Multidisciplinary Design Optimization of Launch Vehicles

Mathieu Balesdent

ÉCOLE DOCTORALE
SCIENCES POUR L'INGÉNIEUR, GÉOSCIENCES, ARCHITECTURE

École Centrale de Nantes

Année 2011

Thèse de Doctorat

Spécialité : Génie Mécanique

présentée et soutenue publiquement par

Mathieu BALESDENT

le 3 novembre 2011

à l'Office National d'Etudes et de Recherches Aérospatiales

OPTIMISATION MULTIDISCIPLINAIRE DE LANCEURS

(Multidisciplinary Design Optimization of Launch Vehicles)

JURY

Directeur de thèse :	Philippe DÉPINCÉ
Encadrant Onera :	Nicolas BÉREND
Laboratoires :	Office National d'Etudes et de Recherches Aérospatiales
	Institut de Recherche en Communications et Cybernétique de Nantes

N° E.D. : 498-196

ACKNOWLEDGMENTS

This work has been realized in the Onera's *System Design and Performance Evaluation* Department, in collaboration with the CNES *Launchers Directorate* and the team *Systems Engineering Products, Performance and Perceptions* of IRCCyN. I would like to thank the director of DCPS, Dr. Donath, for giving me the opportunity to achieve this thesis at Onera.

I would like to thank Dr. Désidéri, who did me the honour of chairing my PhD committee. I express my gratitude to Dr. Bassir and Prof. Siarry for reviewing this PhD thesis. I also thank Dr. Laurent-Varin, Prof. Lavagna and Dr. Le Riche for accepting to be part of my PhD committee.

I would like to acknowledge Prof. Dépincé for agreeing to be my advisor during these three years. I would like to express my thankfulness to my Onera supervisor, M. Bérend for his comments, suggestions, edits, and advices during the thesis.

I would like to thank my CNES supervisor, M. Oswald. I thank also M. Amouroux for his help in the elaboration of the launch vehicle disciplinary models.

I owe a great amount of thanks to the people whom I've been working with every day at the Onera's DCPS. Especially, I thank Dr. Piet-Lahanier for her numerous advices all along the thesis.

I would also like to thank the Onera PhD students (Dalal, Ariane, Rata, Walid, Julien, Joseph, François, Yohan and Damien) who have provided the most exciting working environment. Special thanks go to my office mate, Rudy, *qui quæsivit rerum causas*, for those numerous happy hours spent during the thesis.

Last but not least, I would like to thank my parents, who believed in me and always encouraged me. Finally, I would especially like to thank my fiancée Sophie, for all her love and for her understanding when I spent my time working on this thesis. I appreciate her support and her sacrifices very much during this endeavor.

This PhD thesis has been co-funded by CNES and Onera.

Non est ad astra mollis e terris via.
There is no easy way from the earth to the stars.

Hercules Furens
SENECA THE YOUNGER

To Sophie, Cécile & Claude

Contents

List of Figures

List of Tables

Nomenclature

MDO variables

z	Design variables
y	Coupling variables
x	State variables

MDO functions and constraints

c	Coupling functions
f	Objective function
g	Inequality constraints
h	Equality constraints
R	Residuals
X	State variable computation functions
J	Subsystem-level intermediary objective functions
s	State dynamics functions

Launch vehicle design variables

D	Stage diameter
Dne	Nozzle exit diameter
Isp	Specific impulse
Mp	Propellant mass
Md	Dry mass
n_f	Axial load factor
Pc	Chamber pressure
Pe	Nozzle exit pressure
q	Mass flow rate
Rm	Mixture ratio
$\frac{T}{W}$	Thrust to weight ratio

Trajectory variables

r	Radius
v	Velocity
u	Control
α	Angle of attack
γ	Flight path angle
θ	Pitch angle

Notations

z_{sh}	Shared design variables
$\bar{z}_k$	Design variables specific to the kth subsystem
z^*	Local copies of z
$\tilde{z}$	Approximation of z
z°	Feasible z
z^d	Design variables present in the subsystem database
$\widehat{z}$	Estimation of z
z_d	Design variables
z_c	Control variables
z'_{d_i}	Design variables of the ith stage intervening in other stage dynamics
X_c	State variable computation functions ensuring the consistency of the couplings
$\bar{x}$	Mean value of x
σ_x	Standard deviation of x
f^*	Optimized value of f

Acronyms

AAO	All At Once
ATC	Analytical Target Cascading
BLISS	Bi-Level Integrated System Synthesis
CO	Collaborative Optimization
CSSO	Concurrent SubSpace Optimization
DIVE	Discipline Interaction Variable Elimination
EGO	Efficient Global Optimization
ELV	Expendable Launch Vehicle
FD	Finite Differences
FPI	Fixed Point Iteration
GA	Genetic Algorithm
GAGGS	Genetic Algorithm Guided Gradient Search
GLOW	Gross Lift-Off Weight
GSE	Global Sensitivity Equation
GTO	Geostationary Transfer Orbit
IDF	Individual Discipline Feasible
KKT	Karush-Kuhn-Tucker
LDC	Local Distributed Criteria
$LH2$	Liquid Hydrogen
LOX	Liquid Oxygen
LVD	Launch Vehicle Design
MCO	Modified Collaborative Optimization
MDA	MultiDisciplinary Analysis
MDF	MultiDiscipline Feasible
MDO	Multidisciplinary Design Optimization
MDOIS	Multidisciplinary Design Optimization based on Independent Subspaces
MOPCSSO	Multi-Objective Pareto Concurrent SubSpace Optimization
MSTO	Multi Stage To Orbit
NAND	Nested ANalysis and Design
NLP	Non Linear Programming
OBD	Optimization Based Decomposition
POA	Post-Optimality Analysis
POST	Program to Optimize Simulated Trajectories
PSO	Particle Swarm Optimization
RLV	Reusable Launch Vehicle

RP	Rocket Propellant
RSM	Response Surface Method
SA	Simulated Annealing
SAND	Simultaneous ANalysis and Design
SQP	Sequential Quadratic Programming
SNN	Single NAND NAND
SSA	System Sensitivity Analysis
SSN	Single SAND NAND
SSS	Single SAND SAND
SSTO	Single Stage To Orbit
SWORD	Stage-Wise decomposition for Optimal Rocket Design

Introduction

Since the beginning of the XXth century and the works of Konstantin Tsiolkovsky concerning the multi stage rocket design, published in 1903 in “The Exploration of Cosmic Space by Means of Reaction Devices” , the design of launch vehicles has been constantly growing and has become a strategic domain for it enables the access to space. Launch Vehicle Design (LVD) is a very complex optimization process in which the slightest mistake may induce economical, material and human disastrous consequences (*e.g.* explosion of the brazilian VLS launch vehicle in 2002). This sector, due to the development of many new launch vehicles over the last few years, has been becoming more and more competitive. For this reason, it is primordial to be able to design more and more efficient launch vehicles *i.e.* able to put into orbit ever increasing payloads at the lesser cost, ensuring an acceptable level of reliability at the same time. This requires in early design studies an efficient optimization process so as to explore a very large search space in order to quickly obtain an appropriate launch vehicle configuration which meets the given mission requirements.

LVD involves numerous disciplines such as aerodynamics, structure, trajectory, mass budget, *etc.* These disciplines require specific tools and may result in conflicting decisions. For instance, the choice of a large stage diameter has antagonistic effects on launcher performance through aerodynamics and structural sizing. This induces a very complex design process which includes the search of multidisciplinary compromises. Therefore, in addition to the mastery of disciplinary technologies, LVD requires system-oriented design methodologies in order to organize the design process.

The classical engineering design method used in LVD consists of a loop between the different disciplinary tasks. At each iteration of this loop, each discipline is reoptimized according to the data given by the previous discipline. It gives to the next discipline new data, in function of the results obtained during its own optimization. Thus, this design process needs several iterations to converge, may present some problems in order to find (if necessary) the compromises between the disciplines and may not lead to the global optimal design. Consequently, LVD requires the use of adapted tools in order to exploit the couplings between the different disciplines, to make the compromise search easier, and to improve the robustness and the efficiency of the whole design process.

Multidisciplinary Design Optimization (MDO), is a research field of engineering sciences which aims at elaborating design methodologies devoted to solving complex multidisciplinary design problems. To this end, MDO regroups several topics such as the elaboration of new design problem formulations, the creation and the use of meta models in order to reduce the calculation time, the handling of uncertainties in order to improve the robustness or the elaboration of new optimization algorithms when the classical algorithms are inefficient.

Many MDO methods can be found in literature [Balling and Sobieszczanski-Sobieski, 1994; Alexandrov and Hussaini, 1995; Sobieszczanski-Sobieski and Haftka, 1997; Agte et al., 2009; Tosserams et al., 2009]. These methods can be decomposed into two categories, with regard to the presence of one or several optimization levels.

The most used MDO method in LVD is the MultiDiscipline Feasible (MDF) method [Duranté et al., 2004; Tsuchiya and Mori, 2004; Bayley and Hartfield, 2007]. This single level method splits up the design problem according to the different disciplines and associates a global optimizer (system level) and disciplinary analysis tools (subsystem level). In order to ensure the consistency of the couplings, this method involves at the subsystem level a multidisciplinary analysis (MDA). The MDF method makes the search of the global optimum easier with respect to the classical engineering method because it uses a single optimizer which handles all the design variables. Moreover, this method allows the search of disciplinary compromises, thanks to the use of the MDA. Nevertheless, the handling of all the design variables by the single optimizer may induce a very large search space. This may pose several problems such as the requirement of an appropriate initialization and a good knowledge of the design variable search domain in order to converge. Therefore, this method is only adapted to small design problems for which the search domain can be well known in advance.

In order to address more important design problems, some MDO methods involving several levels of optimization have been proposed. These methods (*e.g.* Collaborative optimization and its derivatives [Braun and Kroo, 1995; Alexandrov and Lewis, 2000; DeMiguel and Murray, 2000], Bi-Level Integrated Systems Synthesis [Sobieszczanski-Sobieski et al., 1998, 2000], *etc.*) decompose the design problem according to the different disciplines. They use disciplinary optimizers at the subsystem level and a global optimizer at the system level in order to coordinate the different disciplinary optimizations while optimizing the global objective function.

LVD is a specific MDO problem. Indeed, it involves a dynamical system to optimize and combines the optimizations of the design variables and the trajectory. The trajectory optimization induces a trajectory simulation with stage separations. This latter involves a highly non linear ordinary differential equation system which is complex to integrate. Moreover, the trajectory is coupled with all the other disciplines and is subjected to strict equality constraints, due to the mission requirements. These constraints are very difficult to satisfy and considerably limit the feasible search domain *i.e.* the domain for which all the constraints are satisfied.

All the MDO methods applied to LVD in literature use a decomposition of the design problem into the different disciplines (Fig. 1). In such a decomposition, the trajectory is most often considered as a black box for the optimizer and is optimized in the same way as the other disciplines. Therefore, this decomposition may not be the most adapted to the specific LVD problem. Another decomposition of the design problem which exploits the couplings between the trajectory and the design variable optimization seems to be a valuable way of improvement of the whole optimization process.

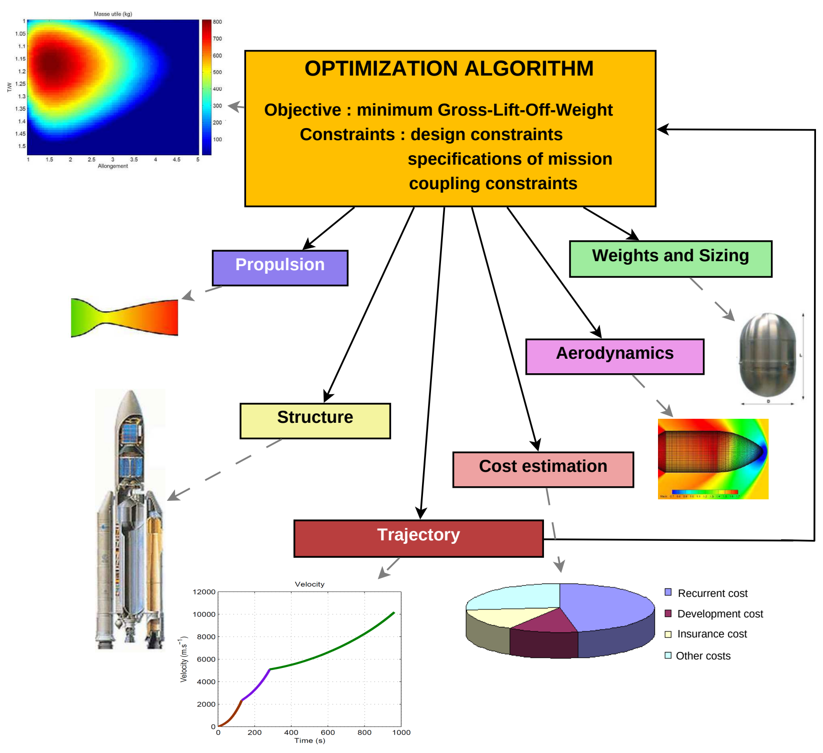


Figure 1: Classical Launch Vehicle Design decomposition

This thesis is focused on the elaboration of new MDO methods dedicated to LVD. These methods, called Stage-Wise decomposition for Optimal Rocket Design (SWORD), allow to place the trajectory optimization at the center of the design process. The SWORD methods transform the initial complex design optimization problem into the coordination of smaller single stage MDO problems, which are easier to solve. To this end, new MDO formulations of the LVD problem and an associated optimization strategy are proposed. The SWORD methods are analyzed and compared to the most used MDO method in literature (MDF).

This manuscript is organized into three parts. The first part is devoted to drawing up a panorama of the MDO methods used in LVD. Chapter 1 presents the fundamental concepts used to describe a general MDO process. In Chapter 2, we detail the main MDO methods applied in LVD. Chapter 3 concerns the description of the main optimization algorithms used in MDO and the different techniques employed to perform the trajectory optimization. Finally, Chapter 4 details the analysis of the application of the MDO methods in LVD, and expresses some possible ways of improvement with regard to this specific MDO problem.

The second part of this document is related to the description and the analysis of the proposed MDO formulations. To this end, we express in Chapter 5 four MDO formulations in order to solve the LVD problem. Chapter 6 is devoted to the description of the application case selected in this study : the optimization of three-stage-to-orbit launch vehicle for the minimization of the Gross-Lift-Off-Weight. Finally, Chapter 7 compares the SWORD formulations with the most used MDO method, theoretically and numerically in case of global search study.

The last part of this document is devoted to the description and the analysis of the proposed optimization strategy dedicated to the SWORD formulations. To this end, this part is organized into two chapters. Chapter 8 concerns the elaboration of the optimization strategy. In this chapter, we detail the different parts of the optimization strategy and we describe the proposed optimization algorithms. Finally, Chapter 9 describes the numerical analysis of the optimization strategy. In this chapter, we analyze each of the optimization strategy phases and we study the performance of the whole optimization process (*i.e.* MDO formulation and optimization algorithms), in terms of calculation time, quality of the optimum and robustness to the initialization. Finally, we conclude on the efficiency of the SWORD methods, their use in Launch Vehicle Design preliminary studies and we detail several ways of improvement.

The appendices present some model considerations concerning the application case, and detail some optimization techniques and mathematical demonstrations used in Chapter 8. The last appendix is constituted of an extended abstract of this thesis written in French.

Part I

Panorama of Multidisciplinary Design Optimization methods used in Launch Vehicle Design

Chapter 1

Generalities about Multidisciplinary Design Optimization

Contents

1.1 Introduction

Multidisciplinary Design Optimization (MDO), also named *Multidisciplinary Optimization* is a relatively recent field of engineering sciences whose objective is to address more efficiently design problems incorporating different disciplines. The MDO has been used in a great number of domains such as structure, automotive, electronics or aerospace engineering, and allows to solve complex problems which are difficult to handle with the classical design methods. This research field could grow up thanks to the increase of computation power in the second half of the XXth century. Indeed, this increase has made the numerical optimization of complex problems possible and has paved the way for the complete system design.

By handling the different disciplines simultaneously, the MDO techniques facilitate the search of the global optimal design, which might not be obtained when the disciplines are handled sequentially. Indeed, in most of the design problems, the different disciplines may lead to antagonistic decisions (*e.g.* structure and aerodynamics in launch vehicle design, as we shall see later). In such cases, the MDO techniques aim to find compromises between the different disciplines in order to reach the global optimal design.

Handling at the same time a series of disciplines increases significantly the complexity of the problem to solve. One of the branches of the MDO field is dedicated to the elaboration of new formulations of the optimization problem which aim at reducing the complexity of the problem and at allowing the more efficient use of the traditional optimization methods. The works presented in this document lie within this scope.

Instead of disciplinary codes in a computer or computer networks, the MDO may also address design problems involving engineer teams all over the world. Indeed, due to the globalization of the industries, the system design can be split up into different research centers located in different countries. In this case, the data exchanges between the teams become a crucial point of the design process and MDO provides new tools for the designers in order to make the design process more efficient.

This part is devoted to the description of the application of the MDO methods to the Launch Vehicle Design. The first chapter of this part concerns the presentation of the fundamental concepts necessary to describe a MDO process. In the second chapter, we present the main MDO methods used in literature. The third chapter concerns the description of the classical optimization algorithms used in a MDO process and the main optimal control methodologies. Finally, in the fourth chapter, we analyze the application of the MDO methods in Launch Vehicle Design in order to compare the different presented methods and to propose several ways of improvement.

1.2 Mathematical formulation of the general MDO problem

In this section, we define the fundamental notions necessary to describe a classical MDO process. The Launch Vehicle Design problem can be decomposed into different ways. These decompositions imply different optimization architectures. Indeed, the different disciplines can be considered separately or can be optimized simultaneously inside a same system. Hereinafter, we will use, with the same signification, both terms *discipline* and *subsystem*, in order to describe the different processes at the subsystem level (Fig. 1.1). A subsystem may regroup several disciplines and can be a part of a system to optimize (*e.g.* a stage of a launch vehicle). In this section, we firstly describe the general mathematical formulation of a MDO problem and then we expose the different involved variables, functions and constraints. Finally, we detail the different ways to handle the disciplinary equations and couplings.

1.2.1 Problem formulation

The general formulation of a MDO problem can be written as follows :
Formulation I.1 (**General MDO formulation**)

$$
\begin{aligned}
\textbf{Minimize} \quad & f(x, y, z) \\
\textbf{With respect to} \quad & z \in \mathcal{Z} \\
\textbf{Subject to} \quad & g(x, y, z) \leq 0 && (1.1) \\
& h(x, y, z) = 0 && (1.2) \\
& \forall i \in \{1, \cdots, n\}, \forall j \neq i, y_i = \{c_{ji}(x_j, y_j, z_j)\}_j && (1.3) \\
& \forall i \in \{1, \cdots, n\}, R_i(x_i, y_i, z_i) = 0 && (1.4)
\end{aligned}
$$

A general MDO process is illustrated in Figure 1.1. All the different variables and functions intervening in the MDO process will be described in the following sections.

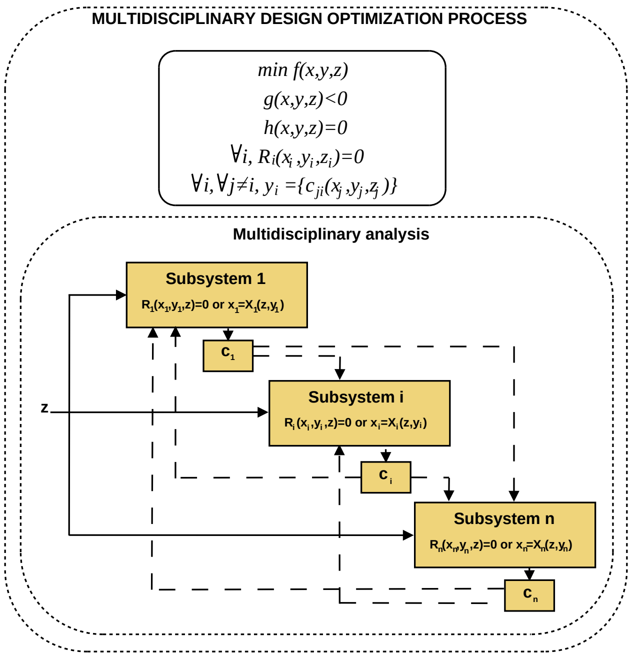

MULTIDISCIPLINARY DESIGN OPTIMIZATION PROCESS
min f(x,y,z)
g(x,y,z)<0
h(x,y,z)=0
∀i, R_i(x_i,y_i,z_i)=0
∀i,∀j≠i, y_i ={c_ji(x_j,y_j,z_j)}
Multidisciplinary analysis
Subsystem 1
R_1(x_1,y_1,z)=0 or x_1=X_1(z,y_1)
c_1
z
Subsystem i
R_i(x_i,y_i,z)=0 or x_i=X_i(z,y_i)
c_i
Subsystem n
R_n(x_n,y_n,z)=0 or x_n=X_n(z,y_n)
c_n

Figure 1.1: Used nomenclature

1.2.2 Types of variables

A general MDO process involves several types of variables. These variables play specific roles and are used at the different steps of the MDO process. We can differentiate three categories of variables in a general MDO problem:

- z : design variables. These variables evolve all along the optimization process in order to find the optimal design. They can be used in one or several subsystems:

$z = \{z_{sh}, \bar{z}_k\}$. The subscript $_{sh}$ stands for the variables which are shared between different subsystems (global variables) and $\bar{z}_k$ denotes the variables which are specific to one subsystem (local variables). Moreover, we use the notation z_k to describe the variables which refer to the k^{th} subsystem.

- y : coupling variables. These variables are used to link the different subsystems and to evaluate the consistency of the design with regard to the coupling.

- x : state (or disciplinary) variables. These variables (which are not the design variables) can vary during the disciplinary analysis in order to find an equilibrium in the state equations (disciplinary equations). Unlike z, the state variables are not independent degrees of freedom but depend on the design variables z, the coupling variables y and the state equations. The cases in which x are given by explicit functions of z and y are uncommon in engineering applications. Indeed, these variables are most often defined by implicit functions, that generally require specific optimization methods for solving complex industrial problems.

1.2.3 Types of constraints

The constraints can be divided in two categories :

- g : inequality constraints,

- h : equality constraints.

1.2.4 Types of functions

The different functions used in a MDO problem are :

- $f(x, y, z)$: objective function. This function quantifies the quality of the design and has to be optimized by the MDO process.

- $c(x, y, z)$: coupling functions. These functions are used to compute the coupling variables which come out of a subsystem. We will note $c_{ij}(x_i, y_i, z_i)$ the coupling variables from the subsystem i to the subsystem j.

- $R_i(x_i, y_i, z_i)$: residual functions. The residuals quantify the satisfaction of the state equations (1.4).

- $X_i(y_i, z_i)$: state variable computation functions. These functions yield the roots x_i of the equations (1.4).

1.2.5 Disciplinary equations

The disciplinary (subsystem) equations can be formulated in two ways :

- Non-residual form (explicit):
  $$x_i = X_i(y_i, z_i) \tag{1.5}$$
  In this case, the state variables can be explicitly determined from the design and coupling variables.

- Residual form (implicit) :
  $$R_i(x_i, y_i, z_i) = 0 \tag{1.4}$$
  In this form, there is no explicit relation to determine the state variables from the design and coupling variables.

We can distinguish two methods for handling the disciplinary equations (Fig. 1.2) :

Disciplinary analysis

Definition 1.2.1. *Given the design and coupling variables (respectively z_i and y_i), the disciplinary analysis consists in finding the values of the state variables x_i such that the state (disciplinary) equations $R_i(x_i, y_i, z_i) = 0$ are satisfied.*

Generally, when the disciplinary equations are expressed in the residual form, a specific solver (*e.g.* a Newton algorithm) is used in order to find the roots of the equations (1.4). When the non residual-form is used, the iterative loop is not required in the subsystem because the state variables are directly expressed from the design and coupling variables by explicit relationships. In this case, the calculation scheme consists in sequentially evaluating the different relationships in order to compute the disciplinary outputs (Eq. 1.5).

Disciplinary evaluation

Definition 1.2.2. *Given the design, coupling and state variables (respectively z_i, y_i and x_i), the disciplinary evaluation consists in calculating the value of the residuals $R_i(x_i, y_i, z_i)$.*

In this scheme, the equations (1.4) are not solved. Furthermore, the state variables x are not handled in the subsystem but are considered as inputs in the same way as z and y. The disciplinary evaluation just computes the value of R_i but does not solve the equations $R_i = 0$. Consequently, in case of residual form, this process takes much less computation time than the disciplinary analysis.

As we shall see later, this dichotomy in the way to handle the disciplinary equations is the principal difference between the *All At Once* and the *Individual Discipline Feasible* formulations.

1.2.6 Coupling

From the variables y_i and z_i coming in the subsystem i and the state variables x_i, the coupling variables which come out of the subsystem can be calculated with the coupling functions $c_i(x_i, y_i, z_i)$. The double indexation $c_{ij}(x_i, y_i, z_i)$ denotes that these coupling variables are

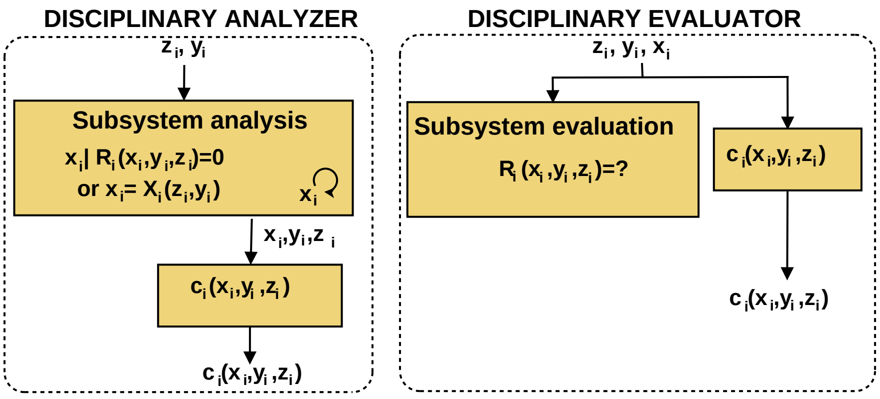


Figure 1.2: Disciplinary analyzer and disciplinary evaluator

transmitted from the i^{th} subsystem to the j^{th} subsystem. The coupling is consistent when the set of coupling variables y_i is equal to the set returned by the different coupling functions (Eq. 1.3).

