbi, and Yannick Guerin. Efficient global optimization of constrained mixed variable problems. *Journal of Global Optimization*, 73:583–613, 2019.

[42] Julien Pelamatti, Loïc Brevault, Mathieu Balesdent, El-Ghazali Talbi, and Yannick Guerin. Bayesian optimization of variable-size design space problems. *Optimization and Engineering*, 22:387–447, 2021.

[43] Sebastian Raschka. Model evaluation, model selection, and algorithm selection in machine learning. *arXiv preprint arXiv:1811.12808*, 2018.

[44] Carl Edward Rasmussen, Christopher KI Williams, et al. *Gaussian processes for machine learning*, volume 1. Springer, 2006.

[45] Paul Saves, Youssef Diouane, Nathalie Bartoli, Thierry Lefebvre, and Joseph Morlier. A general square exponential kernel to handle mixed-categorical variables for gaussian process. In *AIAA AVIATION 2022 Forum*, page 3870, 2022.

[46] Elias Schede, Jasmin Brandt, Alexander Tornede, Marcel Wever, Viktor Bengs, Eyke Hüllermeier, and Kevin Tierney. A survey of methods for automated algorithm configuration. *Journal of Artificial Intelligence Research*, 75:425–487, 2022.

[47] George Paul Sutton and Oscar Biblarz. *Rocket propulsion elements*. A Wiley-Interscience publication. Wiley, New York Weinheim, 7. ed edition, 2001.

[48] Kevin Swersky, David Duvenaud, Jasper Snoek, Frank Hutter, and Michael A Osborne. Raiders of the lost architecture: Kernels for bayesian optimization in conditional parameter spaces. *arXiv preprint arXiv:1409.4011*, 2014.

[49] Juan M Tizón and Alberto Román. A mass model for liquid propellant rocket engines. In *53rd AIAA/SAE/ASEE Joint Propulsion Conference*, page 5010, 2017.

[50] Yohann Torres. *Heat and mass transfers in curved cooling channels of rocket engines*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrésis, 2008.

[51] S Trollheden, B Bergenlid, A Aglund, and A Pettersson. Development of the turbines for the vulcain 2 turbopumps. In *35th Joint Propulsion Conference and Exhibit*, page 2342, 1999.

[52] Neil Wu, Gaetan Kenway, Charles A. Mader, John Jasa, and Joaquim R. R. A. Martins. pyoptsparse: A python framework for large-scale constrained nonlinear optimization of sparse systems. *Journal of Open Source Software*, 5(54):2564, 2020.

[53] Li Yang and Abdallah Shami. On Hyperparameter Optimization of Machine Learning Algorithms: Theory and Practice. *Neurocomputing*, 415:295–316, November 2020.

DOI: 10.13009/EUCASS2023-834

Appendix: component modeling and validation

7.1 Component modeling

This subsection briefly presents the key details of the models used for the engine's numerical simulation. The majority of them have been built based on Sutton and Biblarz's book.[47]

Several components of the model are surrogate models built using kriging models from the toolbox SMT.[12] The advantage of kriging models is the availability of their analytical gradient in order to use gradient-based optimizers.

Thrust chamber and combustion properties: The combustion properties in the combustion chamber and in the gas generator are evaluated by a surrogate model of NASA's software *CEA*.[19] The characteristic length of those two chambers is taken as $L^* = 1m$. The characteristic velocity and thrust coefficient efficiencies are defined as: $\eta_{c^*} = 0.97$ and $\eta_{c_F} = 0.98$. In case of a GG, the pressure inside the gas generator p_{GG} is considered equal to the main combustion chamber pressure p_{CC}.

Cooling system: The regenerative cooling model is based on a thermal equilibrium at the throat between the burnt gases and the coolant (which is the LH2 in this problem). If the contrary is not mentioned, all the quantities discussed here are computed at the throat. The gas and the coolant properties (heat capacity, viscosity, *etc*) are estimated with surrogate models built on data from *CEA*[19] and from the library *CoolProp*[8] respectively. The convective coefficients are computed using Bartz's correlation[47] for the gas side and the Dittus-Boelter correlation[31] for the coolant side. The wall temperature can then be retrieved by computing successively the total convective coefficient in the wall and the heat flux. The latter is integrated on the nozzle surface and on the full thrust chamber surface (nozzle and combustion chamber) to obtain the coolant total temperatures at the throat and at the exit of the cooling system respectively, assuming a temperature of 20 K in the tank. As the maximum heat flux on the engine is found at the throat, the wall temperature constraint $T_{wg,t,max} = 1500K$ is considered there. The coolant total temperature can be fed back to the computation of the wall temperature and the estimation of the coolant properties using the *CoolProp* surrogate model. Hence, an iterative process (Gauss-Seidel in this work) is necessary to perform the disciplinary analysis. In terms of dimensions, the inter-channel thickness is $1mm$ and the wall thickness is $0.5mm$.