1.2.7 Multidisciplinary analysis

Definition 1.2.3. *The MultiDisciplinary Analysis (MDA) is a process which aims to satisfy the individual disciplinary feasibilities and the consistency of the couplings between the different subsystems. The MDA consists in finding, for all the subsystems, the variables x_i and y_i such that the state equations are satisfied and the couplings are consistent.*

In other words, the MDA consists in satisfying the system formed by the equations (1.3) and (1.4), as follows :

$$\begin{cases} \forall i \in \{1, \cdots, n\}, \forall j \neq i, y_i = \{c_{ji}(x_j, y_j, z_j)\}_j \\ \forall i, \in \{1, \cdots, n\}, R_i(x_i, y_i, z_i) = 0 \end{cases}$$

A classical way to perform the MDA is to use the *Fixed Point Iteration* (FPI). The FPI generally consists of a loop between the different disciplinary analyses (each of them finding the state variables x in order to make the disciplinary residuals zero). This technique requires some properties of the coupling function and may not converge if the considered function is not contractive [Banach, 1922]. Different numerical schemes of the MDA (Gauss-Seidel and Gauss-Jacobi, *etc.*) are developed in details in [Keane and Nair, 2005].

1.3 Feasibility concepts in MDO

We can distinguish two feasibility concepts in MDO. These concepts are related to the feasibility of either only one subsystem or all the subsystems.

1.3.1 Individual disciplinary feasibility

Definition 1.3.1. *A process is qualified as "individual disciplinary feasible" [Cramer et al., 1994] if at each iteration, the state equations of the different disciplines are satisfied.*

In other words, the individual disciplinary feasibility means that we are always able to find the values of the state variables x which satisfy the equations (1.4) (but the consistency of the couplings is not guaranteed). Trivially, a process in which all the disciplinary equations are in the non-residual form (Eq. 1.5) is intrinsically "individual disciplinary feasible" .

1.3.2 Multidisciplinary feasibility

Definition 1.3.2. *A process is qualified as "multidisciplinary feasible" [Cramer et al., 1994] if at each iteration, the state and coupling variables (respectively x_i and y_i) can be found such that the "individual disciplinary feasibility" is realized and the couplings are consistent.*

A problem in which a MDA is performed is a "multidisciplinary feasible" problem. In this kind of problems, we can express the values of the state variables exclusively in function of the design variables : $x = X_c(z)$ (the subscript $_c$ stands for the consistency of the coupling). Table 1.1 summarizes the level of centralization and the different degrees of feasibility of a MDO process.

Concept	Process used at subsystem level	Variables handled by the subsystems	Variables handled outside the subsystems
No feasibility ensured	Disciplinary evaluation		x,y,z
Individual disciplinary feasibility	Disciplinary analysis	x	y,z
Multidisciplinary feasibility	MDA	x,y	z

Table 1.1: Different feasibility concepts and corresponding involved variables

As we will see in Chapter 2, these two feasibility concepts are the principal difference between the *Multi Discipline Feasible* and the *Individual Discipline Feasible* formulations.

1.4 Description of the Launch Vehicle Design problem

In this section, we briefly present the *Launch Vehicle Design* (LVD) problem. To this end, we describe the different disciplines, variables and constraints involved in such a problem, and we point out the specificities of this design problem with respect to other MDO problems. A launch vehicle is a specific vehicle which aims to put a payload in a certain orbit. A launch vehicle is a dynamical system which moves using rocket propulsion and involves a multi stage architecture. The different stages of the launch vehicle are jettisoned during the flight. These stages can land on Earth (*e.g.* the boosters), enter in the atmosphere (*e.g.* the intermediary stages) or remain in space (*e.g.* the last stage).

1.4.1 Disciplines and variables

The *Launch Vehicle Design* problem is generally decomposed into the different physical disciplines. The classical disciplines involved in such design problems are :

- aerodynamics,
- propulsion,

- structure,
- costs,
- weights & sizing,
- trajectory (performances calculation).

Each of these disciplines involves its own constraints and variables. The design variables z, such as the masses, the diameters, the propulsion variables (chamber pressures, mixture ratios ...), the fairing shape, *etc.* are generally considered at the system level. The trajectory variables (variables of the control law if the trajectory optimization is computed by a direct method or adjoint vector if the optimization uses an indirect method) are usually considered as state variables x (the optimal control techniques will be described in Chapter 3). The optimality conditions associated with these variables can be satisfied at the subsystem level ($R = 0$) or at the system level ($R = 0$ added to the system level equality constraints). In this particular case, the control law variables are at the same level as the design variables z. Typical coupling variables y may be the dry mass, the stage diameters, the thrust to weight ratio, *etc.*

1.4.2 Constraints

The involved equality constraints h in the LVD problem may be composed of the requirements of the mission (desired orbit, payload mass, Gross Lift Off Weight ...) and the inequality constraints g may include for instance the maximum chamber pressure, the maximal angle of attack, the maximum load factor or the minimum nozzle exit pressure. The classical involved coupling constraints may concern the trajectory and the weights & sizing or the trajectory and the aerodynamics, *etc.*

1.4.3 Objective functions

The classical objective functions f are :

- the maximization of the payload mass,
- the minimization of a mass (most often the Gross Lift-Off Weight or the dry mass),
- the minimization of the global or recurring cost.

These objective functions are most often computed by the weight & sizing or the performance calculation (trajectory) subsystems.

1.4.4 Specificities of the Launch Vehicle Design problem

The main difference between the Launch Vehicle Design problem and the other MDO problems concerns the presence of a dynamical system to optimize (trajectory optimization with stage separations). Indeed, the LVD problem combines the coupled optimizations of the design and trajectory variables. The trajectory optimization induces a trajectory simulation with stage separations which is complex to integrate (numerical resolution of a highly non linear ordinary differential equation system). Moreover, the trajectory is subjected to strict equality constraints, due to the mission specifications, which are very difficult to satisfy and considerably limit the feasible search domain (domain for which all the constraints are satisfied). We will study more in details the specificities of the LVD problem when we will describe the application of the MDO methods in LVD in the fourth chapter of this part.

1.5 Conclusion

In this chapter, we have introduced the fundamental notions necessary to describe a MDO process. To this end, the general formulation of a MDO problem and the different variables, functions and constraints involved in such problems have been detailed. From the description of the different ways to handle the disciplinary equations, the different feasibility concepts have been described. Finally, the requirements of the launch vehicle design and the main differences between such design problem and other MDO problems have been briefly described.

The different notations and the concepts introduced in this chapter will be useful in the next chapter which will be devoted to the general description of the main MDO formulations. We will also use these notations in the second part of this document in order to describe our proposed MDO methods.

Chapter 2

Classical MDO methods

Contents

2.1 Introduction

Since the last two decades of the XXth century, a lot of MDO methods have been proposed in the literature. Indeed, many authors such as Sobieski, Braun, Cramer, *etc.* have proposed new MDO formulations in order to more efficiently cope with the engineering problems they have to solve. This chapter is devoted to the description of the main MDO methods present in the literature. We can find many MDO methods which have been applied in a great number of examples. Because the study cases are different, it is difficult to compare these methods in order to determine which is the best. Some review articles [Balling and Sobieszczanski-Sobieski, 1994; Alexandrov and Hussaini, 1995; Sobieszczanski-Sobieski and Haftka, 1997; Agte et al., 2009; Tosserams et al., 2009] provide a state-of-the-art of the different MDO methods. The aim of this chapter is not to make an exhaustive list of the different MDO methods but at first to describe the main MDO methods with common standardized notations introduced in the first chapter. One of the goals of this part is to evaluate the applicability of the expressed methods in launch vehicle design and to bring out the advantages and the drawbacks of these methods with regard to this specific problem.

This chapter is devoted to the description of the MDO methods. To this end, the chapter is organized as follows. In the second section, we describe the single level methods *i.e.* the MDO methods which only require one optimizer. The third section is devoted to the qualitative comparison of these single level methods. The fourth section concerns the description of the main multi level methods (involving more than one optimizer). For each of the described methods, we expose at first the principle of the method, then its mathematical formulation accompanied by an explanatory scheme, and finally we present the advantages and drawbacks of the considered MDO method.

2.2 Single level methods

2.2.1 Multi Discipline Feasible

Principle

The *Multi Discipline Feasible* (MDF) method is the most usual MDO method. It is also called “Nested Analysis and Design” (NAND), “Single NAND-NAND” (SNN) and “All-in-One”. This method is explained in [Balling and Sobieszczanski-Sobieski, 1994; Cramer et al., 1994; Kodiyalam, 1998; Allison, 2004; Gang et al., 2005]. The architecture of the MDF method is similar to the architecture of a classical optimization problem which involves only one subsystem. The main difference is that, in MDF, the subsystem is replaced by a complete multidisciplinary analysis which is performed at each iteration of the optimization process. Thus, all the subsystems are coupled in an analysis module which ensures the multidisciplinary feasibility of the solution at each iteration (Fig. 2.1). In this formulation, the set of design variables is transmitted to the analysis module. This module executes, by a dedicated method (*e.g.* Fixed Point Iteration or Newton method), the multidisciplinary analysis of the system (1.3)-(1.4).

Formulation I.2 (**MDF formulation**)

$$
\begin{aligned}
\textbf{Minimize} \quad & f(X_c(z), y, z) \\
\textbf{With respect to} \quad & z \in \mathcal{Z} \\
\textbf{Subject to} \quad & g(X_c(z), y, z) \leq 0 && (2.1) \\
& h(X_c(z), y, z) = 0 && (2.2)
\end{aligned}
$$

Once the MDA is performed, the analysis module output vector is used by the system level optimizer to compute the objective function and the constraints. The process is repeated at each iteration. In this method, each set of found design variables is a consistent configuration. Furthermore, the disciplines are in charge of finding their local variables x to satisfy their own equations (1.4) (disciplinary analyzers are used). In this manner, the state variables do not intervene in the optimization problem formulation because they are totally handled by the disciplinary analyzers during the MDA at the subsystem level. In the

MDF method, each found solution is multidisciplinary feasible (*i.e.* individual disciplinary equations (1.4) as well as coupling equations (1.3) are satisfied). We can note that the term "multidisciplinary feasible" does not imply the satisfaction of design constraints g and h, but only the MDO ones.

Some systems do not present any feedback between the different subsystems (*i.e.* [Duranté et al., 2004], [Castellini et al., 2010]). In this case, a sequential analysis, without iteration, is possible and the MDF method is the most natural method to solve this kind of problems. Indeed, for these problems, the coupling constraints (Eq. 1.3) are automatically satisfied by the sequential calculation process.

For large scale (industrial) application cases, the different subsystem disciplinary analyses can be composed of groups of specialists (each of them potentially located in a different place all over the world). In this case, the MDA becomes a very complex task and includes :

- transmission of informations between the different groups of specialists (dotted lines in Figure 2.1), and not only between computers or optimization programs,
- management requirements between and inside the different groups (because each group potentially converses with all the others),
- definition of each group action domain and autonomy with respect to the community (in order to ensure as much as possible the consistency of the trade-off establishments).

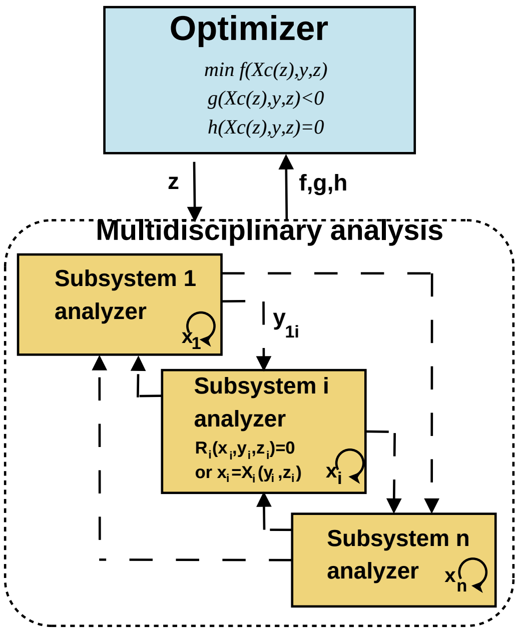


Figure 2.1: MDF method

Advantages and drawbacks of MDF

The main advantage of the MDF method is its simplicity. Indeed, a limited number of optimization variables is used (only the design variables z have to be handled by the optimizer) and classical disciplinary analyzers are used. The method implementation is relatively easy since the system decomposition is not required. Moreover, if the optimization process is stopped, the found solution is consistent with respect to the couplings and the individual disciplinary feasibilities, even if it is not the optimal one.

The MDF method presents important drawbacks. Indeed, the calculation cost is very important and the method does not take advantages of the couplings between the disciplines in order to improve the optimization process, because the optimizer does not control the choices performed by the MDA, which is considered as a black box. The calculation cost is due to the MDA which has to be executed at each iteration of the optimization process. This analysis considers all the subsystems present in the MDO process (the MDF method modularity is very poor). Thus, when a design variable changes, all the other variables have to be recalculated. Furthermore, a gradient-based algorithm used as the optimizer will need a full MDA whenever the derivatives required to compute the gradient or the Hessian will have to be calculated [Kodiyalam, 1998]. Moreover, if the MDA is performed by a FPI, the MDA may not converge if the considered function is not contractive. Consequently, MDF is principally applicable to the optimization problems for which the different subsystems can be quickly evaluated during the MDA or for which the MDA converges in a few iterations.

For large scale application problems (in which the subsystems are engineering teams), at each iteration, the MDA requires a lot of information transmissions and management tasks because each group has to potentially converse with all the others. The MDA is also responsible for defining the action domain of each specialty group (possible conflict resolutions with engineers who want to decide design and to have as much autonomy as possible). Moreover, each group of specialists has to wait for the previous one in order to perform its task (when FPI is used), that can be very time consuming.

2.2.2 Individual Discipline Feasible

Principle

The *Individual Discipline Feasible* (IDF) method [Cramer et al., 1994; Martins and Marriage, 2007], also called “Optimizer-Based-Decomposition” (OBD) [Kroo, 2004], “Single-SAND-NAND” (SSN) [Balling and Sobieszczanski-Sobieski, 1994], allows to avoid a complete multidisciplinary analysis at each iteration of the design process. Like MDF, a single optimizer at the system level is used and disciplinary analysis blocks are called in the different subsystems. The main difference between MDF and IDF is that for IDF, the optimizer is also responsible for the coordination of the different subsystems and uses additional variables (coupling variables y) to ensure it. At each iteration of the optimization process, the different subsystems are individually feasible but the consistency of the couplings between them is not guaranteed. The IDF formulation of a MDO problem may be summarized as follows :

Formulation I.3 (**IDF formulation**)

$$
\begin{aligned}
\textbf{Minimize} \quad & f(X(y,z),y,z) \\
\textbf{With respect to} \quad & (y,z) \in \mathcal{Y} \times \mathcal{Z} \\
\textbf{Subject to} \quad & g(X(y,z),y,z) \leq 0 && (2.3) \\
& h(X(y,z),y,z) = 0 && (2.4) \\
\forall i \in \{1,\cdots,n\}, \forall j \neq i, y_i = & \{c_{ji}(X_j(y_j,z_j,),y_j,z_j)\}_j && (2.5)
\end{aligned}
$$

This method allows to break up the main problem into several subsystems. The coupling variables (and the associate coupling equations (1.3)) are introduced in order to evaluate the consistency of the results found by the different subsystem analyzers. In IDF, the consistency of the solution (multidisciplinary feasibility Eq. 1.3) is not ensured at each iteration but only at the convergence. Consequently, the IDF process should not be stopped before the convergence is reached.
This decomposition method considerably increases the number of variables handled at the system level but allows to improve the computational speed (parallelization is possible). Therefore, unlike the MDF method, a single analysis is performed at each iteration in the different subsystems. The centralization degree of the IDF method is more important than the MDF one. There again, the subsystems determine their state variables (subsystem analyzers are used).

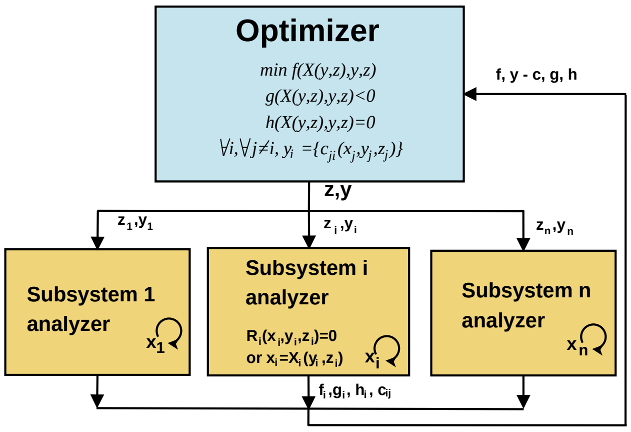


Figure 2.2: IDF method

Advantages and drawbacks of IDF

If the number of coupling variables is relatively small, the IDF method is applicable and provides results in a limited computational time. Since the principle of this method is to introduce additional coupling variables and constraints at the system level, the number of iterations at the system level is more important than in MDF. The optimizer has to dialog with each disciplinary block and transmits to each of them its own coupling variables.

In return, the IDF method presents the advantage of being implementable in a network (parallelization is possible). This particularity can considerably improve the efficiency of the method (calculation time is reduced). Furthermore, since the MDA process is removed, the internal analysis loop is broken up. In this way, if the system level optimizer requires sensitivity calculations of the ith subsystem, the other subsystems do not intervene and no computation time is wasted. Therefore, if an important centralization of the optimization process is possible, the IDF method can be employed to effectively solve a MDO problem.

For large scale applications (for which the subsystems are composed of engineers), the management tasks are less important than in MDF because each group just converses with the coordinator and not with all the others. Indeed, since the MDA is broken, the different engineering teams have not to wait for the results of other teams, that allows to parallelize the human tasks. However, the autonomy of the different teams is limited because the multi disciplinary feasibility is not ensured at the subsystem level, but at the system level.

2.2.3 All At Once

Principle

The *All At Once* (AAO) method, also called "Single-SAND-SAND" (SSS) [Balling and Sobieszczanski-Sobieski, 1994; Balling and Wilkinson, 1997] solves simultaneously the optimization problem and the equations of the different subsystems [Allison, 2004]. The equations (2.8), (2.9) are not satisfied at each iteration of the optimization process but they have to be at the convergence [Cramer et al., 1994] (the design configuration is consistent only at the convergence). In this method, the subsystem equations (residuals) are considered as equality constraints $R = 0$.

AAO is the most elementary MDO method. The control of the process is assigned to a system level optimizer which aims to optimize a global objective and calls subsystem evaluations. The optimizer handles the design variables z, the coupling variables y and the state variables x. At the subsystem level, the disciplinary analyzers are replaced by disciplinary evaluators. The design and the evaluations at the subsystem and system levels are performed at the same time. Therefore, the centralization of the problem is more important than in IDF and MDF. The AAO formulation of the MDO problem can be summarized as follows :

Formulation I.4 (**AAO formulation**)

$$\begin{aligned}
\textbf{Minimize} \quad & f(x,y,z) \\
\textbf{With respect to} \quad & (x,y,z) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{Z} \\
\textbf{Subject to} \quad & g(x,y,z) \leq 0 && (2.6) \\
& h(x,y,z) = 0 && (2.7) \\
& \forall i \in \{1,\cdots,n\}, R_i(x_i,y_i,z_i) = 0 && (2.8) \\
& \forall i \in \{1,\cdots,n\}, \forall j \neq i, y_i = \{c_{ji}(x_j,y_j,z_j)\}_j && (2.9)
\end{aligned}$$

The global state vector is divided into subvectors which are distributed to each subsystem. The residuals R are transmitted to the global optimizer simultaneously with the other variables. Since the residuals are equal to zero only at the optimum, the multidisciplinary and the individual disciplinary feasibilities are not ensured at the intermediary points.

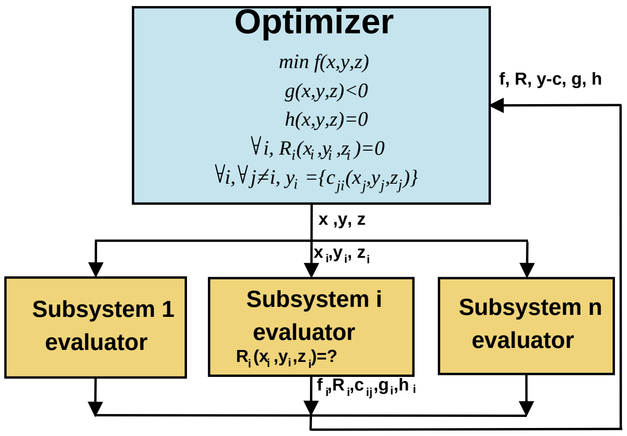


Figure 2.3: AAO method

Advantages and drawbacks of AAO

The AAO method is the less complex method to solve MDO problems because it does not make any difference between the different involved variables which are all handled at the same level. Nevertheless, AAO presents some drawbacks which make it inapplicable to large scale launch vehicle design problems. Indeed, in the case of problems which consider several complex subsystems, the number of variables handled by the optimizer soars, that makes AAO not applicable.

In the situations for which the convergence is not achieved, the AAO method (like IDF) does not provide a feasible design. Moreover, the AAO method is difficult to implement because highly centralized. This method may present some robustness difficulties because the formulation results in a very constrained MDO problem. In relatively small problems, AAO, like IDF, is applicable and allows to parallelize calculations. Thus, the computation time may be considerably reduced.

For large scale applications, the different groups of engineers do not perform the individual disciplinary analysis of the different subsystems but are just reduced to make simple calculations with respect to the instructions given by the system level. Indeed, simple computations of the residuals are involved at the subsystem level, that do not require the intervention of engineering teams at this level. For this reason, the AAO method is just applicable to small problems for which all the calculations can be performed by a small group of generalist engineers.

2.3 Qualitative comparison of the single level methods

Cramer *et al.* [Cramer et al., 1994] have performed a qualitative comparison of the MDF, IDF and AAO methods. Table (2.1) sums up this comparison and adds some considerations concerning the large scale applications. In this table, the term “decision” stands for the action on the design variables z.

	AAO	**IDF**	**MDF**
Use of traditional (engineering) analysis tools	No	Yes	Yes, coupling is required
Individual disciplinary feasibility of the solution	Only at convergence	At each iteration	At each iteration
Multidisciplinary feasibility of the solution	Only at convergence	Only at convergence	At each iteration
Variables handled by the optimizer	x, y, z	y, z	z
Convergence speed expected	Fast	Medium	Slow
- Large scale applications -			
Decomposition into concurrent data processes	Yes	Yes	No
Decomposition into concurrent human tasks	Yes, but not judicious	Yes	No
Autonomy of engineering teams at the subsystem level	None No decision No analysis	Small No decision Ind. discip. analysis	Moderate No decision Multi discip. analysis

Table 2.1: Qualitative synthesis of the single level methods

2.4 Multi level methods

After having described in the previous section the three main single level MDO methods, we present in this section the MDO multi level methods *i.e.* the MDO methods which present more than one optimizer.

2.4.1 Collaborative Optimization

Principle

The *Collaborative Optimization* (CO) method, initially developed by Braun *et al.* [Braun and Kroo, 1995], is a bi level optimization method. This method has been elaborated to give more autonomy to the different subsystems in order to satisfy the compatibility constraints between the different subsystems. The optimization problem is subdivided into disciplinary subproblems. In each subproblem, the local optimizer :

- controls its local design variables,

- is responsible for the satisfaction of its local constraints,
- does not have knowledge of the variables and the constraints of the other subsystems.

Each disciplinary optimizer modifies its variables in order to find an agreement with the other subsystems with regard to the coupling variables. To this end, specific subsystem level objective functions representing the relative errors between the disciplinary outputs and the system level instructions are introduced. At the system level, an optimizer is in charge of the coordination of the whole process and optimizes a global objective function. The main idea of this method is that the disciplinary experts can intervene in the subsystems without being constrained by the other subsystems. The subsystems can use either subsystem analyzers or subsystem evaluators to compute their local equations. The CO formulation of the MDO problem may be summarized as follows :

Formulation I.5 (**First CO formulation** [Braun and Kroo, 1995])

At the system level :

$$\begin{aligned} \textbf{Minimize} \quad & f(y,z) \\ \textbf{With respect to} \quad & (y,z) \in \mathcal{Y} \times \mathcal{Z} \\ \textbf{Subject to} \quad & \forall i \in \{1,\cdots,n\}, J_i^*(z^*, z, y_i, c_i(y_i, z_i^*)) = 0 \end{aligned} \tag{2.10}$$

J_i^* is the optimized objective function of the i^{th} subsystem and z^* the local copies of z. For the i^{th} subsystem, we have the following subproblem :

$$\begin{aligned} \textbf{Minimize} \quad & J_i(z_i, z_i^*, y_i, c_i(y_i, z_i^*)) = \|z_i^* - z_i\|_2^2 + \|y_{ij} - c_{ij}(y_i, z_i^*)\|_2^2 \\ \textbf{With respect to} \quad & z_i^* \in \mathcal{Z}_i \\ \textbf{Subject to} \quad & g_i(y_i, z_i^*) \leq 0 \qquad (2.11) \\ & h_i(y_i, z_i^*) = 0 \qquad (2.12) \end{aligned}$$

y_{ij} are the coupling variables from the i^{th} subsystem to the j^{th} one.

Formulation I.6 (**Second CO formulation**)

The second formulation [Alexandrov and Lewis, 2000] consists in decomposing the system level constraint vector $J_i^* = 0$ into two sets of constraints :

$$\begin{aligned} \textbf{Minimize} \quad & f(y,z) \\ \textbf{With respect to} \quad & (y,z) \in \mathcal{Y} \times \mathcal{Z} \\ \textbf{Subject to} \quad & \forall i \in \{1,\cdots,n\}, J_{i1}^*(z_i, z_i^*) = z_i^* - z_i = 0 \qquad (2.13) \\ & \forall i \in \{1,\cdots,n\}, \forall j \neq i, J_{i2j}^*(y_{ji}, c_{ji}(y_i, z_i^*)) = y_{ji} - c_{ji}(y_i, z_i^*) = 0 \qquad (2.14) \end{aligned}$$

The optimization problem at the subsystem level remains unchanged :

$$\begin{aligned} \textbf{Minimize} \quad & J_i(z_i, z_i^*, y_i, c_i(y_i, z_i^*)) = \|z_i^* - z_i\|_2^2 + \|y_{ij} - c_{ij}(y_i, z_i^*)\|_2^2 \\ \textbf{With respect to} \quad & z_i^* \in \mathcal{Z}_i \\ \textbf{Subject to} \quad & g_i(y_i, z_i^*) \leq 0 \qquad (2.15) \\ & h_i(y_i, z_i^*) = 0 \qquad (2.16) \end{aligned}$$

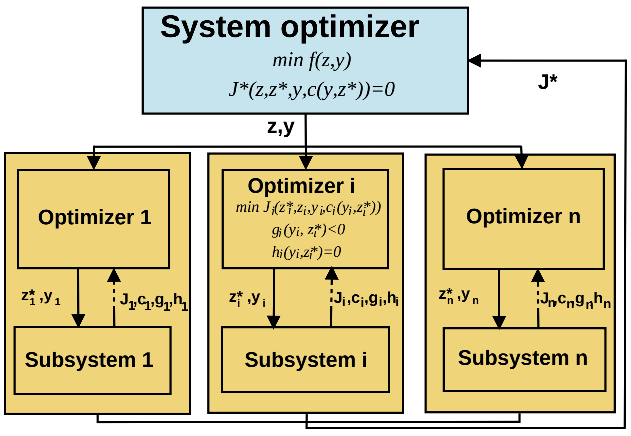


Figure 2.4: CO method

Advantages and drawbacks of CO

CO presents significant advantages with respect to the single level optimization methods. Indeed, it is not necessary to modify the disciplinary codes in order to integrate a discipline into the CO optimization scheme (the modularity of the method is improved). Furthermore, CO allows to use the optimization methods that are the most adapted to each subproblem, with possible actions of disciplinary experts. This method also allows to add or to modify some subsystems without changing the whole design process. Finally, CO is very flexible and the subsystems calculation efficiency can be improved by using response surfaces in addition to the local optimizers [Sobieski and Kroo, 2000].

Unfortunately, practical and theoretical issues with regard to the convergence of CO when using a sum-squared formulation for local J have been observed by some researchers [Cormier et al., 2000; Alexandrov and Lewis, 2000]. Indeed, CO seems not to be robust because it generates instabilities at the convergence and may fail to converge on some simple test problems [Alexandrov and Lewis, 2000]. To overcome this difficulty, DeMiguel *et al.* have proposed other formulations (Inexact Penalty Decomposition, Exact Penalty Decomposition [DeMiguel and Murray, 2006], Modified Collaborative Optimization) which ensure the equivalence of Karush-Kuhn-Tucker (KKT) conditions with respect to the initial MDO problem. Moreover, as the number of coupling variables increases, the CO method efficiency decreases. In other words, if the original MDO problem has a great number of coupling variables, CO can be inefficient in terms of calculation time. Nevertheless, this approach has been shown to be valid and to provide good results for some MDO problems [Braun et al., 1996].

2.4.2 Modified Collaborative Optimization

Principle

The *Modified Collaborative Optimization* (MCO) method, described in [DeMiguel and Murray, 2000; DeMiguel, 2001], presents the same architecture as CO but tries to correct some drawbacks of the CO method (*e.g.* non differentiability problems). MCO replaces the quadratic penalty function used in CO by an exact form. Indeed, in order to avoid the instabilities induced by the quadratic penalty function, MCO changes the quadratic form of local objective into an IDF form with adding elastic variables (s_i and t_i) [Gill et al., 1981]. Furthermore, the equality constraints at the system level are replaced by a penalty function. The MCO formulation of the MDO problem at the system level is :

Formulation I.7 (**MCO formulation**)

$$\begin{array}{rc} \textbf{Minimize} & f(z,y) + w\sum_{i=1}^{n} J_i^*(z,y) \\ \textbf{With respect to} & (y,z) \in \mathcal{Y} \times \mathcal{Z} \end{array}$$

The problem formulation of the ith subsystem is :

$$\begin{array}{rc} \textbf{Minimize} & J_i = \sum (s_i + t_i) \equiv \|z_i^* - z_i\|_1 + \|y_{ij} - c_{ij}\|_1 \\ \textbf{With respect to} & z_i^* \in \mathcal{Z}_i^*,\ s_i = \{z_{si}, c_{si}\} \in \mathcal{S}_i,\ t_i = \{z_{ti}, c_{ti}\} \in \mathcal{T}_i,\ r_i \in \mathcal{R}_i \\ \textbf{Subject to} & h_i(y_i, z_i^*) = 0 \qquad (2.17) \\ & g_i(y_i, z_i^*) + r_i = 0 \qquad (2.18) \\ & z_i^* + z_{si} - z_{ti} = z_i \qquad (2.19) \\ & c_{ij} + c_{si} - c_{ti} = y_i \qquad (2.20) \\ & s_i, t_i, r_i \geq 0 \qquad (2.21) \end{array}$$

with :

- r_i : slack variables added to transform the inequality constraints $g \leq 0$ in equality ones;
- (s_i, t_i) : elastic variables [Gill et al., 1981; Boman, 1999] (s_i, t_i=0 at the convergence);
- w : penalty coefficient (weight).

The algorithm of MCO is the same as the one of the CO method, apart from the fact that at the subsystem level, the different optimizers have to handle the slack variables and the associated constraints.

Advantages and drawbacks of MCO

MCO presents theoretically some advantages in comparison with the CO method. Indeed, MCO avoids instabilities of CO and aims to improve the convergence properties. As we shall see later, MCO has been tested in a *Launch Vehicle Design* problem [Brown and Olds, 2006]. Unfortunately, the results of this study do not allow us to conclude about the efficiency of this method in this specific application case.

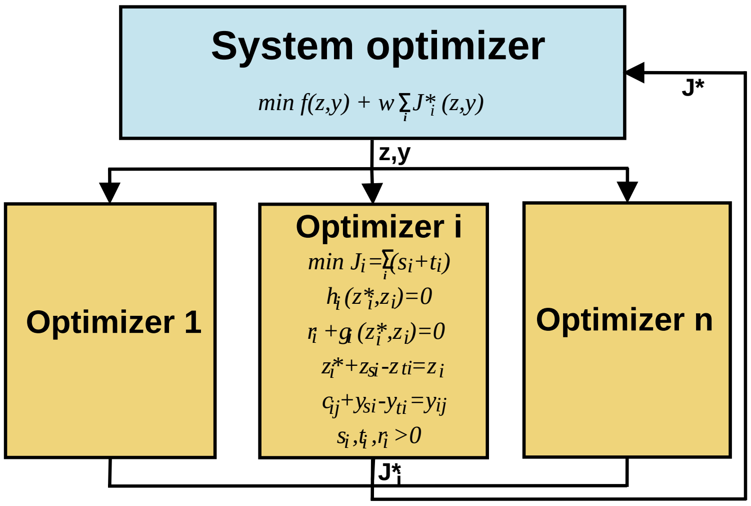


Figure 2.5: MCO method

2.4.3 Concurrent SubSpace Optimization

Principle

The *Concurrent SubSpace Optimization* (CSSO) method was formulated by Sobieszczanski-Sobieski [Sobieszczanski-Sobieski, 1988a]. This iterative method is also based on a system decomposition strategy which allows the subsystems to contribute independently to the optimization process. The global problem is solved by a system level optimizer which ensures the coordination of the different subsystems and aims to find a compromise between the different solutions proposed at the subsystem level.

Approximations of the coupling variables are used in the different subsystems in order to determine their influence on the objective f and on the constraints g and h. With this method, when performing the subsystem optimizations, the effects of a variable variation in one subsystem to the constraints of the other subsystems can be determined. This method introduces a concept of responsibility for the constraint violation and uses cumulative constraints [Sobieszczanski-Sobieski, 1988a], which consist in considering the partial satisfaction of a constraint in one discipline by the influences of the other disciplines. This concept is made possible by introducing additional (coordinating) variables to the initial problem.

The approximations of the coupling variables can be performed by using neural networks [Sellar and Batill, 1996] or response surfaces [Renaud and Gabriele, 1993, 1994; Sellar et al., 1996; Wujek et al., 1996, 1997]. Other expansions of the CSSO method using approximate models [Rodriguez et al., 1998, 2001; Perez et al., 2002] can be found in the literature but are not developed in this document. The CSSO method uses a multidisciplinary analysis to coordinate the optimization process and often performs sensitivity analysis by using the *Global Sensitivity Equation* (GSE). The resolution of the GSE [Sobieszczanski-Sobieski, 1990] allows to quickly obtain the total influence of the different variables to the objective function. This technique will be described in Chapter 8 and Appendix C.

Formulation I.8 (**CSSO formulation**)

The system level optimizer solves the following problem :

$$\begin{aligned} \textbf{Minimize} \quad & f(\tilde{y}, z_{sh}) \\ \textbf{With respect to} \quad & z_{sh} \in \mathcal{Z}_{sh} \\ \textbf{Subject to} \quad & g(\tilde{y}, z_{sh}) \leq 0 \qquad (2.22) \\ & h(\tilde{y}, z_{sh}) = 0 \qquad (2.23) \end{aligned}$$

where $\tilde{y}$ represent the approximations of the coupling variables. The optimization problem of the i^{th} subsystem can be formulated as follows :

$$\begin{aligned} \textbf{Minimize} \quad & f(\tilde{y}_{ji}, z_{sh}, \bar{z}_i), \ j \neq i \\ \textbf{With respect to} \quad & \bar{z}_i \in \bar{\mathcal{Z}}_i \\ \textbf{Subject to} \quad & g_i(\tilde{y}_{ji}, z_{sh}, \bar{z}_i) \leq 0 \qquad (2.24) \\ & h_i(\tilde{y}_{ji}, z_{sh}, \bar{z}_i) = 0 \qquad (2.25) \end{aligned}$$

Unlike the CO method, the shared design variables are considered as constants during the concurrent optimizations at the subsystem level. In [Huang and Bloebaum, 2004], a CSSO-based method is presented : the Multi-Objective Pareto Concurrent SubSpace Optimization (MOPCSSO). MOPCSSO has been developed to solve multi-objective problem with a CSSO architecture and integrates the concept of Pareto Optimality [Pareto, 1971]. This method allows to solve multi-objective large scale problems with a CSSO-based method.

Advantages and drawbacks of CSSO

The main characteristic of the CSSO method is the use of approximate disciplinary models to estimate the effects of the variables of the other disciplines and to solve the problem as a decoupled one. These approximate linearized models create a database which is used by the local optimizers in order to optimize the objectives and to satisfy the constraints. In that way, CSSO method can reduce the calculation time of the optimization process. In brief, if the problem is relatively small and the model approximations are easy to formulate, CSSO can be very efficient and gives solutions in a reduced calculation time.

Unfortunately, the efficiency of the CSSO method highly depends on the approximate models of the coupling variables. Moreover, for large scale problems, the necessary time to build these models can be longer than the time saved by using them. This feature can make CSSO not enough attractive with regard to the other MDO methods.

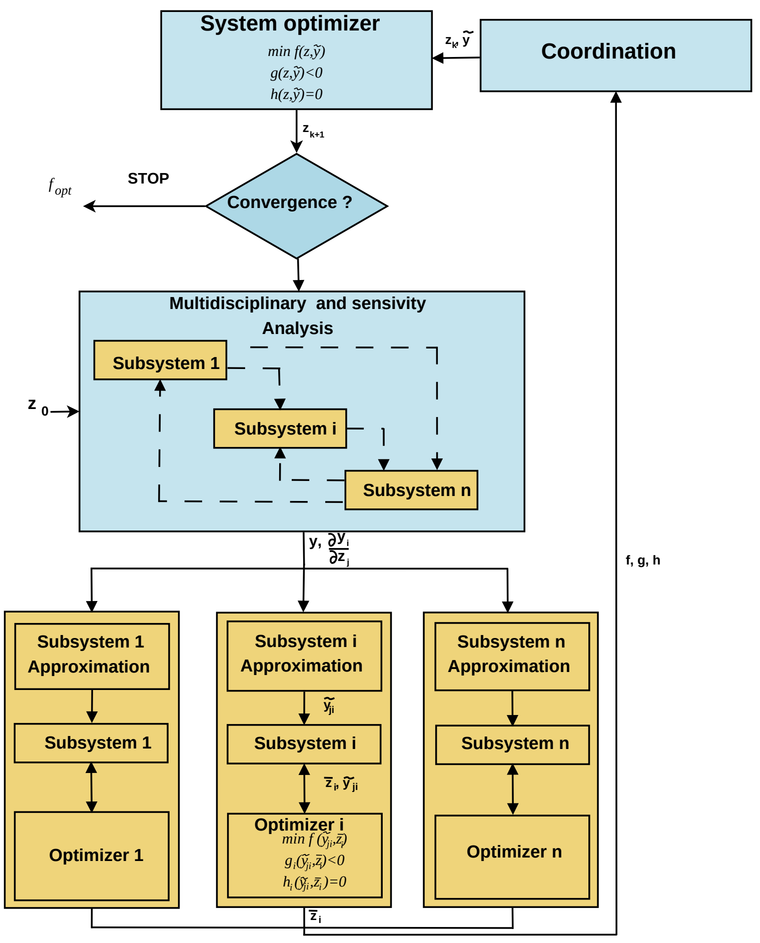


Figure 2.6: CSSO method

2.4.4 Bi-Level Integrated Systems Synthesis

Principle

The *Bi-Level Integrated System Synthesis* (BLISS) method was initially proposed by Sobieszczanski-Sobieski [Sobieszczanski-Sobieski et al., 1998, 2000]. In this document, BLISS and BLISS2000 [Sobieszczanski-Sobieski et al., 2003] are described. BLISS2000 aims to improve the efficiency of the classical BLISS method by using response surfaces and other approximate models.

BLISS (Fig. 2.7) is an iterative multi level method which is organized into a global optimizer at the system level and a set of disciplinary optimizers at the subsystem level. This method resorts to a system analysis and subsystem analyses in order to improve the shared and individual design variables $z_{sh}, \bar{z}_i$. The method is based on a gradient approach and optimizes successively the contributions of the shared and individual design variables to the objective function.

Formulation I.9 (**BLISS formulation**)

At the k^{th} iteration, the system level optimizer solves the following problem :

$$\begin{array}{rc} \textbf{Given} & f_k, z_{sh_k} \\ \textbf{Minimize} & f_k + \frac{\partial f}{\partial z_{sh}} \Delta z_{sh} \\ \textbf{With respect to} & \Delta z_{sh} \end{array}$$

The i^{th} subsystem solves the problem :

$$\begin{array}{rc} \textbf{Given} & z_{sh}, \bar{z}_i, y_i \\ \textbf{Minimize} & \frac{\partial f}{\partial \bar{z}_i} \Delta \bar{z}_i \\ \textbf{With respect to} & \Delta \bar{z}_i \\ \textbf{Subject to} & g(\bar{z}_i, z_{zh}, y_i) \leq 0 \qquad (2.26) \\ & h(\bar{z}_i, z_{sh}, y_i) = 0 \qquad (2.27) \end{array}$$

with Δz the optimization variable increments at the current iteration. In BLISS2000, weighting factors w are added to structure the set of disciplinary outputs o_i. Here, we denote by output o all the variables which come from a disciplinary box. The set of output variables contains the coupling variables c and the state variables x which result from the subsystem optimizations. The objective of each subsystem is to minimize a weighted sum of their outputs, considering the global variables, the coupling variables and the weights as constants. In BLISS 2000, local optimizers are replaced by sets of response surfaces which approximate the optimized outputs c from the inputs given by the global optimizer. All along the optimization process, response surfaces can be improved by adding and discarding some points in order to estimate the behavior of the subsystems (variables $\tilde{o}_i$).

Formulation I.10 (**BLISS2000 formulation**)

The problem at the system level can be formulated as follows :

$$\begin{array}{rc} \textbf{Minimize} & f(z, y, w) = \tilde{o}_j \\ \textbf{With respect to} & (y, z, w) \in \mathcal{Y} \times \mathcal{Z} \times \mathcal{W} \\ \textbf{Subject to} & \forall i \in \{1, \cdots, n\}, \forall j \neq i, y_i = \{\tilde{c}_{ji}\}_j \qquad (2.28) \end{array}$$

For the ith local optimizer, the optimization problem is the following :

$$\begin{array}{rc} \textbf{Minimize} & \sum_k w_{ik}.o_{ik} \\ \textbf{With respect to} & \bar{z}_i \in \bar{\mathcal{Z}}_i \\ \textbf{Subject to} & g_i(y_i, \bar{z}_i) \leq 0 \qquad (2.29) \\ & h_i(y_i, \bar{z}_i) = 0 \qquad (2.30) \end{array}$$

The variables z_{sh}, y, w are considered as constants during the subsystem optimization.

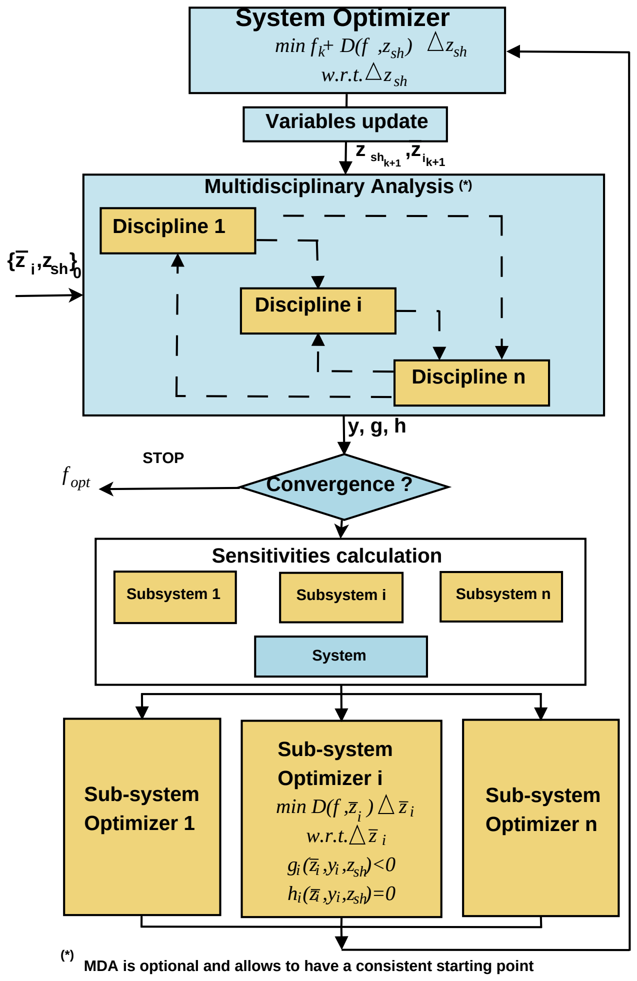


Figure 2.7: BLISS method

Advantages and drawbacks of BLISS

The main advantage of the BLISS method is to separate the system level optimization and the optimizations of the different subsystems. Thus, the use of specific optimization tools for each subsystem is possible, because the disciplines are considered as black boxes by the system level optimizer, which has the knowledge of only the outputs of the different subsystems.

The BLISS and BLISS2000 methods, like all the gradient-based iterative methods, require limits on the optimization variables. Indeed, these methods will not be able to converge if the search space is too large or poorly defined. Like all the processes using approximate models, the efficiency of BLISS2000 is highly dependent of the quality of the response surfaces which approximate the solutions of the optimization subproblems. Nevertheless, these models can be generated off-line by specialists who make them more stable to the design variable variations. If the optimization subproblems are non smooth and the associate response surfaces have to be often redefined, BLISS2000 may provide not accurate results with an explosion of calculation time. BLISS prefers to handle a small number of global design variables and gives results in limited calculation time for well decomposed problems.

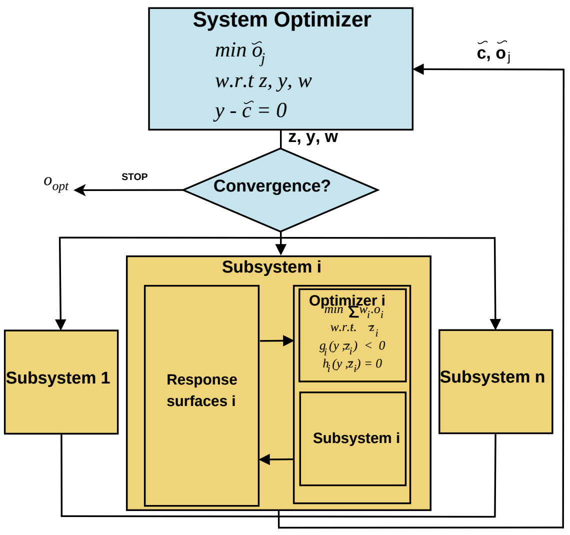


Figure 2.8: BLISS2000 method

2.4.5 Analytical Target Cascading

Principle

The *Analytical Target Cascading* (ATC) method, described in [Michelena et al., 1999, 2003; Kim, 2001], has been initially developed to formalize industrial product development processes. This formulation is very suitable to solve problems with a hierarchical structure. ATC is a multi level MDO method (possibly involving more than 2 levels) which hierarchically propagates system and subsystem level targets through the different subsystem levels. In this formulation, the initial problem is subdivided into a set of subproblems. The specified design targets are “cascaded” from the system level to the lower levels and are also “rebalanced” to the higher levels after being optimized at the lower levels.

At each level of the design process, a specific optimization problem is formulated to minimize the errors between the level outputs and the propagated objectives, and thus to ensure the consistency concerning the couplings between the upper and lower optimization levels. For some problems, the mathematical formulation of ATC can be similar to the CO one [Allison et al., 2005].

Let S_{ij} be the j^{th} subsystem of the i^{th} optimization level, the optimization problem to solve for this subsystem is the following :

Formulation I.11 (**ATC formulation**)

Minimize $$f_{ij} = \|C_{ij} - y_{(i-1)j}\| + \|z_{sh_{(i-1)j}} - z^*_{sh_{(i-1)j}}\| + \epsilon_{C_{ij}} + \epsilon_{Z_{ij}}$$

With respect to $$(\bar{z}_{ij}, z_{sh_{ij}}, z^*_{sh_{(i-1)j}}, y_{ij}, \epsilon_{C_{ij}}, \epsilon_{Z_{ij}}) \in \bar{\mathcal{Z}}_i \times \mathcal{Z}_{sh_i} \times \mathcal{Z}^*_{sh_{i-1}} \times \mathcal{Y}_i \times \mathcal{E}_{C_i} \times \mathcal{E}_{Z_i}$$

Where $$C_{ij} = c_{ij}(y_{ij}, \bar{z}_{ij}, z^*_{sh_{(i-1)j}}) \tag{2.31}$$

Subject to

$$\sum_{k \in Child_{ij}} \|y_{ij_k} - C_{(i+1)j_k}\| \leq \epsilon_{C_{ij}} \tag{2.32}$$

$$\sum_{k \in Child_{ij}} \|z_{sh_{ij_k}} - z^*_{sh_{ij_k}}\| \leq \epsilon_{Z_{ij}} \tag{2.33}$$

$$g_{ij}(c_{ij}, \bar{z}_{ij}, z^*_{sh_{(i-1)j}}) \leq 0 \tag{2.34}$$

$$h_{ij}(c_{ij}, \bar{z}_{ij}, z^*_{sh_{(i-1)j}}) = 0 \tag{2.35}$$

with $\bar{z}_{ij}$ the design variables of S_{ij}, $z_{sh_{ij}}$ the shared design variables of S_{ij}, $z^*_{sh_{(i-1)j}}$ the local copies of the S_{ij}'s parent shared design variables, C_{ij} the responses of S_{ij}, $C_{(i+1)j_k}$ the responses of the k^{th} S_{ij}'s child, y_{ij} the coupling variables of S_{ij}, $y_{(i-1)j}$ the S_{ij}'s parent coupling variables, $\epsilon_{C_{ij}}$ and $\epsilon_{Z_{ij}}$ the relative tolerances on the satisfaction of the inequality constraints (2.34),(2.35). $Child_{ij}$ stands for the set of S_{ij}'s children.
For the top level, the objective function does not involve the satisfaction of the coupling constraints and the term $y_{(i-1)j}$ is replaced by the real target to reach (the variables $z^*_{sh_{(i-1)j}}$ are not necessary). In the same way, at the bottom level, the equations (2.32) and (2.33) are not necessary and the bottom level subsystem optimizers only involve the variables $\bar{z}_{ij}, z^*_{sh_{(i-1)j}}$.

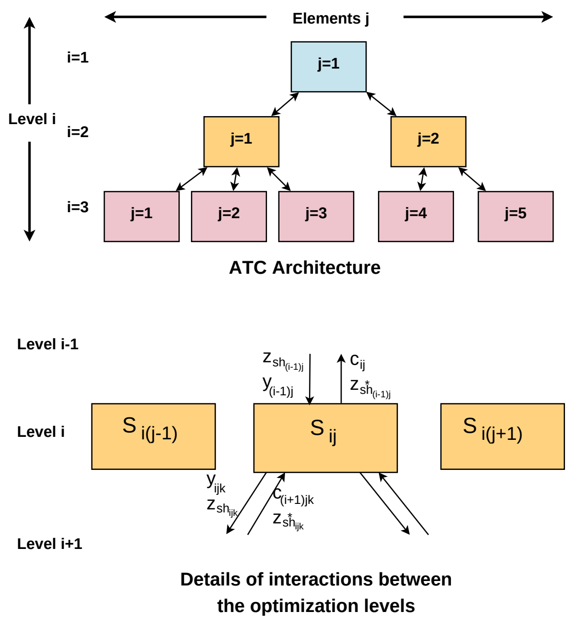


Figure 2.9: ATC method

Advantages and drawbacks of ATC

ATC is a generic formulation adapted to the large scale problems which can be solved with a multi level structure. By partitioning the MDO process into a series of levels, ATC allows to distribute the complexity of the MDO problem into the different subsystems present in the different optimization levels. For that reason, ATC is adapted for the MDO problem which can be decomposed into many small subproblems. ATC has been improved using Lagrangian coordination [Kim et al., 2006] and has been recently adapted to non-hierarchical formulation [Tosserams et al., 2010]. Convergence proof and parallelization processes of ATC have been proposed [Michelena et al., 2003; Han and Papalambros, 2010].