Feed system: The feed system model is inspired from the work of Leonardi *et al.*[30] (even if the engine architectures are different). The central point of the models are the balance equations for 1 (Eq. (23)) and 2 turbines (Eq. (24)).

$$\dot{m}_{tot,f}\frac{\Delta p_f}{\eta_{p,f}\rho_f} + \dot{m}_{tot,o}\frac{\Delta p_o}{\eta_{p,o}\rho_o} = \dot{m}_{turb}c_p T_{in}\eta_t[1 - PR^{((1-\gamma)/\gamma)}] \tag{23}$$

$$\dot{m}_{tot,f}\frac{\Delta p_f}{\eta_{p,f}\rho_f} = \dot{m}_{t,f}c_p T_{in}\eta_{t,f}[1 - PR_f^{(1-\gamma)/\gamma}] \quad \text{and} \quad \dot{m}_{tot,o}\frac{\Delta p_o}{\eta_{p,o}\rho_o} = \dot{m}_{t,o}c_p T_{in}\eta_{t,o}[1 - PR_o^{(1-\gamma)/\gamma}] \tag{24}$$

$\dot{m}_{tot,f}$ and $\dot{m}_{tot,o}$ are the total oxidizer and fuel mass flow rates. $\dot{m}_{turb}$ is the mass flow rate derived in the turbine(s). In the two-turbines case: $\dot{m}_{turb} = \dot{m}_{turb,f} + \dot{m}_{turb,o}$ which are the mass flow rates in the fuel and oxidizer turbines. Note that c_p, T_{in} and γ (heat capacity at constant pressure, inlet temperature, heat capacity ratio) are dependent on the cycle choice. In the GG case, they are computed by evaluating the properties of the burnt gases by the feed system discipline. In the EB case, they are the coolant properties at the cooling system exit and are computed by the cooling discipline. The losses are assumed proportional to the chamber pressure: $\Delta p = a p_{CC}$, with a a real number depending on the propellant path from the pump outlet to the chamber. On the fuel side, $a_f = 1.45$ is considered, on the oxidizer side, $a_o = 1.3$. The pump and turbine efficiencies are assumed constant: $\eta_{p,o} = \eta_{p,f} = 0.7$, $\eta_t = 0.4$, $\eta_{t,o} = 0.4$ and $\eta_{t,f} = 0.6$. The densities in the tank for the fuel (H_2) and the oxidizer (O_2) are respectively $\rho_f = 70.99\text{kg/m}^3$ and $\rho_o = 1143\text{kg/m}^3$. Finally, the shaft rotational speeds are assumed constant and equal to the LE-5B design point.[26] If their are two turbines: $N_{r,f} = 52120$rpm and $N_{r,o} = 17630$rpm. If there is only one turbine the rotational speed is: $N_r = 17630$rpm. Fractions of mass flow rates can be defined $X = \dot{m}_{turb}/(\dot{m}_f + \dot{m}_o)$ and $X_t = \dot{m}_{t,o}/\dot{m}_{turb}$. The second one is exclusive to the two-turbines case and is useful to compute the secondary specific impulse $I_{sp,sec}$. Finally, X and X_t can be derived analytically from those equations.

Mass models: The mass models have been taken from the literature.[23,49] The nozzle is considered as conical and 1cm thick. The thrust chamber is made of steel and the tanks are made of aluminum.

DOI: 10.13009/EUCASS2023-834

7.2 Validation

The models earlier described are validated in this subsection on the design point of the Vulcain 2 engine[4] at sea level. The design variables' value are given on Table 6 and Table 7 shows the computed quantities of interest with respect to data. The quantities of interests are chosen to be representative of the three disciplines (thrust chamber, feed system and cooling) and are added to Table 7 if trustworthy data are available. All the data are taken from the literature, the associated reference is given for each value that is presented. Additional data that are not reported here for conciseness were required to run the models (the turbines efficiencies for instance). They can be found in the references given in this subsection.

Table 6: Validation point

Quantity	$p_{CC}{}^{22}$ [bar]	$OF_{CC}{}^{11}$ [-]	$OF_{GG}{}^{22}$ [-]	$PR_f{}^{51}$ [-]	$PR_o{}^{51}$ [-]	$h_t{}^{50}$ [mm]	$w_t{}^{50}$ [mm]
Value	116	6.1	0.9	15.5	12	11	1.3

Table 7: Computed quantities versus data for the Vulcain 2 engine

Quantity	**Unit**	**Description**	**Computed value**	**Data**	**Relative error [%]**
		General performances			
$I_{sp,sl}{}^{22}$	[s]	specific impulse at sea level	329	320	2.8
		Thrust chamber			
$D_t{}^{4}$	[m]	throat diameter	0.265	0.274	3.3
ϵ^{22}	[-]	area ratio	59	60	1.7
$T_{CC}{}^{50}$	[K]	combustion chamber temperature	3555	3500	1.6
		Feed system			
$\dot{m}_{GG}{}^{22}$	[kg/s]	mass flow rate in the GG	10.5	9.7	8.2
$\dot{m}_{tot,f}{}^{11}$	[kg/s]	fuel mass flow rate	45	45	0
$\dot{m}_{tot,o}{}^{7}$	[kg/s]	oxidizer mass flow rate	246	274	10.2
$T_{GG}{}^{22}$	[K]	temperature in the GG	884	845	4.6
$X^{7,11,22}$	[%]	see subsection 7.1	3.6	3.0	20
$X_t{}^{7,11,22}$	[%]	see subsection 7.1	33.1	37.1	10.8
		Cooling			
$N_{channels}{}^{50}$	[-]	number of cooling channels	364.97	350	4.3
$\Delta T_{0,cool}{}^{50}$	[-]	coolant temperature rise	155	88	76.1

Table 7 shows that the models are estimating the quantities of interest with a fair precision. The highest error is made on the total temperature rise in the cooling channels. Two factors are responsible. First, the heat flux computed by the cooling system discipline is evaluated at the throat, where the maximum occurs. As it is assumed constant on the whole engine, the temperatures at the throat and at the exit of the system are over estimated. Moreover, in those models, the cooling channels are assumed to cover the full nozzle: it is not the case for the Vulcain 2, for which only a portion of the nozzle is cooled. Then, the number of cooling channels is not an integer in our models. It has been relaxed into a continuous variable to simplify the design problem.

Two main improvements can be made regarding those models. The first one is a better regenerative cooling model, implementing for instance equations to model the flow inside the cooling channels. The second one is an accurate model of the feed system where pressure losses are accounted for, as well as the turbines and pumps efficiencies.