2.4.6 Other multi level methods

Some other MDO methods have been proposed in the literature. Among these, we can cite the Decomposition Based Design method [Lee and Jong, 2006], the Exact and Inexact Penalty decomposition methods [DeMiguel, 2001; DeMiguel and Murray, 2006], the Distributed Optimization [Tribes et al., 2005], *etc.* In this section, we just describe three of them which appear as relatively promising for the LVD application.

Discipline Interaction Variable Elimination

The *Discipline Interaction Variable Elimination* (DIVE), described in [Masmoudi and Parte, 2006; Masmoudi et al., 2008; Auroux et al., 2009; Clement, 2009], is derived from BLISS-2000 [Agte, 2005] and aims to simplify the optimization problem by separately handling the global, local and coupling variables. A MDA is used to handle the couplings between the different subsystems. The main characteristic of this method is to use meta-models to simplify the optimization problem. In this method, the optimization variables set is split up into three subsets which are optimized separately at different levels of the optimization process. Firstly, the local design variables are handled using meta-models and minimizing disciplinary objective functions, then the coupling variables are determined minimizing a L2 norm on the coupling constraints. Finally the global shared variables are calculated by the global optimizer in order to minimize a global objective function.

Dynamic Leader Follower

The *Dynamic Leader Follower* (DyLeaf) method, proposed by Tava and Suzuki [Tava and Suzuki, 2003] allows to split up the optimization problem without resorting to a system level optimizer in order to coordinate the optimization process. In this method, derived from the leader-follower game, each of the k optimizers takes successively the status of leader, optimizes its disciplinary criterion and changes its local variables and the global variables. The other $k-1$ optimizers (followers) optimize their own objective, modify their design local variables and consider the global variables as constants. During the optimization, the status of the leader changes as many times as necessary. A selection function is added to the optimization problem in order to determine the optimizer which becomes the leader.

Multidisciplinary Design Optimization based on Independent Subspaces

The *Multidisciplinary Design Optimization based on Independent Subspaces* (MDOIS) [Shin, 2001; Shin and Park, 2005] splits up the large system to optimize into different subsystems and uses iteratively a multidisciplinary analysis and independent subsystem optimizations to solve the MDO problem. This method is relatively simple compared to the other optimization methods because it does not involve complex coupling analyses. In this method, each subsystem optimizes its own objective and the global objective function is the summation of the local ones. This method does not require the use of specific sensitivity calculation techniques such as Global Sensitivity Equation and has been verified theoretically – satisfaction of Karush Kuhn Tucker conditions – under some assumptions (*e.g.* separability, no equality constraint at subsystem level and no intervention of coupling variables in the global objective function). However, this method may get into difficulty in the case of non separable (highly coupled) problems.

2.5 Conclusion

In this chapter, we have described the main MDO methods and their mathematical formulations using the unified notations presented in the previous chapter. Firstly, we have detailed the MDO formulations which involve only one optimizer and we have exposed a qualitative comparison of these methods. Then, we have described the formulations which present multiple levels of optimization. For each of the developed formulations, we have tried to show the advantages and the drawbacks of the considered method.

After having described the main MDO methods, the logical next step will consist in comparing them with regard to their application in the Launch Vehicle Design in order to determine which method is the best for our current study case. This comparison will be the purpose of the fourth chapter.

In this chapter, we have only presented MDO formulations and not the optimization algorithms used by these formulations. Before performing the comparison of the MDO methods in the LVD application case, we have to describe the classical optimization algorithms used in MDO and the main methods employed to optimize the trajectory of the launch vehicle. This is the purpose of Chapter 3.

Chapter 3

Optimization algorithms and optimal control techniques

Contents

3.1 Introduction

This chapter is devoted to the presentation of the optimization algorithms used by the MDO processes described in the previous chapter and the main trajectory optimization methods employed in LVD. The second section describes the classical optimization algorithms used in MDO. To this end, we have chosen to make the distinction between the gradient-based and gradient-free algorithms in order to present the optimization algorithms used in MDO. Indeed, since the Launch Vehicle Design is a complex process which often calls disciplinary black box functions, the sensitivity calculations may not be obtained directly. These sensitivities have often to be computed through finite differences and therefore are computationally expensive. For this reason, the choice of using a gradient-based algorithm or not is crucial and may have important consequences on the results and the process efficiency.

Moreover, the trajectory optimization is a key point in Launch Vehicle Design because it is a very difficult problem to solve. Indeed, it involves a non linear differential system to integrate, is responsible for the performance calculation and is subjected to strict equality constraints which highly restrain the feasible search space. The third section concerns the description of the main trajectory optimization methodologies. In this section, we describe the two main family of methods which are the direct and the indirect methods. In this chapter, since we present classical optimization techniques which are not exclusive to MDO problems, we do not make the distinction between the different kinds of variables x, y and z.

3.2 Optimization algorithms used in MDO

3.2.1 Gradient-based algorithms

The gradient-based methods are classical optimization algorithms. Basically, these algorithms consist in differentiating the objective function and the constraints in order to adjust the variables. Because there exist many optimization algorithms which are very popular and well described in literature, we will not present them in this document. Complete descriptions of these algorithms can be found in [Fletcher, 1987; Nocedal and Wright, 2006; Rao, 2009]. In this paragraph, we just briefly present the algorithm used in this work : the Sequential Quadratic Programming (SQP) algorithm. For more details about this algorithm, one can consult [Gill et al., 1994; Bonnans et al., 2006; Nocedal and Wright, 2006]. The SQP algorithm is one of the most powerful non linear programming algorithm and allows to solve constrained differentiable optimization problems such as :

$$\begin{aligned} \textbf{Minimize} \quad & f(z) \\ \textbf{With respect to} \quad & z \\ \textbf{Subject to} \quad & g(z) \leq 0 && (3.1) \\ & h(z) = 0 && (3.2) \end{aligned}$$

The basic idea of this algorithm is to split up the initial problem into a sequence of quadratic programming subproblems (QP). Each subproblem is a simplification of the initial problem using a quadratic model of the Lagrangian function and a linearization of the constraints. Basically, in order to find the direction of descent at the iteration k, the following QP problem is solved:

$$\begin{aligned} \textbf{Minimize} \quad & \mathcal{L}(z_k, \lambda_k, \mu_k) + \nabla\mathcal{L}(z_k, \lambda_k, \mu_k)^T \Delta z + \tfrac{1}{2}\Delta z^T \nabla^2 \mathcal{L}(z_k, \lambda_k, \mu_k)\Delta z \\ \textbf{With respect to} \quad & \Delta z \\ \textbf{Subject to} \quad & g(z_k) + \nabla g(z_k)^T \Delta z \leq 0 && (3.3) \\ & h(z_k) + \nabla h(z_k)^T \Delta z = 0 && (3.4) \end{aligned}$$

with $\mathcal{L}(z_k, \lambda_k, \mu_k)$ the Lagrangian of the initial problem, λ (respectively μ) the Lagrange multipliers vector associated to the equality (respectively inequality) constraints and Δz the direction of descent. The Lagrangian of the initial problem is given by :

$$\mathcal{L}(z, \lambda, \mu) = f(z) + \lambda^T h(z) + \mu^T g(z) \qquad (3.5)$$

Thus, the SQP algorithm involves two nested iterative processes :

- the main process, which updates the values of the real objective function and constraints, their gradients and the Hessian of the objective function (most often using a quasi-Newton updating method like the Broyden-Fletcher-Goldfarb-Shanno method), finds the Lagrange multipliers and checks the convergence criterion,

- the QP process, which solves the quadratic programming problem and is called at each iteration of the main process. The QP problems are solved using classical gradient-based optimization methods for constrained problem.

The basic implementation of the SQP algorithm is the following :

Algorithm 1: SQP algorithm

```
Initialize the SQP algorithm by $z_0$, $\lambda_0$ and $\mu_0$, if possible in the vicinity of the optimum;
repeat
    Approximate the problem with a linearly constrained QP problem at the current $z$;
    Solve the QP problem to obtain the descent direction $\Delta z^*$;
    Compute $\lambda_k^*$ and $\mu_k^*$ (optimized values of $\lambda_k$ and $\mu_k$);
    if $\Delta z^* \neq 0$ then
        Increment the new point $z_{k+1} = z_k + \alpha d$ (with $\alpha$ a step length parameter used to facilitate the convergence of the algorithm and chosen to satisfy the Armijo conditions [Armijo, 1966]), $\lambda_{k+1} = \lambda_k^*$, $\mu_{k+1} = \mu_k^*$
until $\Delta z^* = 0$;
```

In literature, we can find several gradient-based calculation techniques (such as Post-Optimality [Braun et al., 1993], System Sensitivity Analysis [Olds, 1994], Global Sensitivity Equation [Sobieszczanski-Sobieski, 1990], Local Distributed Criteria [Filatyev et al., 2009]) which present some interest in a MDO framework. These algorithms will be described later in this document.

The gradient-based algorithms present some limitations. Indeed, these algorithms require some differentiability properties on the objective function and on the constraints, allow to find only local solutions, have to be initialized in the vicinity of the optimum and need restricted bounds on the optimization variables in order to converge. Nevertheless, when these requirements are satisfied, the gradient-based algorithms are the fastest methods and allow to efficiently find (local) optima.

3.2.2 Gradient-free algorithms

The gradient-free algorithms may present some interest in the MDO field because the engineering (industrial) simulation codes may not have been designed to provide the sensitivity informations in an efficient manner. Moreover, these algorithms allow to work with non differentiable functions (and constraints) whereas the classical gradient-based algorithms require some differentiability and smoothness properties of the objective and the constraints. We can find many gradient-free algorithms in literature. We have chosen to describe in this section three of them : Nelder & Mead, Genetic Algorithm and Efficient Global Optimization.

Nelder and Mead Algorithm

The Nelder & Mead algorithm [Nelder and Mead, 1965] is one of the most used gradient-free optimization algorithms to solve non linear problems. This method is also called the non linear simplex method but is not to be confused with the linear simplex method. This method involves a simplex of n dimensions and uses simple geometric transformations in order to modify the initial simplex and to get closer to an optimum. A $n-$simplex is an $n-$dimensional polytope which is the convex hull of its $n+1$ vertices. Let $z_1, \cdots, z_{n+1}$ be the $n+1$ vertices of the current simplex at the i^{th} iteration (the points are ordered according to the values of the objective function) and z_0 the center of gravity of the n first points, the different geometric operations used to modify the simplex are the following ones :

Algorithm 2: Nelder & Mead algorithm

```
Step 1 – reflection :
    Compute a reflected point with respect to the following relationship :
        $z_r = z_0 + \alpha(z_0 - z_{n+1})$        (3.6)
    if $z_r$ is worst than $z_n$ then
        Go to Step 3
    else
        if $z_r$ is better than the $k^{th}$ point ($k \in \{2, \cdots, n\}$) then
            this point is placed between $z_{k-1}$ and $z_k$;
            $z_{n+1}$ (worst point) is not considered anymore in the current simplex
        else
            Go to Step 2
Step 2 – expansion :
    Compute an expanded point as follows :
        $z_e = z_r + \beta(z_r - z_0)$        (3.7)
    if $z_e$ is better than $z_r$ then
        Replace $z_{n+1}$ by $z_e$
    else
        Replace $z_{n+1}$ by $z_r$
    Go to Step 1
Step 3 – contraction :
    Calculate the contracted point :
        $z_c = z_0 + \delta(z_0 - z_{n+1})$        (3.8)
    if $z_c$ is better than $z_{n+1}$ then
        replace $z_{n+1}$ by $z_c$;
        Go to Step 1
    else
        Go to Step 4
Step 4 – shrinking :
    For all but the best point, replace the point by :
        $z_i = z_1 + \gamma(z_i - z_1), \quad \forall i \in \{2, \cdots, n+1\}$        (3.9)
    Go to Step 1
```

α , β , δ and γ are the reflection, contraction, expansion and reduction parameters. Classical values of these parameters are : $\alpha = \beta = 1$ and $\delta = \gamma = 1/2$.

Some improvements of the classical Nelder & Mead algorithm can be found in literature [Durand and Alliot, 1999; Chelouah and Siarry, 2003; Luersen et al., 2004]. These improvements concern the insertion of bounds on the optimization variables and the globalization of the standard Nelder & Mead algorithm by using multistart strategies or by hybridizing the algorithm with a global search such as Genetic Algorithms. Some convergence properties of the Nelder & Mead algorithm can be found in literature [Lagarias et al., 1998].

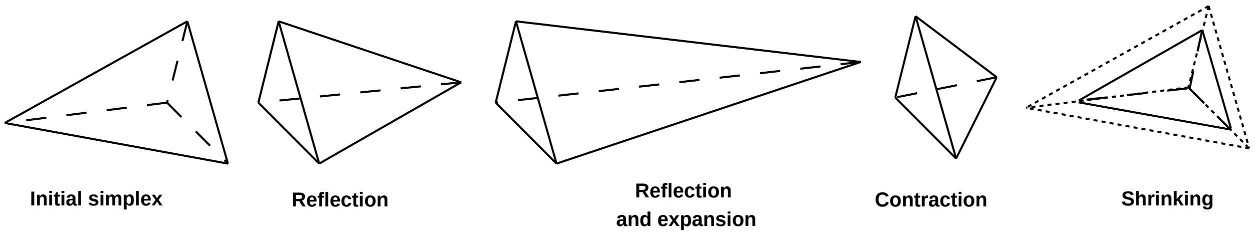


Figure 3.1: Nelder and Mead algorithm

Genetic Algorithms

The Genetic Algorithm (GA) is a very popular metaheuristic algorithm. This algorithm is part of Evolutionary Algorithms. The Evolutionary algorithms are based of the Theory of Evolution of Charles Darwin and aim at optimizing an objective function by defining a population and evolving it according to similar rules as the laws of Natural Evolution principle. The differences between the different Evolutionary Algorithms concern the ways to perform the initialization, the selection and the evolution of the different individuals present in the population.

The Evolutionary Algorithms are based on the following general schedule (Fig. 3.2) :

Algorithm 3: Evolutionary Algorithms

```
Initialization : generation of an initial population (possible random generation),
repeat
    Select of a population of parents;
    Evolve of the population;
    Select of a new generation from the actual parents population and the generated
    children;
until Termination criterion is reached;
```

In this section, due to the fact that we will only use the Genetic Algorithm, we only detail this algorithm. The reader can consult [Bäck, 1996] for a complete description of the Evolutionary Algorithms. The Genetic Algorithm was proposed in the early 1970s by John Holland [Holland, 1975]. Its use was popularized thanks to the works of Goldberg [Goldberg, 1989]. Some convergence analyses about the GA have been performed in literature [Rudolph, 2004; Sharapov and Lapshin, 2006]. A standard representation of an individual is

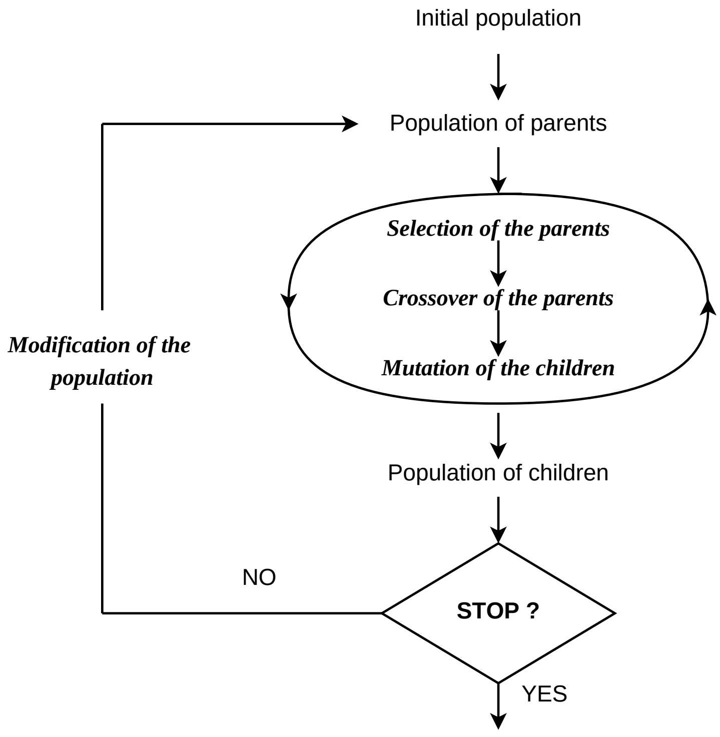


Figure 3.2: Basic scheme of an evolutionary algorithm

performed through a vector of bits (inducing a discretization of the search space). Basically, considering a generation, three operations are performed : the selection of the parents, the crossover of the parents to generate the children and the mutation to randomly create new individuals (Fig. 3.2).

Selection

The selection operator consists in choosing the individuals in the current generation which will constitute the following generation and these which will be involved during the crossover and mutation operations. The different individuals are evaluated and can be ordered (elitism) according to their *fitness* (particular type of objective function which is used to summarize the quality of the individuals). Different selection operators are proposed in literature [Sivaraj and Ravichandran, 2011] (*e.g.* roulette, tournament, best, random *etc.*). There are two phases of selection :

- the first phase is performed on the initial generation in order to select the parents which will be involved in the crossover and mutation operations,
- the second phase is performed once the children have been created and is used to select the individuals which will be present in the next generation.

Crossover

Once the parents have been selected, the crossover operation consists in sharing the code of the parents in order to create children. Thus, the created children are new individuals which share the characteristics of the selected parents. Since the parents have been selected with respect to a performance criterion, the children are expected to present good properties. There exist different ways to perform the crossover (*e.g.* one-point, two-point, uniform, *etc.*).

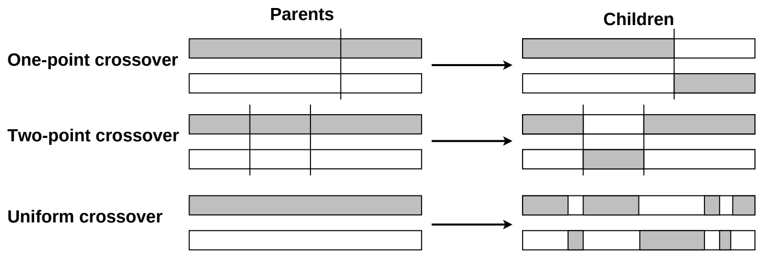


Figure 3.3: Different ways to perform the crossover

Mutation

The mutation operator consists in modifying the children in order to create some diversity in the population. Basically, it consists in changing randomly one or more bit values of the code of selected individuals (Fig. 3.4). The mutation operator allows to create new individuals and to optimize the criterion by performing local variations on the current population. Different ways can be found in literature in order to perform the mutation (*e.g.* flip bit, boundary, non-uniform, uniform, gaussian, *etc.*). The probability of the mutation can be static or can be adapted all along the optimization process.

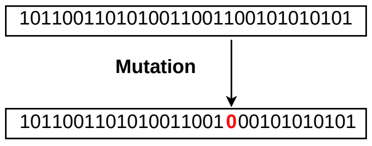


Figure 3.4: Example of mutation

Settings of the control parameters

In order to efficiently converge, the GA has to be tuned. Indeed, the user has to choose the appropriate size of population and the selection, crossover and mutation control parameters corresponding to the problem to solve. These tunings can have a great impact on the behavior of the algorithm and the found results. For example, a too low probability of mutation may not allow the algorithm to converge to the global optimum whereas a too high probability may induce some instabilities and may lead to a total random search. In the same manner, a too low number of individuals may not allow to reach the global optimum (local minimum phenomena) whereas a too high number of individuals will increase the diversity of the population, but will require an important computation time. Some authors have proposed some guidelines to help the choice of the control parameters or some adaptive techniques to limit the intervention of the user [Srinivas and Patnaik, 1994; Eiben et al., 1999] but the choice of the GA control parameters is a relatively problem dependent issue and the user has often to find the parameters which are the most appropriate to his optimization problem.

Efficient Global Optimization

The Efficient Global Optimization (EGO) algorithm [Jones et al., 1998; Sasena, 2002] uses response surfaces based on kriging techniques in order to propose new points to improve the objective function. In this way, this global optimization algorithm aims to reduce the number of calls of the objective function and therefore aims to save calculation time. For more details about the optimization methods using response surfaces, one can consult [Jones, 2001].

From an initial database, the EGO algorithm generates a response surface using a kriging method, and optimizes a dedicated function called the *expected improvement*, which is a compromise between the predicted value of the objective function and the estimation of the lack of information in some areas of the search space, in order to propose new points of interest. Let us describe this algorithm more in detail.

Elaboration of the meta model

EGO is based on the kriging technique to perform the optimization. The kriging technique [Matheron, 1963; Kleijnen, 2009] is used to approximate a function f by a gaussian process:

$$F(z) = m^T(z)b + G(z) \tag{3.10}$$

with :

- $m(z)$: a vector of *a priori* regression function (constant, polynomial,*etc.*),
- b : a vector of regression parameters to estimate from the available data,
- $G(z)$: a centered gaussian process with a covariance $k(.,.)$ which depends on a parameters vector to estimate,

$$k(G(z_i), G(z_j)) = \sigma_G^2 R(z_i, z_j) \tag{3.11}$$

with :

- σ_G^2 : the variance of the process,
- $R(.,.)$: a parametrized correlation function.

The kriging technique consists in finding the Best Linear Unbiased Predictor of $F(.)$ [Kleijnen, 2009]. Basically, we suppose that $R(z_i, z_j)$ only depends on the relative difference between z_i et z_j (noted h). Many choices are possible to model the correlation function [Santner et al., 2003]. One of the most classical models of the correlation function is the power exponential function :

$$R(h) = exp\left(-\sum_{k=1}^{N}\left|\frac{h_k}{\theta_k}\right|^{p_k}\right) \tag{3.12}$$

with N the dimension of the vector z, h_k the k^{th} component of h, $0 < p_k \leq 2$ and θ_k which can be estimated by maximum likelihood. p_k quantifies the regularity of the function f with respect to the kth component of z and θ_k quantifies the sensitivity of the function f with respect to this component. From the n points present in the database and the corresponding true values of the function to estimate : $\mathcal{Z}_n = [z_1, \cdots, z_n]$, $f_n = [f(z_1), \ldots, f(z_n)]$, the predictor of the mean of the gaussian process at z is given by :

$$\hat{F}(z) = m(z)^T \hat{b} + r(z)^T . R^{-1}(f_n - M\hat{b}) \tag{3.13}$$

with :

- $\hat{b} = (M^T R^{-1} M)^{-1} M^T R^{-1} f_n$, the maximum likelihood estimation of the regression parameters vector b,
- $M = [m(z_1), \cdots, m(z_n)]^T$, the $n \times N$ matrix made up of the regression function vectors,
- $r(z) = [R(z, z_1), \cdots, R(z, z_n)]^T$, the vector of correlation coefficients between z and the current database points,
- $R_{i,j} = R(z_i, z_j)$, the $n \times n$ matrix composed of the correlation coefficients of the current database points.

One of the most important advantages of the kriging model is to calculate the variance of the prediction error which is given by [Schonlau, 1997] :

$$\hat{\sigma}^2(z) = \sigma_G^2(1 - r(z)^T R^{-1} r(z)) \tag{3.14}$$

This characteristic is very important for the global search algorithm and is directly used in the EGO algorithm.

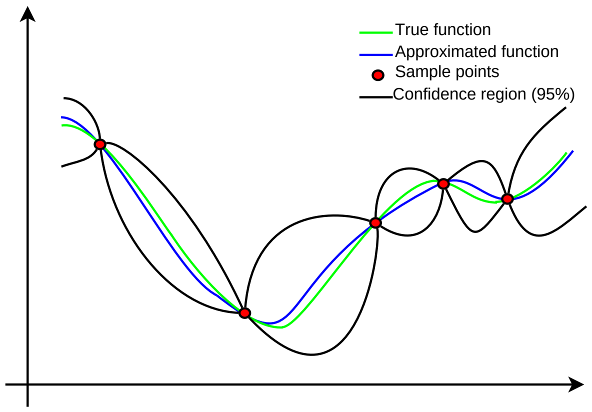


Figure 3.5: Example of kriging model [Marzat et al., 2011]

Description of the optimization algorithm

The principle of EGO consists in using the kriging prediction of the $(n+1)^{th}$ point in order to get closer to the global optimum. This point is chosen by maximizing a criterion quantifying the interest of evaluating f in z, knowing $\mathcal{Z}_n$, f_n, $\hat{F}(z)$ et $\hat{\sigma}^2(z)$:

$$z_{n+1} = \arg\max_z J\left(z, \mathcal{Z}_n, f_n, \hat{F}(z), \hat{\sigma}^2(z)\right) \tag{3.15}$$

A judicious choice for J is the *Expected Improvement* [Jones, 2001], which is defined as follows :

$$EI(z) = \hat{\sigma}(z)[u\Phi(u) + \phi(u)] \tag{3.16}$$

with :

$$u = \frac{f^n_{min} - \hat{F}(z)}{\hat{\sigma}(z)} \tag{3.17}$$

$$f^n_{min} = \min_{i=1,\cdots,n} \{f(z_i)\} \tag{3.18}$$

- ϕ : the probability density function of the normalized gaussian law $\mathcal{N}(0,1)$,
- Φ : a cumulative distribution function.

The optimization of the *Expected Improvement* allows to make a compromise between the local search (numerator of u) and the exploration of unchartered areas (denominator of u), that allows to perform a global optimization.
Finally, the EGO algorithm is organized as follows:

Algorithm 4: Efficient Global Optimization algorithm

```
Perform an initial (experimental) sampling of the search space (creation of initial
database), repeat
    Given the current database, update the kriging model;
    Optimize the Expected Improvement function with using the kriging model, in
    order to find the new point of interest;
    Calculate the true value of the function at this point of interest and update the
    database.
until Termination criterion is reached;
```

Remark 3.2.1. *Some hybrid methods, combining the heuristics and the gradient-based methods have been proposed in literature. These methods aim to benefit from the efficiency of the gradient-based methods and the heuristic algorithm ability to explore large space and perform global optimization. Most often, the employed strategy consists in firstly using a heuristic in order to find an initialization in the convergence domain of a local method which is then applied in order to converge quickly toward the optimum (see for example [Akhtar and Linshu, 2005a, 2006; Briggs et al., 2007a], etc.). Amongst the hybrid methods, we can cite the Genetic Algorithm Guided Gradient Search (GAGGS) method [Geethaikrishnan et al., 2009] which has been applied in order to optimize the trajectory of Launch Vehicles [Geethaikrishnan et al., 2008].*

3.3 Optimal control

This section aims at giving an overview of the main optimal control techniques. A detailed description of the numerical methods used in the trajectory optimization can be found in [Betts, 1998; Rao, 2009]. Classically, the trajectory optimization problem consists in finding the control law u such that :

- the objective function is optimal,
- the constraints are satisfied,
- the dynamics of the system (state vector) is respected.

The dynamics of the system is commonly defined by an ordinary differential equation :

$$\dot{x} = s(x(t), u(t), p) \tag{3.19}$$

where x is the state vector, u the control law, s the state dynamics function and p the design parameters which may intervene during the trajectory integration. Initial conditions at the initial time of the trajectory t_0 are added to the problem:

$$h_0(x(t_0), u(t_0), p, t_0) = 0 \tag{3.20}$$

In most of the cases, the initial state $x(t_0)$ is fixed (coordinates of the launching pad when the launch vehicle lifts off from the ground or the values of the state vector at the launch vehicle separation in the case of airborne launching) :

$$x(t_0) = x_0 \tag{3.21}$$

We define also terminal conditions at the final time of the trajectory t_f :

$$h_f(x(t_f), u(t_f), p, t_f) = 0 \tag{3.22}$$

The final time t_f may be fixed or not. In order to avoid cumbersome notations, we suppose in this part that we have only equality constraints and no path constraints. The objective function is the following :

$$F = f(x(t_f), t_f) \tag{3.23}$$

We define the augmented Lagrangian :

$$L = [f + \mu^T h_f]_{t_f} + \int_{t_0}^{t_f} \lambda^T(t)[s(x(t), u(t), p) - \dot{x}]dt \tag{3.24}$$

where λ is the adjoint vector and μ the Lagrange multiplier vector associated to h. Then, we define the Hamiltonian :

$$H = \lambda^T(t)s(x(t), u(t), p) \tag{3.25}$$

In application of the Pontryagin Maximum Principle [Pontryagin et al., 1962], the necessary conditions (Euler-Lagrange equations) to have an optimum (obtained by setting the first variation of the Lagrangian to zero) are :

$$\dot{\lambda} = -H_x^T \tag{3.26}$$
$$0 = H_u^T \tag{3.27}$$
$$\lambda(t_f) = \tilde{F}^T_{x_{t=t_f}} \tag{3.28}$$
$$0 = (\tilde{F}_t + H)_{t=t_f} \tag{3.29}$$

where $\tilde{F}$ stands for $f + \mu^T h_f$ and H_x (respectively H_u) is the partial derivative vector of H with respect to x (respectively u).
Equation (3.26) is called the adjoint equation, Eq. (3.27) is the control equation, and Eq. (3.28-3.29) are the transversality equations. We can distinguish two main categories of optimization techniques to solve the optimization trajectory problem. These are the direct and the indirect methods.

3.3.1 Direct methods

The direct methods are the most used in the launch vehicle trajectory optimization because they are easy to implement and can be relatively well initialized comparatively to the indirect methods. These methods are used in many trajectory optimization programs (*e.g.* Program to Optimize Simulated Trajectories - POST - [Brauer et al., 1977]). In this kind of methods, the control law is defined by a finite set of parameters u_{p_i} which are handled by the optimizer in order to meet the optimality conditions. Accordingly, the control law is function of the time and the control parameters $u(u_{p_i}, t)$. The most simple way to define u is the piece-wise linear interpolation:

$$u(u_{p_i}, t) = \begin{cases} u_{p_{11}} + u_{p_{12}} t & \text{if} \quad t \in [t_0, t_1[\\ u_{p_{21}} + u_{p_{22}} t & \text{if} \quad t \in [t_1, t_2[\\ & \vdots \\ u_{p_{i1}} + u_{p_{i2}} t & \text{if} \quad t \in [t_{i-1}, t_i[\\ & \vdots \\ u_{p_{n1}} + u_{p_{n2}} t & \text{if} \quad t \in [t_{n-1}, t_n] \end{cases} \tag{3.30}$$

We can also use more elevated order polynomials or B-Splines [De Boor, 1978] if regularity properties concerning the control law are required. The optimization problem is resumed as follows :

$$\begin{aligned} \textbf{Minimize} \quad & f(x(t_f), t_f) \\ \textbf{With respect to} \quad & u_{p_i}, i = 1, \cdots, n \\ \textbf{Subject to} \quad & \dot{x}(t) = s(x(t), u(u_{p_i}, t), p) && (3.31) \\ & h_f(x(t_f), u(u_{p_i}, t_f), p, t_f) = 0 && (3.32) \end{aligned}$$

This problem is a typical Non Linear Programming problem (NLP) and can be solved with a classical gradient-based method such as Sequential Quadratic Programming. Additional constraints (inequality or equality) can be added to the initial problem. One can notice that, for this method, the calculation of the adjoint function is not necessary. Indeed, only the expression of the state dynamics, the control law and the initial conditions are sufficient to simulate the trajectory, compute the objective function, the final conditions and their derivatives and so to perform the optimization. The direct methods are generally effective to solve launch vehicle trajectory optimization problems. Indeed, the main advantages of these methods are the avoidance of the adjoint vector dynamics calculation, the easy implementation and the initialization which is easy to find. Nevertheless, these methods suffer from several drawbacks such as a relatively low convergence speed and an important calculation cost, principally due to the sensitivity calculations when using gradient-based algorithms. Indeed, in most of the cases, the sensitivity calculations are performed by finite difference approximations (Eq. 3.33 or 3.34), which imply as many trajectory simulations as the number of parameters defining the control law. This information is required at each iteration of the NLP.

$$X_{u_{pi}} = \frac{X(u_{pi}+\delta_u) - X(u_{pi})}{\delta_u} \quad (3.33)$$

$$X_{u_{pi}} = \frac{X(u_{pi}+\delta_u) - X(u_{pi}-\delta_u)}{2\delta_u} \quad (3.34)$$

where X stands for f or h_f. Moreover, the control law found by these methods is defined by a finite set of parameters and consequently is not the real optimal one. Indeed, the performance of the found control law is intrinsically dependent of the chosen parametric model. In these methods, a compromise has to be found between the complexity of the control law (number of parameters) and the expected quality of the optimum.

3.3.2 Indirect methods

In the indirect methods, the control law is not defined by a finite set of parameters anymore, but is directly deduced from the control equation (3.27) and the initial values of the state and co-state vectors (adjoint). The aim of these methods is not to decrease the objective function at each iteration but to solve the necessary conditions of optimality (Pontryagin Maximum Principle [Pontryagin et al., 1962; Bryson and Y-C., 1975]) :

$$\dot{x}(t) = H_\lambda(x(t), u(t), p, t), t \in [t_0, t_f] \quad (3.35)$$

$$\dot{\lambda}(t) = -H_x(x(t), u(t), p, t), t \in [t_0, t_f] \quad (3.36)$$

$$u(t) = \underset{v}{\operatorname{argmin}}\, H(x(t), v, p, t), t \in [t_0, t_f] \quad (3.37)$$

$$x(t_0) = x_0 \quad (3.38)$$

$$\lambda(t_f) = \tilde{F}^T_{x_{t=t_f}} \quad (3.39)$$

$$0 = h_f(x(t_f), u(t_f), p, t_f) \quad (3.40)$$

The complete resolution of the problem consists of the resolution of differential system (3.35), (3.36), (3.37) according to the boundary conditions (3.38), (3.39) and (3.40). This problem is called a *Two Point Boundary Value Problem* (TPBVP).

Classically, this problem involves a shooting method which consists in finding the initial values of the adjoint vector $\lambda(t_0)$, the Lagrange multipliers μ and the final time t_f in order to satisfy the equations (3.22), (3.28) and(3.29). The problem amounts to solving the shooting function. This is commonly achieved by a Newton-Raphson method [Bonnans et al., 2006]. The convergence of this process is quadratic. Moreover, in indirect methods, the control law u is defined at each point by the Pontryagin Maximum Principle (3.27). In that way, the found control law is a true optimum. An example of application of this method is the OPTAX tool [Bérend and Talbot, 1996; De Biasi, 2002] developed at the CNES which employs the indirect shooting method to solve the launch vehicle trajectory optimization problem during the exoatmospheric phase.

$$\begin{aligned}
&\textbf{Find} && \lambda(t_0), t_f, \mu && \\
&\textbf{To solve} && \tilde{F}^T_{x_{t=t_f}} - \lambda(t_f) &= 0 \quad &(3.41)\\
& && h_f(x(t_f), u(t_f), p, t_f) &= 0 \quad &(3.42)\\
& && (\tilde{F}_t + H)_{t=t_f} &= 0 \quad &(3.43)
\end{aligned}$$

$$\begin{aligned}
&\textbf{Subject to} && \dot{x}(t) &= H_\lambda(x(t), u(t), p, t) \quad &(3.44)\\
& && \dot{\lambda}(t) &= -H_x(x(t), u(t), p, t) \quad &(3.45)\\
& && H_u &= 0 \quad &(3.46)
\end{aligned}$$

Nevertheless, these methods suffer from several numerical drawbacks. Firstly, the main difficulty of the methods is to find an initial guess of the adjoint vector in order to initialize the algorithm. Indeed, we have to be able to find an initialization that induces a point which is relatively close to the solution at the final time in order to ensure the convergence of the Newton algorithm. Moreover, due to the integration of the trajectory, a small variation on the initial adjoint vector can induce great repercussion at the final time and may possibly make the trajectory diverge. Finally, for the expression of the optimality conditions, indirect methods require the analytical differentiation of f, s and h, which may pose several problems in case of complex physical models (*e.g.* aerodynamics) and limit the flexibility of the method comparatively to the direct methods. For that reasons, these methods are used only for the exoatmospheric phase (for which drag can be neglected).

3.3.3 Single *vs* multiple shooting methods

Heretofore, the described direct and indirect methods have considered the trajectory in one piece. Indeed, from the initial conditions and the values taken by the optimization variables, the trajectory is integrated and the final conditions are returned to the optimizer in order to evolve the optimization variables. This way to proceed may cause several problems with regard to the behavior of the optimization algorithms. The main problem is that small variations on variables occurring at the beginning of the trajectory may be accentuated all along the trajectory and so may cause very harmful effects such as the trajectory divergence. This phenomenon is increased by the highly non linear nature of the dynamics function s. This aspect is crucial for the indirect methods but may have an importance for the behavior of the direct methods as well.
In order to overcome this problem, multiple shooting methods have been developed [Keller, 1968]. The basic idea of these methods is to split up the trajectory into several sections. Thus, the trajectory is integrated on each elementary section, that reduces the harmful variable variation amplification phenomenon. The values of the different initial states at the beginning of each elementary trajectory section have to be included to the initial optimization variables set. In order to ensure the continuity between the different trajectory sections, additional equality constraints are needed and included to the optimization problem. Let us describe

the principle of multiple shooting methods in case of direct methods :
The multiple shooting methods split up the initial time interval $[t_0 \ t_f]$ into N smaller ones such that :

$$t_0 < t_1 < \ldots < t_i < \ldots < t_N = t_f \tag{3.47}$$

The values of the state vector at the beginning of each section $\tilde{x}_2, \cdots, \tilde{x}_N$ (and corresponding constraints $h_1, \cdots, h_{N-1}$) have also to be introduced and are added to the initial optimization variables set. Let $x_i(t_{f_i})$ be the final state vector obtained after the integration of the trajectory during the ith phase. The resulting set of additional constraints which have to be handled is :

$$\begin{aligned}
h_1 &: x_1(t_1) - \tilde{x}_2 = 0 && (3.48)\\
h_2 &: x_2(t_2) - \tilde{x}_3 = 0 && (3.49)\\
&\vdots \\
h_i &: x_i(t_i) - \tilde{x}_{i+1} = 0 && (3.50)\\
&\vdots \\
h_{N-1} &: x_{N-1}(t_{N-1}) - \tilde{x}_N = 0 && (3.51)
\end{aligned}$$

$x_i(t)$ is the state vector $x(t)$ for $t \in [t_{i-1}, t_i]$.
The problem formulation with using the multiple shooting direct methods is the following:

Minimize $$f(x(t_f), t_f)$$

With respect to $$\begin{aligned} &u_{p_i}, && i = 1, \cdots, N \\ &\tilde{x}_j, && j = 1, \cdots, N-1 \end{aligned}$$

Subject to
$$\begin{aligned}
\dot{x}_i(t) = s(x_i(t), u(u_{p_i}, t), p), \quad & t \in [t_{i-1}, t_i], \qquad i = 1, \cdots, N && (3.52)\\
h_j : x_j(t_j) - \tilde{x}_{j+1} = 0, \quad & \qquad j = 1, \cdots, N-1 && (3.53)\\
h_f(x(t_f), u(u_{p_N}, t_f), p, t_f) = 0 & && (3.54)
\end{aligned}$$

u_{p_i} denotes the set of control variables involved in the i^{th} trajectory part.
Indirect methods can also come in multiple shooting formulations [Fraser-Andrews, 1999] using the same methodology. We have to note that, in this case, the additional variables include the values of the adjoint vector at the beginning of the different parts $\lambda_j, j = 1, \cdots, N-1$.

Multiple shooting methods lead to an increase of $N.n_x$ of the number of optimization variables and constraints (with n_x is the dimension of the state vector), but allow to improve the robustness of the methods because the propagation of the non linearities is limited to each trajectory part. State discontinuities are allowed during the optimization process.

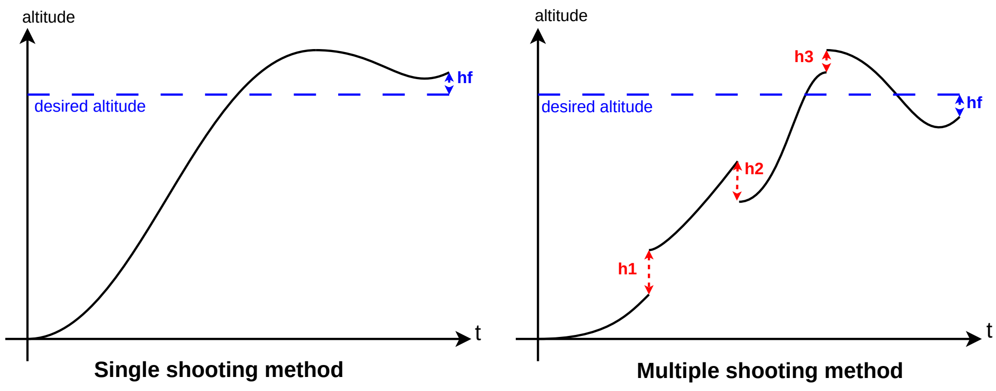


Figure 3.6: Single shooting *vs* multiple shooting methods for a four-section trajectory

3.3.4 Collocation methods

The collocation methods [Russell and Shampine, 1972] may also be applied to the direct and indirect methods and consist in discretizing the dynamics in time and replacing the dynamic equations (state and control dynamics for the direct methods, state, adjoint and control dynamics for the indirect methods) by piece-wise continuous polynomials. Defect constraints are added to the optimization problem in order to ensure the consistency between the real dynamics and the approximating polynomials at the discretization points. Thus, the optimization problem results in a very large sparse NLP problem. Collocation methods are used for example in [Hargraves and Paris, 1987; Gath and Well, 2001] to efficiently solve the trajectory optimization problem.
Let us denote by $\mathcal{X}_k(t_k, \tilde{x}_k)$ and $\mathcal{U}(t_k, \tilde{u}_k)$ the piece-wise polynomial approximations of the state vector x in the k^{th} interval of discretization. The problem formulation in case of direct collocation is the following :

Minimize $\qquad f(\mathcal{X}_N(t_N, \tilde{x}_N), t_N)$

With respect to $\qquad \tilde{u}_i, \quad i = 1, \cdots, N$
$\qquad \tilde{x}_j, \quad j = 1, \cdots, N$

Subject to

$$\dot{\mathcal{X}}_j(t_j, \tilde{x}_j) = s(\mathcal{X}_j(t_j, \tilde{x}_j), \mathcal{U}_j(t_j, \tilde{u}_j), p), \quad j = 1, \cdots, N-1 \tag{3.55}$$
$$h_j : \mathcal{X}_{j+1}(t_j, \tilde{x}_{j+1}) - \mathcal{X}_j(t_j, \tilde{x}_j) = 0, \quad j = 1, \cdots, N-1 \tag{3.56}$$
$$h_f(\mathcal{X}_N(t_N, \tilde{x}_N), \mathcal{U}_N(t_N, \tilde{u}_N, p)) = 0 \tag{3.57}$$

Additional discretization points may be inserted into the N initial intervals in order to improve the accuracy of the dynamics. The resulting NLP problem induced by the collocation method is very large, but the matrices of the NLP problem (Jacobian, Hessian) are sparse. In that way, exploiting the sparsity of these matrices allows to reduce the computation time. Typical approximating function used in collocation methods are the Lagrange Interpolating Polynomials (degree D). For the k^{th} interval, considering D collocation points between t_{k-1} and t_k, the expression of the Lagrange polynomial [Chachuat, 2006] is given by (*e.g.* for the control law) :

$$u_k(t) = \mathcal{U}_k(t, \tilde{u}_k) \triangleq \sum_{i=0}^{D} \tilde{u}_{i,k} \phi_i^D \left(\frac{t - t_{k-1}}{t_k - t_{k-1}} \right) \quad t_{k-1} \leq t \leq t_k \tag{3.58}$$

ϕ_i^D is defined by :

$$\phi_i^D(\tau) \triangleq \begin{cases} 1, & \text{if } D = 0 \\ \prod_{j=0, j\neq i}^{D} \frac{\tau - \tau_j}{\tau_i - \tau_j}, & \text{if } D \geq 1 \end{cases} \tag{3.59}$$

with τ_j the collocation points ($0 \leq \tau_0 \leq \tau_1 \leq \cdots \leq \tau_M \leq 1$).

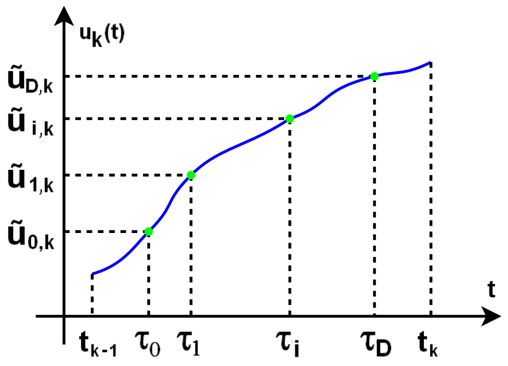


Figure 3.7: Interpolation using Lagrange polynomials

3.3.5 Dynamic Programming

The dynamic programming can also be employed to solve the trajectory optimization problem. This method consists in subdivising the global problem into different subproblems and in combining the different solutions in a recursive manner in order to reach the overall solution. This technique is a generalization of the classical Hamilton-Jacobi Bellman theory [Bellman, 1957]. The main drawback of this method is the "curse of dimensionality" [Bryson and Y-C., 1975]. Indeed, the dimension of the problem quickly soars when the dimension of the state vector or the number of discretization parts grows. Dynamic programming is very difficult to use in realistic cases of launch vehicle trajectory optimization (without MDO). Nevertheless, we can find several papers in literature [Laurent-Varin et al., 2009] using this method to solve the launch vehicle optimal control problem. Due to the fact that this method is very complex to implement in a MDO process and has not been employed in literature concerning MDO problems, we will not describe in details the Dynamic Programming in this thesis. For more details, the reader can refer to [Bellman, 1957; Bertsekas, 2005; Bryson and Y-C., 1975].

3.4 Conclusion

In this chapter, we have briefly described the main optimization algorithms used in the MDO processes and the classical optimization techniques employed to optimize the trajectory of

the launch vehicles. From the MDO methods described in the previous chapter and the optimization algorithms presented in the current chapter, we are now able to compare the different MDO processes which have been applied in literature to solve the Launch Vehicle Design problem. This comparison is the purpose of the next chapter.
Moreover, some of the optimization techniques presented in this chapter will be used in the parts II and III when our proposed MDO methods to solve the LVD problem will be described.

Chapter 4

Analysis of the MDO method application in the Launch Vehicle Design

Contents

4.1 Introduction

In literature, we can find a great number of papers dealing with the application of the MDO methods to the Launch Vehicle Design. In order to propose some ways of improvement with respect to the classical MDO methods used in literature, a complete analysis of the MDO methods presented in the second chapter with regard to the Launch Vehicle Design application is necessary. To this end, this chapter is organized as follows. In the second section of the chapter, we review the state-of-the-art of the MDO methods concerning the different study cases and the optimization criteria. This analysis will help us to determine which MDO method is the most employed in literature and will have to be considered in the next part of this thesis in order to analyze the efficiency of our proposed MDO methods. The third section is devoted to the analysis of the integration of the trajectory optimization into the MDO processes existing in literature. In the fourth section of this chapter, we compare the main MDO methods in a common LVD based application case using different criteria. This comparison will allow us to achieve a comparative synthesis of the MDO methods, and to give in the last section some concluding remarks and to propose some ways of improvement.

4.2 Different study cases and optimization criteria

In literature, we can find many papers dealing with the application of the MDO methods in LVD. The aim of this section is to summarize and to compare (when it possible) the different methods [Balesdent et al., 2011d].

4.2.1 MDO methods and optimization algorithms

The Multi-Discipline Feasible method appears to be the most used MDO method to optimize the design of launch vehicle. Indeed, more than 80% of the surveyed papers for this study use the MDF method to optimize the launch vehicle configuration. Nevertheless, we can find several articles which apply other methods like AAO, IDF, CO, BLISS or MCO (Table 4.1). The gradient-based algorithms (including Local Distributed Criteria, Post-Optimality Analysis, SQP, GSE, *etc.*) are usually preferred to the heuristic algorithms (most often Genetic Algorithms are used). We can also find several articles which propose hybrid algorithms [Akhtar and Linshu, 2005a, 2006; Briggs et al., 2007a; Geethaikrishnan et al., 2008; Rafique et al., 2009a] and combine Genetic Algorithms and gradient-based methods in order to benefit from the advantages of these two kinds of methods.

	Heuristic methods	**Gradient-based methods**
MDF	[Akhtar and Linshu, 2005a,b, 2006] [Bairstow et al., 2006] [Bayley and Hartfield, 2007; Bayley et al., 2008] [Briggs et al., 2007a,b][Collange et al., 2009, 2010] [Castellini et al., 2008, 2010, 2011][Delattre and Mongrard, 2007] [Duranté et al., 2004, 2005][Gang et al., 2005] [Kamm et al., 2007][Lee et al., 2005b][Lemaire et al., 2010] [Qazi and Linshu, 2006][Rafique et al., 2009a,b, 2010] [Schoonover et al., 2000]	[Akhtar and Linshu, 2005a,b, 2006] [Braun et al., 1993][Braun and Kroo, 1995] [Briggs et al., 2007a][Briggs et al., 2007b] [Filatyev and Golikov, 2008; Filatyev et al., 2009] [Geethaikrishnan et al., 2008][Jodei et al., 2006] [Jodei et al., 2009][Kalden, 2007] [Kuratani et al., 2005][Lee et al., 2005a] [Lee et al., 2005b][Moore et al., 1995] [Olds, 1992, 1994][Perrot, 2003] [Qazi and Linshu, 2006][Qu et al., 2004] [Rafique et al., 2009a][Tava and Suzuki, 2002] [Tsuchiya and Mori, 2002, 2004] [Yokoyama et al., 2005, 2007]
IDF	/	[Braun and Kroo, 1995]
AAO	/	[Braun et al., 1996][Brown and Olds, 2005] [Brown and Olds, 2006][Tava and Suzuki, 2003]
CO	[Gang et al., 2005]	[Braun et al., 1996] [Brown and Olds, 2005, 2006] [Cormier et al., 2000][Gang et al., 2005] [Perrot, 2003]
BLISS	/	[Brown and Olds, 2005, 2006]
MCO	/	[Brown and Olds, 2005, 2006]

Table 4.1: Different MDO methods applied to launch vehicle design

4.2.2 Optimization criteria and different study cases

The different study cases are summarized in Table 4.2. These studies indicate the different trends in LVD. In this table, RLV (respectively ELV) stands for Reusable (respectively Expendable) Launch Vehicle and SSTO (respectively MSTO) stands for Single (respectively Multi) Stage to Orbit.

Criterion	RLV-SSTO	RLV-MSTO	ELV-MSTO
Payload mass	[Kalden, 2007] [Yokoyama et al., 2005, 2007]	/	[Bairstow et al., 2006] [Castellini et al., 2008, 2010, 2011] [Collange et al., 2009] [Luo et al., 2004]
Cost	[Braun et al., 1996] [Kuratani et al., 2005] [Moore et al., 1995] [Villeneuve and Mavris, 2005]	[Kuratani et al., 2005] [Villeneuve and Mavris, 2005]	[Bairstow et al., 2006] [Bayley and Hartfield, 2007] [Bayley et al., 2008] [Castellini et al., 2010] [Collange et al., 2009, 2010] [Delattre and Mongrard, 2007] [Lemaire et al., 2010] [Duranté et al., 2004, 2005]
Dry weight	[Braun et al., 1993] [Braun and Kroo, 1995] [Brown and Olds, 2005, 2006] [Cormier et al., 2000] [Olds, 1992] [Olds and Walberg, 1993] [Olds, 1994]	/	/
GLOW	/	[Perrot, 2003] [Tsuchiya and Mori, 2002, 2004]	[Akhtar and Linshu, 2005a,b, 2006] [Briggs et al., 2007a,b] [Castellini et al., 2011] [Jodei et al., 2006, 2009] [Liu et al., 1997] [Luo et al., 2004] [Qazi and Linshu, 2006]

Table 4.2: Different study cases of the MDO methods

4.3 Trajectory handling

In the vast majority of the registered papers, the trajectory optimization is performed with a direct method. The trajectory optimization is often considered as a black box and a parametrical control law is found in order to reach the target orbit. With the notations of Chapter 1, the parameters (of the command law and the state when collocation method is used [Tava and Suzuki, 2002]) are considered as state variables x and are optimized in a disciplinary analysis (state equation solving : $R(x, y, z) = 0$). In most of the cases, the whole trajectory is optimized in one run.

Nevertheless, we can find several papers in which the trajectory is treated in different ways. In a few cases (more often MDF methods : [Duranté et al., 2004; Tsuchiya and Mori, 2004; Delattre and Mongrard, 2007; Castellini et al., 2008]), the control law variables and the design variables z are handled at the same level and the residuals $R(x, y, z)$ are added to the set of equality constraints (1.2) to satisfy at the system level.

The indirect methods are used in [Perrot, 2003] and in the Local Distributed Criteria method [Filatyev and Golikov, 2008; Filatyev et al., 2009] to solve the trajectory optimization problem. In the indirect methods, an adjoint vector is used to indirectly optimize the control law. In this case, the residuals R are composed of the optimality and transversality conditions. The LDC method consists of an iterative method in which the design variable optimization is guided by sensitivity calculations realized by an indirect approach (Post-Optimality Analysis).

The dynamic programming [Liu et al., 1997; Laurent-Varin et al., 2009] (Hamilton Jacobi Bellman approach) is more sporadically used in launch vehicle design because the implementation of this method is not easily applicable to a complete trajectory optimization problem.

4.4 Comparison of the main MDO methods in a common application case

4.4.1 Presentation of the optimization problem

The problem considered in this section is used in many papers to apply the proposed MDO methods. This study case is the optimization of a reusable Single-Stage-To-Orbit (RLV SSTO) able to deliver 11 t of payload to the International Space Station (supposed to have a circular orbit at an altitude of 407 km and an inclination of 51.6°). The base vehicle configuration is optimized using three disciplines : Performance (trajectory optimization), Propulsion and Weights & Sizing. The objective of the design process is to minimize the vehicle dry weight.

4.4.2 Comparison of AAO, MDF and CO

In [Braun et al., 1996], CO, AAO and MDF are compared in the case of the SSTO launch vehicle optimization. Moreover, these methods can be compared with System Sensitivity Analysis used in [Olds, 1992, 1994] to solve an analog optimization problem.

In AAO [Braun et al., 1996], the optimizer has to handle 40 variables. Hence, for 70 iterations, 2870 analysis calls are required (more than 10000 calls are required by using MDF). Both AAO and MDF reach the same solution (within 0.05% in the dry weight).
In comparison with MDF, AAO allows to considerably reduce the computation time. This is due to the fact that a consistent vehicle model is required only at the convergence. Since the disciplinary convergence is not required at each design point, the computation time is considerably reduced. Moreover, AAO can obtain accurate derivatives without requiring strict convergence of the disciplinary models, that improves the process flexibility.

In the CO formulation, there are 23 design variables at the system level and 72 at the subsystem level. At the system level, the Jacobian is obtained from the use of Post-Optimality information [Braun et al., 1993; Braun, 1996]. A SQP algorithm NPSOL [Gill et al., 1986] is used to perform the system level optimization. The objective function of the obtained solution has the same order of magnitude as the one obtained by MDF [Moore et al., 1995]. In [Braun and Kroo, 1995], the results obtained by CO are compared with the results obtained by AAO and MDF in [Braun et al., 1995].

From the results of this study, we can highlight two main advantages of the CO method.

Firstly, CO is a modular method. Indeed, we can add a discipline without spending much time to modify the optimization process. Therefore, CO is more flexible than MDF and AAO. The second main advantage of CO is that communication requirements (between the disciplines) are reduced. This is principally due to the fact that in CO, a part of the optimization process is delegated to the different subspaces and the optimizer converses only with the subsystems once their optimizations are performed.

4.4.3 Comparison of FPI, AAO, BLISS2000, CO and MCO

FPI, AAO, BLISS2000, CO and MCO have been compared in [Brown and Olds, 2006] in the same realistic SSTO optimization problem. Conventional disciplines and legacy codes are taken into account in order to be representative of classical RLV-SSTO early design studies.

Using the FPI method, each discipline handles its own variables, is subject to its own constraints and optimizes its part of the entire system. FPI is not a MDO method but is just a way to solve the Multidisciplinary Analysis problem and gives a reference to compare the methods.

In the AAO formulation, the optimizer is subject to two inequality constraints, 7 equality constraints and involves 13 variables. The results of this study show that AAO produces a 3.94% reduction in the dry mass versus the standard FPI approach.

The BLISS method studied in [Brown and Olds, 2006] is BLISS-2000 [Sobieszczanski-Sobieski et al., 2003]. BLISS-2000 differs from BLISS-RSM [Kodiyalam, 2000; Altus, 2002] in the way the response surfaces are used. The results of this study show that BLISS is able to find an optimum which presents an improvement of 3.76% in comparison with the traditional FPI method.

In the CO formulation, the objective of each discipline optimizer is to minimize a quadratic sum of coupling variable differences. The different disciplines handle the local copies of coupling variables and the objective of the global optimizer is to minimize the dry weight, subject to the coupling consistency. The results show that CO is able to perform 4.39% reduction in the objective versus the FPI method, but provides a large convergence error.

In the MCO formulation, the objective at the system level is to optimize the weighted sum of the dry mass and the 3 local objectives of the different disciplines. The system level optimizer handles 9 design parameters. The study shows that in this test case, MCO led to inconsistent results. Indeed, the resulting solution varied greatly with the value of the penalty parameter at the system level. Due to this phenomenon, it is impossible to conclude from this study about the efficiency of the MCO method.

Unfortunately, this study does not allow to conclude about which of the AAO, CO, BLISS2000 and MCO methods is the best in terms of the quality of the optimum, because the results found by these different methods are very close. Nevertheless, conclusions of this paper show that the use of MDO methods can improve the design process (with respect to FPI). For the considered test case (which is not sufficient to provide general conclusions about the methods), the study shows that AAO is very difficult to implement and is not robust (because it results in a very large and highly constrained problem) but converges quickly with a good convergence error. CO is also difficult to implement, provides a better optimum, but with a large convergence error. BLISS2000 is the MDO method which presents the best performance in the considered study case because it is not very difficult to implement, converges more quickly and has a better convergence error than CO. In order to extend this comparison, we can compare the different methods through different qualitative criteria. This is the purpose of the following section.

4.5 Comparative synthesis of the main MDO methods

4.5.1 Calculation costs and convergence speed

The methods which involve a MultiDisciplinary Analysis are the ones that have the largest computation time. Indeed, the MultiDisciplinary Analysis is a very expensive process, especially when it is coupled with a gradient-based method at the system level.

Decoupling the MDA considerably reduces the computation time but introduces more variables and constraints which have to be handled by the optimizer. If the problem contains few disciplines, and therefore few coupling variables, IDF and AAO are the methods which converge fastest [Martins and Marriage, 2007]. As soon as the optimization problem becomes more complex, AAO is inapplicable because it is very difficult to implement.

Another way to improve the process efficiency is to implement a bi-level method. In that way, the main problem can be subdivided into smaller problems and local optimizers share the coupling variables. The question of coupling constraints can be handled as quadratic sums of differences (CO), approximation by using response surfaces (CSSO) or by an IDF formulation (MCO). CO can considerably improve the optimization process but generates instabilities [DeMiguel and Murray, 2000] and an important convergence error [Brown and Olds, 2006].

The performance of CSSO in complex problems is relatively poor. Indeed, the computation time used by the MDA and the generation of the approximation models can exceed the calculation time saved by using them.

BLISS appears to have good convergence properties [Brown and Olds, 2006] and the calculation costs can be reduced by using approximations models (BLISS-2000, BLISS-RSM). But there again, the question of using approximate models has to be studied individually, in order to determine if it allows to save calculation time or not.

4.5.2 Considerations about optimality conditions

Among all the methods presented in this paper, AAO is the standard MDO method because at the convergence, this method ensures to find a local minimum (satisfaction of the Karush-Kuhn-Tucker conditions). In principle, one should demonstrate that all the other methods are equivalent in the satisfaction of the same KKT conditions as AAO (apart from the errors introduced by using approximate models). This proof is given in [Sobieszczanski-Sobieski et al., 2003] for the BLISS method. In other words, BLISS does not introduce any other errors than the ones of the approximate models and of the extrapolations between the different levels (no theoretical errors). Nevertheless, the BLISS2000 method is subjected to convexity and response surfaces errors. Using the implicit function theorem [Kudryavtsev, 2001], the satisfaction of the KKT conditions of the single-level methods such as MDF and IDF can also be proved. All the methods including an iterative scheme in the optimization process (BLISS, CSSO, *etc.*) and requiring intrinsically sensitivity calculations need convexity properties of the MDO problem to solve. This characteristic is a key difference with respect to other MDO methods (*e.g.* MDF, IDF, CO, *etc.*) which allow to employ heuristics such as GA, and consequently do not require strict convexity of the problem.

4.5.3 Method deftness

Robustness to initialization

In terms of robustness to initialization, defined here as the ability to converge to an optimum from a large initialization domain, AAO seems to be the worst method. Indeed, since all the variables and all the constraints are handled by a single optimizer, AAO needs to be initialized in the vicinity of the optimum to converge. Generally, in case of formulations which use approximate models, the robustness of the methods is correlated with the quality of the models and a particular effort has to be made on the generation of these models.

Applicability to large scale systems

The heavily centralized formulations are difficult to apply with large scale systems. Indeed, because the optimizer handles a lot of coupling variables, these methods are difficult to implement in case of large scale systems and have poor convergence properties. In that way, AAO appears to be inapplicable in large scale problems.

CO seems to perform well for only a low number of coupling variables. BLISS, considering disciplines as black boxes, is applicable to large scale systems without difficulty, even if system analyses are required because they can be performed in parallel. If the disciplinary models are not too complex, CSSO is also applicable in large scale systems, but its efficiency is dependent on the generated approximate models.

Concerning the application of the MDO methods in an industrial context (when engineering teams intervene), the single-level methods are difficult to use because they are not adapted to the organization of the industrial structures. Moreover, these methods give very small autonomy to the engineering teams which have no decision power on the design and only provide disciplinary analyses.

Contrariwise, the multi-level methods (such as CO, CSSO, BLISS, *etc.*) can distribute the disciplinary optimizations to speciality engineering teams and furthermore are more adapted to industrial problems. However, the efficiency of CSSO is to be put in perspective because, in this formulation, the different subsystems require the approximations of the other subsystem coupling influences.

Adaptability and implementation difficulty

Since a lot of complex disciplines are taken into account in the launch vehicle optimization process, the time needed to integrate modifications is an important criterion to compare the different MDO methods. The bi-level methods (BLISS, CO, CSSO, MCO) seem to be more adaptable to a variation of the number of disciplines. For example, comparative studies in LVD [Braun et al., 1996] show that the modification time of CO is 75% less than the standard approach one and 66% less than the AAO one. In BLISS, since the disciplines are considered as black boxes, the global optimizer does not intervene in the black boxes and adding a discipline does not change the process architecture.

The implementation difficulty of the CSSO method is heavily dependent on the disciplinary models. There again, since approximate models have to be generated in CSSO, a very complex discipline can be approximated by non smooth models which provide instabilities and consume a lot of computation time.

Another important criterion which has to be taken into account is the different method adaptability when the disciplinary models or the application case change (*e.g.* moving from a Single-Stage-To-Orbit launch vehicle to a Two-Stage-To-Orbit, minimizing the dry mass instead of maximizing the payload mass, *etc.*). MDF requires a global modification every time the design process changes, because all the disciplines are interlinked with the MDA. IDF and AAO are more flexible because each discipline evolves independently from the others. However, these methods are specific to an application case and when this case changes, all the process can be reorganized.

There again, the bi level methods present some advantages with respect to the single level methods. Indeed, these methods (especially CO and BLISS) are generic at the subsystem level (black boxes for the system level) and the whole design process has not to be revised when a disciplinary model changes. This point is not true for CSSO because for a given subsystem, the influences of the other subsystem behaviors have to be approximated. For CO and MCO, because the different subsystems optimize a quadratic sum or relative errors, a change in the system level objective does not imply any modification at the subsystem level. For BLISS2000, the adaptability of the method to a modification of the global objective function is ensured by the weighting factors, which can evolve all along the optimization process.

4.6 Conclusion and ways of improvement

In the light of this study, we identify several ways of improvement relating to the use of MDO methods in the specific problem of launch vehicle design.

4.6.1 Inclusion of design stability aspects in the optimization process

The design stability aspect, defined here as the insensitivity to uncertainties associated with both design parameters and design models, is a key factor in LVD. This particularity especially takes sense in early phase studies, where simplified models are employed (*e.g.* in structure discipline, finite element codes are often replaced by simplified relationships).

From this point of view, it appears essential that this aspect has to figure as an objective of the optimization process [Du and Chen, 2002]. Indeed, to make sure that the process goes smoothly, it would be interesting to insert a stability indicator, in order to help the designer to evaluate the quality of the solution found by the optimization process. In case of the use of gradient-based algorithms, the design stability with respect to the constraints can be evaluated by the Lagrange multipliers, which can be used to quantify the variation of the objective function with respect to a violation of saturated constraints.

4.6.2 Trajectory optimization in the design process

The launch vehicle design optimization is heavily coupled to the optimal trajectory calculation. Indeed, the entire process is dependent of the results of the trajectory optimization, in which the requirements of the mission intervene.

Almost all examples in literature use a direct shooting method to optimize the launch vehicle trajectory. This choice is principally due to the fact that in the direct method, the initialization of the parameters is easier to find than in the indirect method. Nevertheless, the direct method only provides suboptimal solutions.

The real optimal solution is given by the indirect shooting method [Bryson and Y-C., 1975; Bryson, 1999]. This method is more complex to implement but gives precious informations about the convergence, *i.e.* about the ability of the proposed launch vehicle configuration to reach the specifications of the mission. These informations (adjoint vector and sensitivity of the objective with regard to the parameters) could be exploited in the global optimization process (the optimality conditions would be integrated in the system level equality constraints $h = 0$), in order to change the design parameters z and the state variables x. These considerations could improve the efficiency of the optimization process (like in the LDC method [Filatyev and Golikov, 2008; Filatyev et al., 2009]).

4.6.3 Heuristics and gradient-based algorithms coupling

In order to have a robust (to starting points), efficient and global optimization process, coupling evolutionary and gradient-based algorithms at all the optimization levels may be an interesting way of improvement. Indeed, a lot of synergisms could emanate from the coupling of these two types of methods in the MDO process, like in the GAGGS method [Geethaikrishnan et al., 2008]. With the evolutionary algorithms, the process could explore the entire design space, and therefore avoid the local optima phenomena. By using gradient-method algorithms, the efficiency of the method could be considerably improved, and sensitivity informations could be exploited by the evolutionary algorithms in order to bring the population to optimal solutions.

4.6.4 Another subdivision of the optimization process

All the examples in literature consider a subdivision of the problem into the disciplines such as propulsion, structure, performance calculation, *etc.* The trajectory optimization is considered as a standard discipline. In order to improve the adaptability and the flexibility of the optimization process, a subdivision into a stage-wise decomposition (instead of the decomposition into the different disciplines) could place the trajectory optimization at the root of the optimization process. This kind of decomposition could provide good results in a MDO context. Indeed, the optimization process would be subdivided in as many subsystems as the number of stages. Each subsystem would regroup the different disciplines which intervene in the stage design and these subsystems would be piloted by the trajectory optimization (optimal control, maybe by indirect shooting method). Such a decomposition could be handled by a multi-level process (*e.g.* a collaborative method).

Following the previous study, we have chosen to work on the elaboration of new MDO formulations using a transversal subdivision of the LVD process in order to place the trajectory at the center of the optimization process (Part II). Moreover, we want to develop a dedicated optimization strategy, which aims at coupling heuristics and gradient-based algorithms, in order to improve the robustness of the design process (Part III).

Part II

Multidisciplinary Design Optimization formulations using a stage-wise decomposition

Chapter 5

Presentation of the stage-wise decomposition formulations

Contents

5.1 Introduction

This part is devoted to the description of the stage-wise decomposition formulations. One of the main concluding remarks of the previous part concerns the transversal reorganization of the optimization process. Indeed, it emerged from the analysis of the MDO methods applied to the LVD problem that a decomposition according to the different stages instead of the different disciplines – which is classically performed in literature – could improve the efficiency of the MDO process. To this end, we propose in this part different formulations exploiting the decomposition of the optimization process into the different flight phases. In these formulations, the subsystems are not the disciplines anymore but are composed of the different stage optimizations. Each stage gathers all the disciplines and is optimized separately. The global trajectory is decomposed into elementary flight phases which are optimized during the stage optimizations.

The decomposition into the flight phases has already been applied to optimize the trajectory of launch vehicles [Beltracchi, 1992; Rahn and Schöttle, 1994; Ledsinger and Olds, 2002; Perrot, 2003]. The existing methods in literature consist in optimizing separately the different flight phases in order to split up the original optimization problem into smaller ones. These methods have been applied to optimize either the trajectory only or the trajectory and a small number of design variables (most often the propellant masses). This decomposition has been used to optimize the trajectory of expendable [Beltracchi, 1992] and reusable launch vehicles [Rahn and Schöttle, 1994; Ledsinger and Olds, 2002; Perrot, 2003].

In [Beltracchi, 1992], a decomposition process is proposed to solve the all-up trajectory optimization problem concerning a booster and an upper stage in order to deliver a maximal payload into a target orbit. The optimizations of the booster and the upper stage trajectories are performed separately and are coordinated through the definition of a park orbit between the two subsystems. The optimization strategy is an iterative process between the optimizations of two subsystem trajectories and a system level coordinator, and sensitivity calculations are transmitted from the subsystem to the system levels in order to improve the optimization process. The results of this study show that the solution obtained by the proposed decomposition strategy is valid and can be a good guess for a traditional optimization considering simultaneously all the optimization variables, which presents an important calculation time (and may fail) when initialized with a point which is not in the vicinity of an optimum.

In [Rahn and Schöttle, 1994], the optimization of branching trajectories of a reusable two-stage-to-orbit launch vehicle is solved (Sänger-type mission). The optimization process handles the trajectory variables and the propellant masses of the different stages. The proposed strategy involves a system level optimizer to coordinate the orbiter and booster flight phases through the definition of a staging point. This study allows to show the capability of the flight phase decomposition to optimize the trajectory and a few mass variables. Nevertheless, the problem treated in this paper does not perform a complete multidisciplinary optimization and does not consider all the design variables and disciplines other than the trajectory and the propellant mass definition (no disciplinary couplings).

In [Ledsinger and Olds, 2002], FPI, CO and two formulations of IDF are compared in the optimization of branching trajectories of a fully reusable two-stage launch vehicle. The coordination of the different branches is performed through the handling of a staging point at the system level. This paper shows that the use of MDO methods associated to the flight phase decomposition allows to transform the non hierarchical initial problem to a hierarchical one and find a better optimum than using the classical engineering method (manual iteration which involves a loop between the sequential optimizations of the different branches). Nevertheless, CO appears to be more computationally expensive than the other used methods for this problem (Kistler K-1 problem).

In [Perrot, 2003], a FPI, a hybrid MDF which presents a MDA and a sub-optimization at the subsystem level, and a collaborative approach which once again optimizes independently the different stages at a subsystem level and coordinates them at a system level *via* a staging point, are compared in the trajectory optimization problem by using direct and indirect methods. As in [Ledsinger and Olds, 2002], a FPI method, which considers the optimization problem as a loop between the optimizations of the different flight phases is compared to the MDF method. Unfortunately, the results obtained for the collaborative approach are not shown. Nevertheless, this study shows that using the MDF method presents some benefits with respect to FPI which presents a lack of consistency of the results (no convergence towards the identified optimum whereas the MDO method allows to converge in a few system-level iterations).

All these studies are mainly focused on the trajectory optimization, involving the trajectory variables (with occasionally a few design parameters concerning the masses *e.g.* the propellant mass) and do not consider all the disciplinary design variables and the induced disciplinary couplings. The trajectory optimization of expendable launch vehicles presents the same characteristics as the RLV trajectory. Indeed, the trajectory can also be decomposed into the different flight phases which correspond to the flights of the different stages. In this part, we use the flight phase decomposition in order to perform the complete multidisciplinary design optimization of expendable launch vehicles. We do consider not only the trajectory but all the different disciplines present in the MDO problem (and induced couplings).

This part is organized in three chapters. In the first chapter, we present the different stage-wise decomposition formulations. This chapter is based on the launch vehicle trajectory analysis in order to describe the different proposed MDO formulations. We first detail the initial launch vehicle design problem, then we identify the different couplings between the subsystems and finally we propose four different ways to handle these couplings. The second chapter is devoted to the description of the application case. We describe the different disciplines, variables and constraints. The second part of this chapter concerns the application of the proposed formulations to the detailed application case. The last chapter deals with the theoretical and numerical comparisons of the different formulations.

5.2 Decomposition of the problem and identification of the couplings

In order to formally describe the proposed stage-wise decomposition formulations, we first define the launch vehicle design problem. Then, we analyze the launch vehicle state vector dynamics in order to identify the couplings between the different flight phases. This analysis serves us to propose four formulations using the stage-wise decomposition to solve the launch vehicle design problem.

5.2.1 Definition of the launch vehicle design problem

Using the classical notations defined in the previous part, the launch vehicle design optimization problem can be formulated as follows :

Formulation II.1

$$\begin{aligned}
\textbf{Minimize} \quad & f(z_{d_1}, \cdots, z_{d_i}, \cdots, z_{d_n}, y_0) \\
\textbf{With respect to} \quad & y_0, z = \{z_{d_k}, z_{c_k}\}, \quad k = 1, \cdots, n \\
\textbf{Subject to} \quad & g_k(z_{d_k}, z_{c_k}, y_0, x) \leq 0 && (5.1) \\
& h_k(x_f) = 0 && (5.2) \\
& c(z_{d_k}, z_{c_k}, y_0, x) = 0 && (5.3) \\
& \dot{x} = s(z_{d_k}, z_{c_k}, y_0, x) && (5.4)
\end{aligned}$$

with :

- f : objective function. In LVD, the objective function is often a mass (*e.g.* minimum lift off mass, maximum payload mass) or a cost criterion.
- z_d : design variables. These variables are the disciplinary optimization variables which take part in the definition of the objective function (*e.g.* propellant masses, diameters, pressures, *etc.*).
- z_c : control variables. These variables allow to define the control law of the launch vehicle. These variables define the trajectory of the launch vehicle and intervene in the constraints definition.
- y_0 : initial coupling variables. These variables are used in the standard LVD problem definition in order to link the different disciplines (*e.g.* coupling concerning the maximal load factor between the trajectory and the mass budget disciplines).
- g : inequality constraints. These constraints can be trajectory (*e.g.* maximal angle of attack, mechanical and thermal loads) or design constraints (*e.g.* maximal nozzle exit diameter).
- h : equality constraints. In LVD, the equality constraints often deal with the satisfaction of the mission requirements. These equality constraints are most often very difficult to satisfy and considerably limit the feasible search domain definition.
- c : initial coupling constraints. These coupling constraints are used to ensure the consistency of the couplings defined by the initial coupling variables y_0.
- x : state vector. The state vector is used to simulate the trajectory of the launch vehicle. x_f stands for the final value of x.
- s : dynamics of the state vector. These functions describe an ordinary differential equation system which allows to define the evolution of the state vector x all along the trajectory.
- n : number of stages.

5.2.2 Decomposition of the state vector dynamics

For an expendable launch vehicle, the global trajectory is composed of the different stage flight phases, linked by stage separations inducing non continuous variations of the state vector (mass jettisonings). Each stage presents its own design and control variables, constraints and dynamics functions. The dynamics of the launch vehicle state vector can be decomposed according the different stage flight phases as follows (Fig 5.1) :

$$\begin{cases} x = x_1,\ t \in [0, t_{f_1}], \\ \quad\vdots \\ x = x_i,\ t \in [t_{f_{i-1}}, t_{f_i}], \\ \quad\vdots \\ x = x_n,\ t \in [t_{f_{n-1}}, t_f]. \end{cases}$$

and,

$$\Big\{ \dot{x_1} = s_1(z_{d_1}, z'_{d_2}, \cdots, z'_{d_n}, z_{c_1}, y_0, x_1) \tag{5.5}$$

$$\vdots$$

$$\begin{cases} \dot{x_i} = s_i(z_{d_i}, z'_{d_{i+1}}, \cdots, z'_{d_n}, z_{c_i}, y_0, x_i) & (5.6) \\ u_{i0} = u_{i-1}(t_{fi-1}) & (5.7) \\ x_{i0} = x_{i-1}(t_{fi-1}) & (5.8) \end{cases}$$

$$\vdots$$

$$\begin{cases} \dot{x_n} = s_n(z_{d_n}, z_{c_n}, y_0, x_n) & (5.9) \\ u_{n0} = u_{n-1}(t_{f_{n-1}}) & (5.10) \\ x_{n0} = x_{n-1}(t_{f_{n-1}}) & (5.11) \end{cases}$$

with :

- t_{f_i} : final time of the ith flight phase,
- $u(z_c, t)$: control law,
- x_{i0} (respectively u_{i0}): initial state vector (respectively control law vector) of the i^{th} flight phase,
- $x_i(t_{f_i})$ (respectively $u_i(t_{f_i})$) : final state vector (respectively control law vector) of i^{th} flight phase,
- z'_{d_i} : design variables of the i^{th} stage which play a part in the dynamics of other stage flight phases (*e.g.* all the diameters of the different stages play a part in the dynamics of the first stage through drag calculation). z'_{d_i} are defined by :

$$z'_{d_i} = \{z_{d_{ij}} \in z_{d_i} | \exists k \neq i, \exists (z_{d_k}, z_{c_k}, y_0, x_k), \nabla_{z_{d_{ij}}} s_k(z_{d_k}, z_{c_k}, y_0, x_k) \neq 0\} \tag{5.12}$$

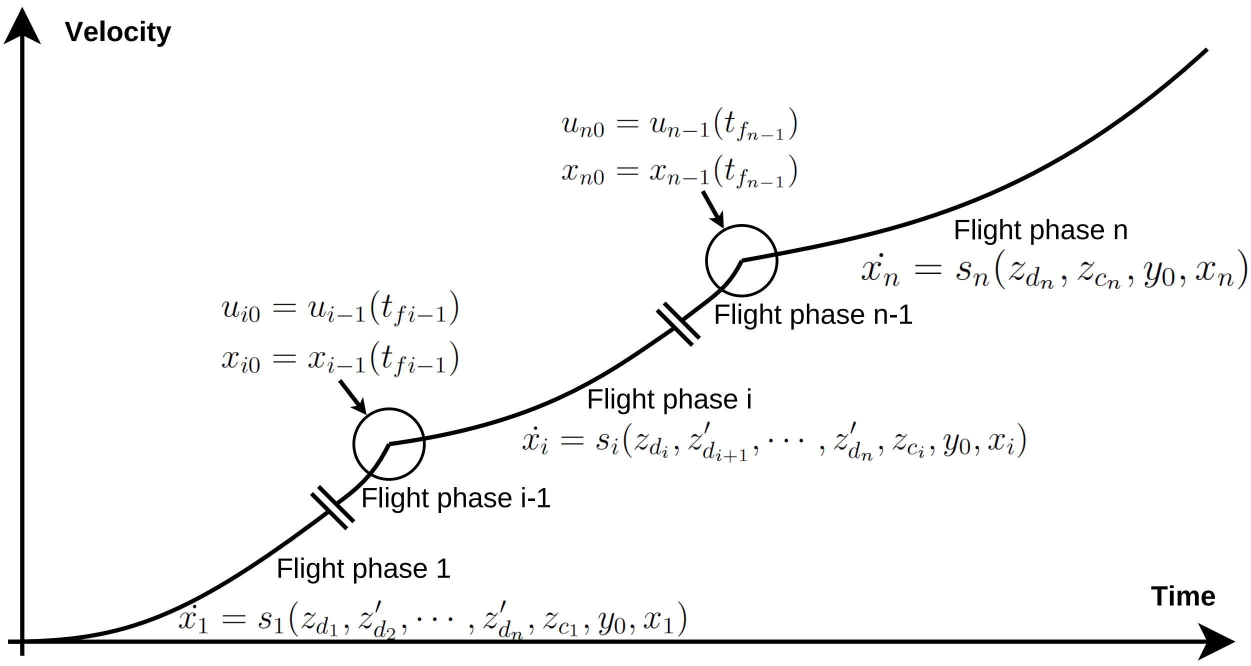

Velocity
$u_{n0} = u_{n-1}(t_{f_{n-1}})$
$x_{n0} = x_{n-1}(t_{f_{n-1}})$
Flight phase n
$\dot{x_n} = s_n(z_{d_n}, z_{c_n}, y_0, x_n)$
Flight phase n-1
$u_{i0} = u_{i-1}(t_{fi-1})$
$x_{i0} = x_{i-1}(t_{fi-1})$
Flight phase i
$\dot{x_i} = s_i(z_{d_i}, z'_{d_{i+1}}, \cdots, z'_{d_n}, z_{c_i}, y_0, x_i)$
Flight phase i-1
Flight phase 1
$\dot{x_1} = s_1(z_{d_1}, z'_{d_2}, \cdots, z'_{d_n}, z_{c1}, y_0, x_1)$
Time

Figure 5.1: Decomposition of the state vector dynamics

In order to be consistent all along the trajectory, the state vector and the control law have to be continuous at the stage separations (except for the masses). For that purpose, the equations (5.7-5.8) and (5.10-5.11) are required.

Remark 5.2.1. *The mass is a particular state variable because for multi stage launch vehicles, its dynamics presents discontinuities at the stage separations. The continuity of the control law is not always required during the trajectory. Indeed, the control law can be discontinuous in the exoatmospheric phases or if the control concerns the thrust. Nevertheless, in this study, we maintain the continuity of the control law during the whole trajectory.*

Remark 5.2.2. *Given a stage, only the design variables of the upper stages intervene in its flight phase.*

Indeed, when the i^{th} stage begins its flight phase, all the k underneath stages ($k < i$) have been jettisoned and thus do not take part in the flight of the i^{th} stage which explains the absence of the $z'_{d_{k,k<i}}$ in the equations of the (5.5,5.6,5.9).

5.2.3 Determination of the coupling variables

The different differential systems present some couplings which have to be handled. These couplings come in addition to the initial couplings y_0 and are due to the subdivision into the different flight phases. All the couplings are represented by the underlined variables in the following equations.

$$\left\{ \dot{x_1} = s_1(z_{d_1}, \underline{z'_{d_2}, \cdots, z'_{d_n}}, z_{c_1}, \underline{y_0}, x_1) \right. \tag{5.13}$$

$$\vdots$$

$$\begin{cases} \dot{x_i} = s_i(z_{d_i}, \underline{z'_{d_{i+1}}, \cdots, z'_{d_n}}, z_{c_i}, \underline{y_0}, x_i) & (5.14) \\ u_{i0} = \underline{u_{i-1}(t_{f_{i-1}})} & (5.15) \\ x_{i0} = \underline{x_{i-1}(t_{f_{i-1}})} & (5.16) \end{cases}$$

$$\vdots$$

$$\begin{cases} \dot{x_n} = s_n(z_{d_n}, z_{c_n}, \underline{y_n}, x_n) & (5.17) \\ u_{n0} = \underline{u_{n-1}(t_{f_{n-1}})} & (5.18) \\ x_{n0} = \underline{x_{n-1}(t_{f_{n-1}})} & (5.19) \end{cases}$$

The couplings symbolize the influence of the variables y_0 and z'_{d_i} on the stage dynamics and the consistency of the differential system integrations (values of the state vector and the control law at the initial and final times of the flight phases).

5.2.4 Proposition of the optimization strategy

Henceforth, we describe the case of the minimization of the Gross-Lift-Off-Weight (GLOW) which is a standard study case in Launch Vehicle Design, as described in Table 4.2 (Chapter 4). With a few modifications, the formulation of the MDO problem can be applied in the case of payload maximization [Balesdent et al., 2010b, 2011e] or cost minimization.

In the case of GLOW minimization, the global mass is defined as the sum of the different stage masses and the mass of the upper composite (composed of the payload, the payload adaptor and the fairing). Thus, the objective function can be decomposed as follows :

$$min(f) \equiv min(f_1) + min(f_2) + \cdots + min(f_i) + \cdots + min(f_n) + f_{upper\ composite} \tag{5.20}$$

where f_i stands for the mass of the i^{th} stage (the upper composite is supposed to be fixed). Using the stage-wise decomposition strategy, we can propose the following optimization scheme :

$$\text{min(f)} \Longrightarrow \left\{ \begin{array}{l} min(f_1(z_{d_1}, y_0)) \left\{ \begin{array}{l} \dot{x_1} = s_1(z_{d_1}, z'_{d_2}, \cdots, z'_{d_n}, z_{c_1}, y_0, x_1) \\ g_1(z_{d_1}, z'_{d_2}, \cdots, z'_{d_n}, z_{c_1}, y_0, x_1) \leq 0 \\ c(z_{d_1}, z'_{d_2}, \cdots, z'_{d_n}, z_{c_1}, y_0, x_1) = 0 \end{array} \right. \\ \qquad\qquad\qquad\quad \vdots \\ min(f_i(z_{d_i}, y_0)) \left\{ \begin{array}{l} \dot{x_i} = s_i(z_{d_i}, z'_{d_{i+1}}, \cdots, z'_{d_n}, z_{c_i}, y_0, x_i) \\ x_{i0} = x_{i-1}(t_{f_{i-1}}) \\ u_{i0} = u_{i-1}(t_{f_{i-1}}) \\ g_i(z_{d_i}, z'_{d_{i+1}}, \cdots, z'_{d_n}, z_{c_i}, y_0, x_i) \leq 0 \\ c(z_{d_i}, z'_{d_i+1}, \cdots, z'_{d_n}, z_{c_i}, y_0, x_i) = 0 \end{array} \right. \\ \qquad\qquad\qquad\quad \vdots \\ min(f_n(z_{d_n}, y_0)) \left\{ \begin{array}{l} \dot{x_n} = s_n(z_{d_n}, z_{c_n}, y_0, x_n) \\ x_{n0} = x_{n-1}(t_{f_{n-1}}) \\ u_{n0} = u_{n-1}(t_{f_{n-1}}) \\ g_n(z_{d_n}, z_{c_n}, y_0, x_n) \leq 0 \\ c(z_{d_n}, z_{c_n}, y_0, x_n) = 0 \\ h_n(x_n(t_{f_n})) = 0 \end{array} \right. \end{array} \right.$$

In order to decouple the different optimizations, we have to introduce additional coupling variables and corresponding constraints, which relate to the junctions of the initial and final conditions of the flight phases :

$$y = \{x_{kf}, u_{kf}, y_0\}, \quad k = 1, \cdots, (n-1) \tag{5.21}$$

where y_0 are the coupling variables of the initial problem and y the coupling variables of the proposed methodology. For simplification reasons, x_{kf} (respectively u_{kf}) stands for $x_k(t_{f_k})$ (respectively $u_k(t_{f_k})$). Therefore, a possible optimization strategy consists in splitting up the optimization process into two levels, with a subsystem level in charge of performing the n optimizations corresponding to the n subsystems and a system level which aims to coordinate the subsystem optimizations, *i.e.* which handles the shared variables, coupling variables and coupling constraints.
We note $\{\bar{z}_{d_k}\} = \complement_{\{z_{d_k}\}}\{z'_{d_k}\}$, the design variables of the subsystem k which do not play a part in the optimization of other flight phases and $z_{sh} = \bigcup_{i=1}^{n} z'_{d_i}$ the shared design variables. Moreover, we denote by $\bar{z}_k$ all the variables (design and control) which play a part only in the k^{th} flight phase. Thus, we have the following notations :

$$z_{d_k} = z'_{d_k} \cup \bar{z}_{d_k} \tag{5.22}$$
$$\bar{z}_k = \bar{z}_{d_k} \cup z_{c_k} \tag{5.23}$$
$$z_{sh} = \bigcup_{i=1}^{n} z'_{d_i} \tag{5.24}$$

5.3 Stage-wise decomposition formulations

5.3.1 First formulation

Principle

The first method transforms the global MDO problem into $n+1$ smaller ones (n stages and 1 coordinator). In this formulation, each subsystem optimization is analog to a single-stage-launch vehicle MDO problem. Indeed, each subsystem handles the design and trajectory variables of the considered stage in order to minimize its gross mass (objective function f_i for the ith subsystem), and to reach the requirements of its mission (targets prescribed by the system level coordinator).

The system level handles the shared design variables, which directly have an influence on the state vector dynamics $s_{1,\cdots,n}$ during the integration of the trajectory, and the coupling variables. The set of coupling variables for this formulation is the following :

$$y = \{\widehat{x_{1f}}, \cdots, \widehat{x_{n-1f}}, \widehat{u_{1f}}, \cdots, \widehat{u_{n-1f}}, y_0\} \tag{5.25}$$

Thereafter, for convenience reasons, we note $\widehat{x_{kf}}$ (respectively $\widehat{u_{kf}}$) the system level coupling variables corresponding to the final state vector (respectively the final value of control law) of the kth stage. We can remark that one component of the state vector (mass) plays a particular part in the optimization. Indeed, given the stage i, the initial value of this component (which will be denoted x_{i0}^m) is the objective function f_i to minimize (mass component). This objective function, which has an influence on the optimizations of the $j < i$ subsystems cannot be know by anticipation by the j^{th} subsystems and has to be estimated at the system level. Thus, in order to ensure the consistency of the optimization process, additional equality constraints (5.32) have to be introduced in the optimization process :

$$c_{02} = \widehat{x_{1f}^m} - f_2^* \tag{5.26}$$

$$\vdots$$

$$c_{0i} = \widehat{x_{i-1f}^m} - f_i^* \tag{5.27}$$

$$\vdots$$

$$c_{0n} = \widehat{x_{n-1f}^m} - f_n^* \tag{5.28}$$

where $\widehat{x_{k-1f}}$ in the equations (5.26–5.28) stands for the estimated objective function of the k^{th} subsystem (analog to the payload mass of the k-1th stage) and f_k^* stands for the real optimized objective function of the k^{th} subsystem. Each subsystem handles its own design and control variables $\bar{z}_{d_k}$ and z_{c_k}, and is subject to inequality and coupling constraints inherent to its own configuration and flight phase. Additional equality coupling constraints which represent the reach of the final state prescribed by the system level have to be introduced at the system level :

$$c_{i1} = \widehat{x_{if}^{-m}} - x_{if}^{-m} \tag{5.29}$$

with x_{if}^{-m} stands for the state vector without the mass component, and the junction of the control law at the stage separations :

$$c_{i2} = \widehat{u_{if}} - u_{if} \tag{5.30}$$

In the equations (5.29-5.30), x_{if} (respectively u_{if}) is the (real) i^{th} final state vector (respectively control law) calculated at the subsystem level thanks to the integration of the ith stage trajectory whereas $\widehat{x_{if}}$ (resp. $\widehat{u_{if}}$) is the final state vector (resp. control law) extracted from y (Eq. 5.21). For the last stage, c_i (Eq. 5.36) are replaced by the equality constraints h_i (Eq. 5.34) symbolizing the mission requirements and there is no coupling constraint about the control law.
Thus, the coupling constraints (Eq. 5.29 and 5.30) ensure the consistency of the optimization process with respect to the integration of the trajectory (continuity of the dynamics Eq. 5.4). The formulation of the optimization process is the following :

Formulation II.2
At the system level :

$$\begin{aligned} \textbf{Minimize} \quad & f = \sum_{i=1}^{n} f_i^*(z_{sh}, y) + M_{upper\ composite} \\ \textbf{With respect to} \quad & z_{sh}, y \\ \textbf{Subject to} \quad & g_0(z_{sh}, y) \leq 0 && (5.31) \\ & c_0(z_{sh}, y) = 0 && (5.32) \end{aligned}$$

At the subsystem level (for the ith stage) :

$$\begin{aligned} \textbf{Given} \quad & z_{sh}, y \\ \textbf{Minimize} \quad & f_i(\bar{z}_{d_i}, z_{sh}, y) \\ \textbf{With respect to} \quad & \bar{z}_{d_i}, z_{c_i} \\ \textbf{Subject to} \quad & g_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) \leq 0 && (5.33) \\ & h_i(x_{i_f}) = 0 && (5.34) \\ & c_i(x_{i_f}, u_{i_f}, y) = 0 && (5.35) \\ & c(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) = 0 && (5.36) \\ & \dot{x}_i = s_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) && (5.37) \end{aligned}$$

where Eq. (5.34) are only considered for $i = n$ and Eq. (5.35) for $i \neq n$.

Remarks concerning the first formulation

The first formulation allows to reduce the MDO problem to the coordination of n smaller ones. Indeed, the ratio between the dimension of each subsystem problem with respect to the global MDO problem dimension is approximatively $1/n$. Moreover, by decomposing the trajectory into n smaller parts corresponding to the flights of the stages, this formulation allows to break up the propagation of the perturbations all along the trajectory and thus to satisfy more easily the equality constraints. For these reasons, we may expect a better behavior of the global optimization process for the formulation 1 with respect to the initial MDF formulation (Form. II.1).

The possible drawback of the first formulation is the coupling constraints (5.32) at the system-level. Indeed, in this formulation, the convergence of the different subsystems and the satisfaction of the system level equality constraints are not correlated. A possible improvement of this formulation consists therefore in reorganizing the subsystem optimizations in order to correlate the system level equality constraint satisfaction with the subsystem convergence *i.e.* to find a formulation in which the subsystem convergence implies the satisfaction of the system level coupling constraints. This is the purpose of the second formulation.

5.3.2 Second formulation

Principle

The second formulation applies the technique of Collaborative Optimization [Braun et al., 1996] to the stage-wise decomposition. Indeed, the different subsystems do not aim at minimizing the stage masses anymore but expect to reach the targets prescribed by the system level coordinator.

In this formulation, the objective function at the system level remains unchanged. Here, the subsystems do not aim to optimize $f_{i,i=1,\cdots,n}$ but have to minimize quadratic sums $J_{i,i=1,\cdots,n}$ of relative differences with respect to the coupling variables specified at the system level. The minimization of these quadratic sums replaces the satisfaction of the equality constraints at the subsystem level (Eq. 5.29, 5.30, 5.35).
On the other hand, the $n-1$ equality constraints (Eq. 5.26–5.28) are replaced by n equality constraints (Eq. 5.41–5.43) which concern the convergence of the different subsystems. The set of coupling variables for this formulation is the following :

$$y = \{\widehat{x_{10}^m}, \widehat{x_{1f}}, \cdots, \widehat{x_{n-1f}}, \widehat{u_{1f}}, \cdots, \widehat{u_{n-1f}}, y_0\} \tag{5.38}$$

An additional variable $\widehat{x_{10}^m}$, which stands for the launch vehicle mass at the beginning of the first flight phase has to be added to the optimization problem. This variable is used in the first stage objective function formulation.

Remark 5.3.1. *In the original Collaborative Optimization formulation [Braun, 1996], the system level handles all the optimization variables and the different subsystems only handle local copies of the corresponding design variables in order to minimize their objective function. The proposed second formulation differs from CO in the repartition of the design variables. Indeed, in the second formulation, the different subsystems do not handle local copies of the design variables but deal with the variables corresponding to the configuration and the trajectory of the stages. In the same way, the system level optimizer does not handle all the design variables but only the shared and coupling variables. Nevertheless, the second formulation uses the CO formulation to define the subsystem level objective function and the system level coupling constraints.*

The formulation of the optimization process is the following :

Formulation II.3
At the system level :

$$\begin{aligned}
&\textbf{Minimize} && f = \sum_{i=1}^{n} f_i^*(z_{sh}, y) + M_{upper\ composite} \\
&\textbf{With respect to} && z_{sh}, y \\
&\textbf{Subject to} && g_0(z_{sh}, y) \leq 0 && (5.39) \\
& && c_0(z_{sh}, y) = 0 && (5.40)
\end{aligned}$$

with :

$$\begin{aligned}
c_{01} &= J_1^* && (5.41) \\
&\vdots \\
c_{0i} &= J_i^* && (5.42) \\
&\vdots \\
c_{0n} &= J_n^* && (5.43)
\end{aligned}$$

At the subsystem level (for the ith stage) :

$$\begin{aligned}
&\textbf{Given} && z_{sh}, y \\
&\textbf{Minimize} && J_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y) = \sum_{j \neq m} \left(\frac{x_{if}^j - \widehat{x_{if}^j}}{\Pi_j} \right)^2 + \left(\frac{u_{if} - \widehat{u_{if}}}{U} \right)^2 + \left(\frac{x_{i0}^m - \widehat{x_{i-1f}^m}}{M} \right)^2 \\
&\textbf{With respect to} && \bar{z}_{d_i}, z_{c_i} \\
&\textbf{Subject to} && g_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) \leq 0 && (5.44) \\
& && c(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) = 0 && (5.45) \\
& && \dot{x}_i = s_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) && (5.46)
\end{aligned}$$

with Π_j, U and M : different normalization coefficients on the objective function at the subsystem level which are used to enhance the behavior of the optimization process. For $i = 1$, we have $\widehat{x_{i-1f}^m} = \widehat{x_{10}^m}$, for $i = n$, $\left(\frac{u_{if} - \widehat{u_{if}}}{U}\right)^2$ is not considered in the objective function and $x_{if}^j - \widehat{x_{if}}$ are the orbit injection conditions.

Remarks concerning the second formulation

The main advantage of the second formulation with respect to the first one is the improvement of the satisfaction of the system level coupling constraints by the modification of the different subsystem objective functions. Indeed, in this formulation, the convergence of the subsystems implies the satisfaction of the system level objective function. A second advantage of this formulation is the avoidance of the equality constraints h and c_i at the subsystem level. This is due to the inclusion of these constraints into the penalized objective function. On the other hand, this formulation increases the number of coupling variables (insertion of the estimated mass of the first stage into the coupling variables set) and the coupling constraints (n coupling constraints instead of $n-1$ for the first formulation).

In the first and second formulations, the system level aims to coordinate the optimization process. To this end, these formulations require equality constraints at the system level. Even if the second formulation improves the global behavior of the optimization process, these two proposed formulations may cause some numerical difficulties during the optimization (*e.g.* difficulty to satisfy the constraints, search spaces of valid solutions are very restricted). For this reason, an alternative formulation is proposed, in order to get rid of the equality constraints (Eq. 5.32 and 5.40).

5.3.3 Third formulation

Principle

The aim of the third formulation is to avoid the coupling equality constraints at the system-level. The equality constraints at the system level of the first formulation concern the estimation of the optimized masses f_i^* found during the different subsystem optimizations, in order to ensure the consistency of the optimization process. We can notice that the i^{th} subsystem requires only the estimations of $f_{j,j>i}$. Thus, an iterative handling of the f_i may be considered at the subsystem level in order to remove the coupling equality constraints at the system level. For that purpose, the different stages are optimized from the uppermost to the lowest stages, and once a stage optimization is performed, the optimized mass of the considered stage is transmitted to the lower subsystem which can begin its own optimization. In this formulation, the use of the coupling variables $\widehat{x_{if}^m}$ is not necessary anymore. Thus, the search space at the system level is reduced by $n-1$ dimensions. The coupling variable set used in the third formulation is the following :

$$y = \{\widehat{x_{1f}^{-m}}, \cdots, \widehat{x_{n-1f}^{-m}}, \widehat{u_{1f}}, \cdots, \widehat{u_{n-1f}}, y_0\} \tag{5.47}$$

The problem formulation of the different subsystems is the same as in the first formulation : each subsystem aims at minimizing its own mass and at satisfying its equality and inequality constraints. The only difference between the first and third formulations at the subsystem level concerns the way to perform the different subsystem optimizations. For the first formulation, the different subsystem optimizations can be performed independently. This is not true for the third formulation (non hierarchical process).

The optimization process is the following :

Formulation II.4
At the system level :

$$\begin{aligned} \textbf{Minimize} \quad & f = \sum_{i=1}^{n} f_i^*(z_{sh}, y) + M_{upper\ composite} \\ \textbf{With respect to} \quad & z_{sh}, y \\ \textbf{Subject to} \quad & g_0(z_{sh}, y) \leq 0 \end{aligned} \tag{5.48}$$

At the subsystem level :

```
i=n
While i>0
    For the i^th stage :
        Given              $z_{sh}, y, \sum_{j=i+1}^{n} f_j$
        Minimize           $f_i(\bar{z}_{d_i}, z_{sh}, y)$
        With respect to    $\bar{z}_{d_i}, z_{c_i}$
        Subject to         $g_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) \leq 0$    (5.49)
                           $h_i(x_{i_f}) = 0$    (5.50)
                           $c_i(x_{i_f}, u_{i_f}, y) = 0$    (5.51)
                           $c(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) = 0$    (5.52)
                           $\dot{x}_i = s_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i)$    (5.53)
i←i-1
```

where Eq. (5.34) are only considered for $i = n$ and Eq. (5.35) for $i \neq n$.

Remarks concerning the third formulation

This formulation allows to avoid the mass coupling constraints at the system level and consequently reduces the dimension of the search space at this level. Furthermore, this formulation allows the user to have multidisciplinary feasible solutions even if the optimization process has not converged at the system level. Indeed, the sufficient condition to have a multidisciplinary feasible design is only the convergence of the different subsystems. Moreover, if the system level optimization problem does not involve constraints on the shared and coupling variables (which is the case in the application case described in the following chapter), this obtained design is a real feasible design. As a consequence, in this formulation, the different optimizations of the stages cannot be performed in parallel, which may be a drawback when calculation parallelization is desirable in order to save computation time.

5.3.4 Fourth formulation

Principle

The fourth formulation is quite different from the previous three. Indeed, this formulation uses the flight-phase decomposition but does not aim at optimizing the different stage designs at the subsystem-level anymore. This formulation separates the handling of the design variables to the handling of the trajectory variables. In this bi-level formulation, the system level optimizer is responsible for handling all the design variables z_d and the subsystems only aim at ensuring the feasibility of the trajectory by handling the control variables z_c. In this formulation, we distinguish the constraints depending only on the design and coupling variables g_{d0} from the ones depending on the trajectory g_{ci}. In the same manner, c_j stands for the coupling constraints on the state vector whereas c are the coupling constraints of the initial problem. The coupling variable set at the system level is the same as in the third formulation. With the same notations, the fourth formulation is the following:

Formulation II.5
At the system level :

$$
\begin{aligned}
\textbf{Minimize} \quad & f = \sum_{i=1}^{n} f_i(z_{sh}, \bar{z}_{d_i}, y) + M_{upper\ composite} \\
\textbf{With respect to} \quad & z_{sh}, \bar{z}_{d_i}, y, \quad i = 1, \cdots, n \\
\textbf{Subject to} \quad & g_{d0}(z_{sh}, \bar{z}_{d_i}, y) \leq 0 && (5.54) \\
& c_0(x_{i_f}, u_{i_f}, y) = 0, \quad i = 1, \cdots, n && (5.55)
\end{aligned}
$$

c_0 is a constraint vector composed of the J_i, in the same manner as in the second formulation. At the subsystem level (for the ith stage) :

$$
\begin{aligned}
\textbf{Minimize} \quad & J_i = \sum_{j \neq m} \left(\frac{c_{ij}(x_{i_f}^{j}, u_{i_f}, y)}{\Pi_j} \right)^2 \\
\textbf{With respect to} \quad & z_{c_i} \\
\textbf{Subject to} \quad & g_{ci}(z_{sh}, \bar{z}_{d_i}, z_{c_i}, y, x_i) \leq 0 && (5.56) \\
& c(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) = 0 && (5.57) \\
& \dot{x}_i = s_i(\bar{z}_{d_i}, z_{sh}, z_{c_i}, y, x_i) && (5.58)
\end{aligned}
$$

where c_{ij} stands for $(x_{i_f}^{j} - \widehat{x_{if}^{j}})$ and $(u_{i_f} - \widehat{u_{if}})$ and for i=n, $J_n = \sum_{j \neq m} \left(\frac{h_j(x_{n_f}^{j})}{\Pi_j} \right)^2$.

Remarks concerning the fourth formulation

The fourth formulation is different from the previous three in the way the coupling constraints are handled. Indeed, this formulation introduces the decomposition between the design and control variables and splits up only the trajectory into the different parts. In this manner, this method mixes a MDF formulation with a multiple shooting method in which the different parts of the trajectory are not only simulated but also optimized separately. This formulation is a compromise between the MDF formulation (in which all the optimization variables are handled at the system level) and the three first formulations (in which the system level handles the least number of optimization variables as possible).

Remark 5.3.2. *The fourth formulation can be also described within the IDF paradigm. Indeed, since the subsystems just aim at satisfying the equations (5.55), these can be considered as disciplinary analyzers (see definition 1.2.1). In this case, the control variables z_{c_i} can be considered as state variables and the c_{ij} are just residuals to cancel under some constraints.*

Table 5.1 surveys the different characteristics of the formulations.

	F1	F2	F3	F4
Mass objective function at the subsystem level	✓	✗	✓	✗
State junctions explicitly handled as equality constraints	✓	✗	✓	✗
State junctions handled as quadratic objective function	✗	✓	✗	✓
Requirement of mass coupling variables	✓	✓	✗	✗
Independent optimizations of the subsystems	✓	✓	✗	✓

Table 5.1: Characteristics of the proposed formulations

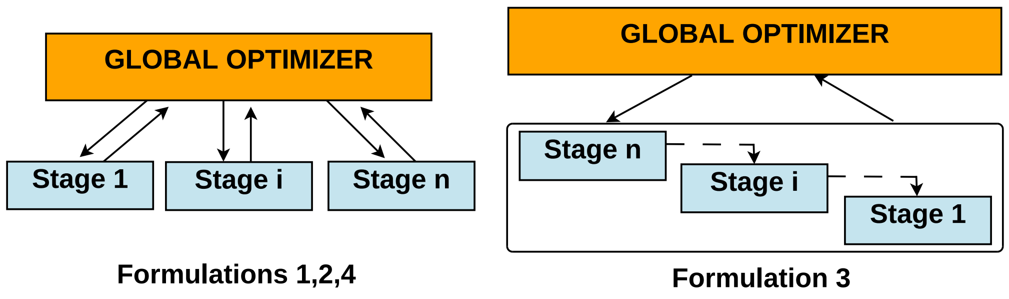


Figure 5.2: SWORD formulations

5.4 Conclusion

In this chapter, we have detailed formulations using the stage-wise decomposition. After the description of the LVD MDO problem, we based on the trajectory analysis to propose four formulations which allow to distribute the complexity of the initial MDO problem from the system to the subsystem levels in order to improve the overall efficiency of the optimization process. Three of the formulations (1, 2 and 4) are hierarchical and allow to parallelize the different subsystem optimizations. The non hierarchical formulation (3) allows to handle the mass coupling directly at the subsystem level and therefore involves the less dimension of the system level search space. The future works will naturally consist in applying the different proposed formulations in a practical LVD problem in order to compare the proposed formulations both against each other and with respect to a reference MDO method. Before performing this comparison, we have to elaborate an application case which allows us to compare the formulations. This is the purpose of the next chapter.

Chapter 6

Application case : optimization of a three-stage-to-orbit launch vehicle

Contents

6.1 Introduction

In order to test the different formulations proposed in the previous chapter, a study case has to be elaborated. This chapter is devoted to the description the application case and the different used disciplinary models. The problem to solve consists of the optimization of a three-stage-to-orbit expendable launch vehicle. The objective function is the Gross-Lift-Off-Weight. The payload mass is fixed to 4 tons. The target orbit is a 250-35786km altitude Geostationary Transfer Orbit, with an injection at the perigee (250km). The considered disciplines are the propulsion, the aerodynamics, the trajectory, and the mass budget & geometry design. The launch vehicle is composed of three liquid propulsion stages ($LOX/LH2$) and a fairing (not subject to optimization). The number of stages (discrete variable) is not optimized. Since this application case is performed at the early phase design studies, simplified models (and response surfaces) have been developed in order to avoid time consuming calculation codes. Nevertheless, different couplings have been taken into account in order to be representative of the real interactions between the different disciplines. For a complete description on the launch vehicle models, one can consult [Marty, 1986; Humble et al., 1995; Sutton and Biblarz, 2001].

This chapter is organized as follows : In the first section of this chapter, the different disciplines are briefly described. Then, the optimization variables and the different disciplines interactions are detailed. The last two sections of this chapter concern the description of the initial and proposed formulations of the MDO LVD problem.

6.2 Description of the disciplines

6.2.1 Propulsion

In this study, all the propulsion stages have been chosen as cryogenic propulsion stages. Many other choices of propellants are possible (*e.g.* $LOX/RP-1$ [Balesdent et al., 2010b, 2011e], N_2O_4/N_2H_4, *etc.*). We have chosen the cryogenic propulsion because it is one of the main candidates for future launch vehicles due to its performance. The different variables which play a part in the liquid propulsion design process are the chamber pressure Pc, the nozzle exit pressure Pe and the mixture ratio Rm. The chamber (respectively nozzle exit) pressure Pc (respectively Pe) is the pressure of the gases in the combustion chamber of the engine (respectively at the exit of the nozzle). The mixture ratio Rm is the ratio between the oxidizer (LOX) and the fuel ($LH2$). The aim of the propulsion module is to compute the specific impulse (Isp) from Pc, Pe and Rm. The Isp is used to calculate the thrust of the launch vehicle and represents the impulse (momentum variation) per unit mass of used propellant.

Model

From the mixture ratio Rm ; the isentropic coefficient at the throat γ_t, the flame temperature T_c and the molecular mass at combustion M_c for the couple $LOX/LH2$ are computed by using response surfaces generated from standard data [Humble et al., 1995; Sutton and Biblarz, 2001] (Fig. 6.1). From these parameters, the characteristic velocity of the propellants (C^*) is determined :

$$C^* = \eta_c . \frac{\sqrt{\gamma_t . R . T_c}}{\gamma_t \left(\frac{2}{\gamma_t+1}\right)^{\frac{\gamma_t+1}{2(\gamma_t-1)}}} \tag{6.1}$$

with $R = 8314/M_c$ the gas constant and η_c the combustion efficiency (0.98). From the isentropic coefficient and the pressure ratio $\frac{Pc}{Pe}$, the nozzle area ratio ϵ is given by :

$$\epsilon = \frac{\left(\frac{2}{\gamma_t+1}\right)^{\frac{1}{\gamma_t-1}} \cdot \frac{Pc}{Pe}^{\frac{1}{\gamma_t}}}{\sqrt{\left(\frac{\gamma_t+1}{\gamma_t-1}\right) \cdot \left(1 - \left(\frac{Pe}{Pc}\right)^{\frac{\gamma_t-1}{\gamma_t}}\right)}} \tag{6.2}$$

Finally, the specific impulse is given by :

$$Isp = \lambda_n . \frac{C^*}{g_0} \left(\gamma_t \sqrt{\left(\frac{2}{\gamma_t - 1}\right) \cdot \left(\frac{2}{\gamma_t + 1}\right)^{\frac{\gamma_t+1}{\gamma_t-1}} \cdot \left(1 - \frac{Pe}{Pc}\right)^{\frac{\gamma_t-1}{\gamma_t}}} + \frac{\epsilon}{Pc}(Pe - P_a) \right) \tag{6.3}$$

with λ_n, the nozzle efficiency (0.98) and P_a the local atmospheric pressure.

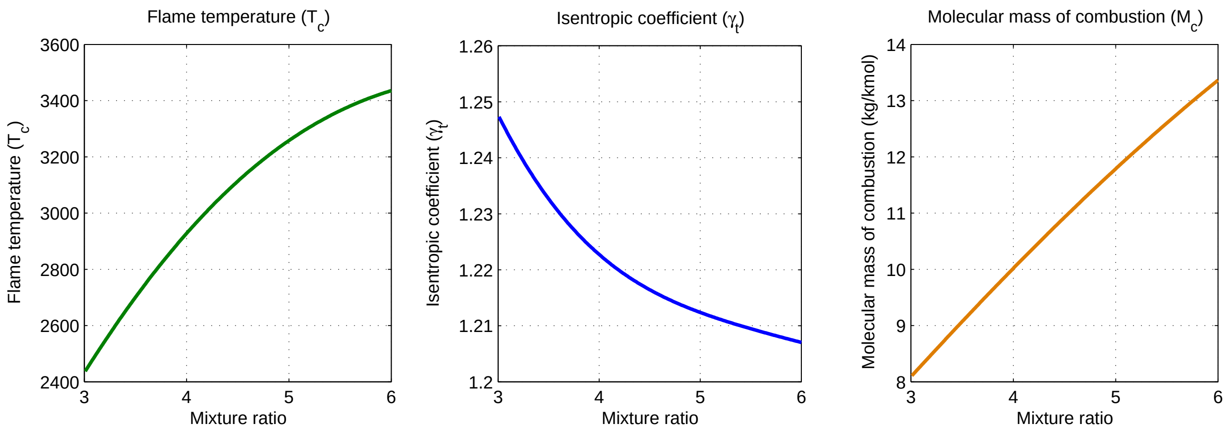


Figure 6.1: Propulsive parameters models

Optimization variables and induced constraints

We have chosen to handle Rm, Pc and Pe as optimization variables for this discipline. In order to avoid non realistic configuration, we introduce an inequality constraint between Pc and Pe which ensures the avoidance of the breakaway of the jet in the divergent skirt (Summerfield criterion [Summerfield, 1951]) :

$$Pe \leq 0.4 P_a(h) \tag{6.4}$$

with h the altitude of the launch vehicle. In the same manner, in order to avoid non realistic configurations despite the lack of complete structural and aerodynamic analyses, we introduce a limitation concerning the maximal nozzle exit diameter Dne :

$$Dne_i \leq 0.8 D_i \tag{6.5}$$

with Dne_i the maximal nozzle exit diameter of the i^{th} stage and D_i the diameter of the considered stage.

6.2.2 Aerodynamics

We use a zero-lift aerodynamics which is a generally valid assumption in the early design of launch vehicles. The drag coefficient (C_X) is interpolated from the Mach using piece-wise linear functions (Fig. 6.2). For simplification reasons, we do not take into account the influences of other parameters (*e.g.* the angle of attack) on the drag coefficient calculation. Once again, this assumption is valid in early design studies. The coefficient C_X is used to calculate the drag of the launch vehicle all along trajectory simulation. The Mach is defined as the ratio between the velocities of the launch vehicle and the sound at the considered altitude which is a function of atmospherical density and temperature. These data are calculated according to the 1976 US Standard Atmosphere model [NASA et al., 1976].

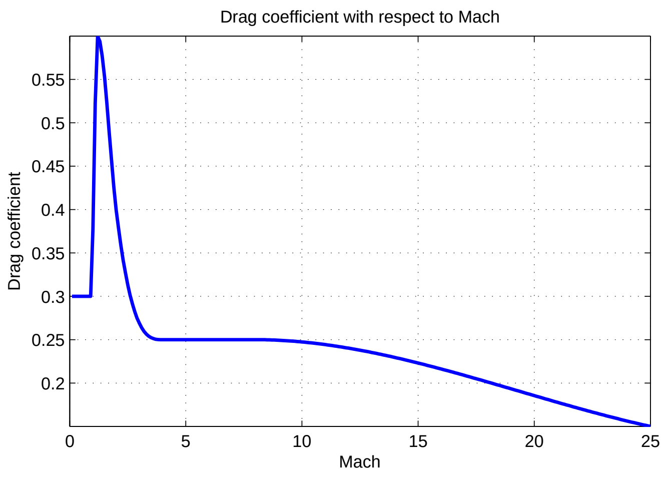


Figure 6.2: Drag coefficient as a function of the Mach number

6.2.3 Trajectory

The trajectory model used in this study is derived from the following classical 3D dynamics equations, written in an Earth-centered, Earth-fixed referential.

$$\dot{r} = V.\sin\gamma \tag{6.6}$$

$$\dot{V} = \frac{T\cos(\theta-\gamma)-D}{m} - g(r)\sin\gamma + \omega_E^2 r\cos\phi(\sin\gamma\cos\phi - \cos\gamma\sin\phi\cos\psi) \tag{6.7}$$

$$\dot{\gamma} = \frac{[L+T\sin(\theta-\gamma)]\cos\mu}{mV} + \left(\frac{V}{r} - \frac{g(r)}{V}\right)\cos\gamma + 2\omega_E\sin\psi\cos\phi + \frac{\omega_E^2 r\cos\phi(\cos\gamma\cos\phi + \sin\gamma\sin\phi\cos\psi)}{V} \tag{6.8}$$

$$\dot{\lambda} = \frac{V\cos\gamma\sin\psi}{r\cos\phi} \tag{6.9}$$

$$\dot{\phi} = \frac{V\cos\gamma\cos\psi}{r} \tag{6.10}$$

$$\dot{\psi} = \frac{[L + T\sin(\theta-\gamma)]\sin\mu}{mV\cos\gamma} + \frac{V\cos\gamma\sin\psi\tan\phi}{r} + 2\omega_E(\sin\phi - \cos\psi\cos\phi\tan\gamma) + \frac{\omega_E^2 r\sin\phi\cos\phi\sin\psi}{V\cos\gamma} \tag{6.11}$$

$$\dot{m} = -q \tag{6.12}$$

with :

- r : radius (m),
- V : norm of the velocity vector ($m.s^{-1}$),
- γ : flight path angle (rad),
- ϕ : latitude (rad),
- λ : longitude (rad),
- ψ : flight path heading (rad),
- μ : bank angle (rad),
- θ : pitch angle (rad),
- ω_E : angular velocity of the Earth ($2\pi/86164.09\ rad/s$)
- T : Thrust (N)
- D : Drag (N),
- L : Lift (N),
- $g(r)$: gravity acceleration at r (m^2/s),
- m : mass (kg),
- q : mass flow rate ($kg.s^{-1}$).

The used referential is given in Figure 6.3.

Choice of state vector and model simplifications

We have chosen to use a 3 degrees of freedom non rotating round Earth model. Indeed, we only consider a planar trajectory in the equatorial plan. We have chosen to use this model because it is sufficiently representative of the real problem and it meets the main need of this study which is the development of MDO methods. The introduced simplifications induce :

$$\psi = \frac{\pi}{2} \tag{6.13}$$

$$\mu = 0 \tag{6.14}$$

$$\phi = 0 \tag{6.15}$$

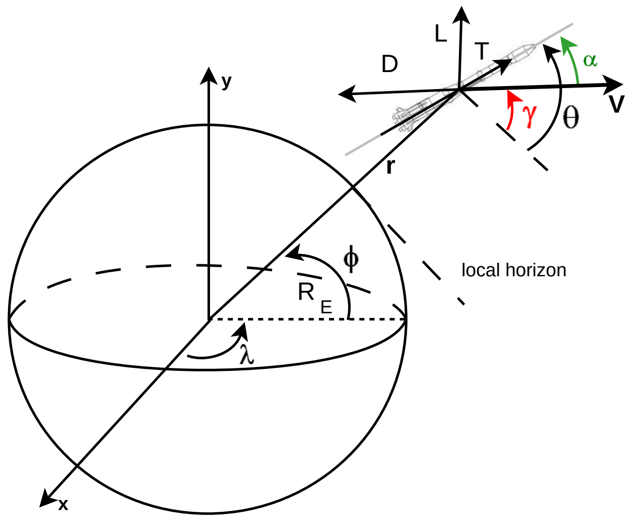


Figure 6.3: Earth-centered, Earth-fixed reference frame

Consequently, we have chosen to handle the following state vector :

$$\text{x}=\begin{bmatrix} \text{r} \\ \text{V} \\ \gamma \\ \lambda \\ \text{m} \end{bmatrix}$$

with :

- α : angle of attack (rad)
- R_E : Earth radius (6378.137 km)

State equations

With the simplifications described above, the state equations become :

$$\begin{aligned} \dot{r} &= V.\sin\gamma && (6.16)\\ \dot{V} &= -\frac{1}{2}\rho(r)\frac{S_{ref}.C_X(Mach)}{m}V^2 - g(r).\sin\gamma + \frac{T.\cos(\theta-\gamma)}{m} && (6.17)\\ \dot{\gamma} &= \left(\frac{V}{r}-\frac{g(r)}{V}\right)\cos\gamma + \frac{T.\sin(\theta-\gamma)}{m.V} && (6.18)\\ \dot{\lambda} &= \frac{V.\cos\gamma}{r} && (6.19)\\ \dot{m} &= -q && (6.20) \end{aligned}$$

with :

- S_{ref} : aerodynamic reference surface,

- $\rho(r)$: local atmospheric density at r,

The thrust is given by :

$$T = g_0.q.Isp(r) \tag{6.21}$$

The air density in function of the altitude is given by (the employed geoid is a spherical Earth) :

$$\rho(r) = \rho_0 exp(-\frac{r - R_E}{h_{ref}}) \tag{6.22}$$

with :

- $\rho_0 = 1.22557 kg.m^3$,
- $h_{ref} = 7254.24m$.

Using the keplerian gravity model without perturbation, the gravity acceleration in function of r is given by :

$$g(r) = \frac{G.M}{r^2} \tag{6.23}$$

with :

- G : gravitational constant $(6.67428.10^{11} m^3 k^{-1} s^{-2})$,
- M : mass of the Earth $(5.9736.10^{24} kg)$.

The dynamic pressure $Pdyn$ during the flight is given by :

$$Pdyn = \frac{1}{2}\rho(r)V^2 \tag{6.24}$$

The axial load factor n_f is determined by the following relationship :

$$n_f = \frac{g_0.q.Isp(r) - \frac{1}{2}\rho(r)S_{ref}V^2 C_X \cos(\theta - \gamma)}{m.g_0} \tag{6.25}$$

The optimal control is performed through a direct method [Betts, 1998]. The used control law is a parametric law determined by defining discretization points all along the trajectory. These points are optimized in order to satisfy the optimality conditions of the problem. The control law is defined by interpolation between the discretization points. In order to reduce the computation volume, we have chosen piece-wise linear functions to interpolate the control law (Fig. 6.4).

During the flight, the fairing is jettisoned when the aerothermal flux gets below $1135 W/m^2$. The aerothermal flux (ϕ) is evaluated with using the formula :

$$\phi = \frac{1}{2}\rho(r)V^3 \tag{6.26}$$

In order to avoid any unrealistic trajectories with regard to the structural loads, the maximal angle of attack is constrained to 4° during the atmospheric flight (before the fairing jettisoning) and 15° during the exoatmospheric flight (after the fairing jettisoning). The ODE dynamics are integrated numerically with a 4^{th} order Runge Kutta numerical solver [Dormand and Prince, 1980].

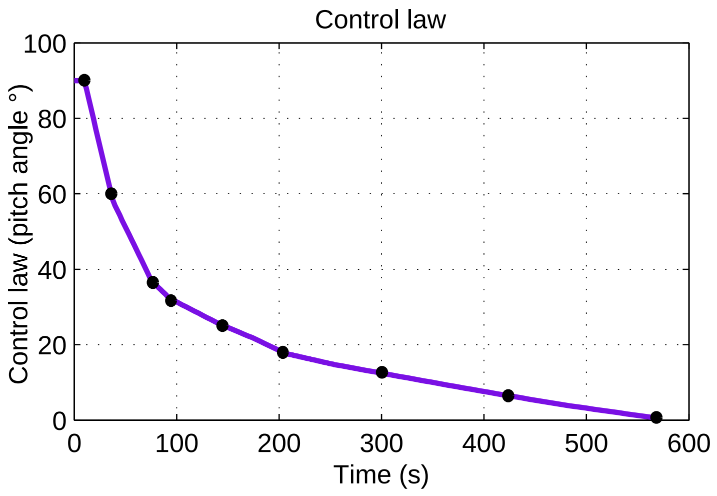


Figure 6.4: Example of control law

6.2.4 Mass budget and geometry design

The selected mass budget & geometry design models essentially come from the CNES PERSEUS project models [Amouroux, 2008] and classical mass estimation relationships developed in [Humble et al., 1995; Sutton and Biblarz, 2001]. This module aims at calculating the dry mass of the launch vehicle in order to compute the objective function and to simulate the trajectory. The dry mass of each stage is the sum of the main launch vehicle components (Fig. 6.5) :

- tanks,
- turbopumps,
- pressurant system,
- combustion chamber,
- nozzle,
- engine.

The complete description of the dry mass computation model is given in Appendix A. The total dry mass (Md) depends on the following parameters (Fig. 6.5):

- propellant mass : Mp,
- mixture ratio : Rm,
- chamber pressure : Pc,
- nozzle exit pressure : Pe,
- mass flow rate : q,
- stage diameter : D.

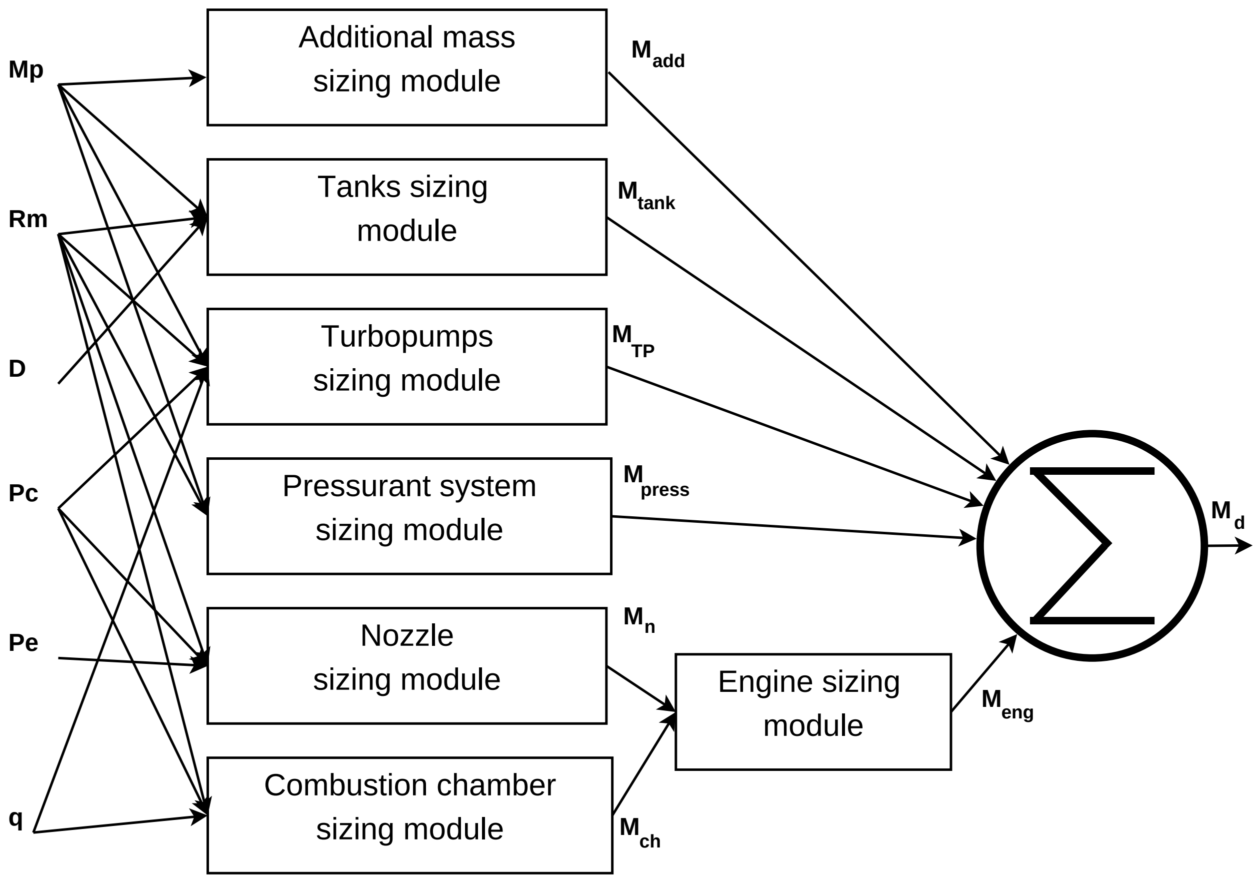


Figure 6.5: Dry mass sizing module

with :

- M_{tank} : mass of the tanks,
- M_{eng} : mass of the engine,
- M_n : mass of the nozzle,
- M_{ch} : mass of the combustion chamber,
- M_{press} : mass of the pressurant system,
- M_{TP} : mass of the turbopumps,
- M_{add} : additional masses.

In order to take into account the loads sustained by the launch vehicle during the flight, a linear dependency (c_f) of the dry mass on the maximal sustained axial load factor has been added into the mass sizing module (Fig. 6.6). This term introduces a coupling between the trajectory and the mass sizing disciplines.

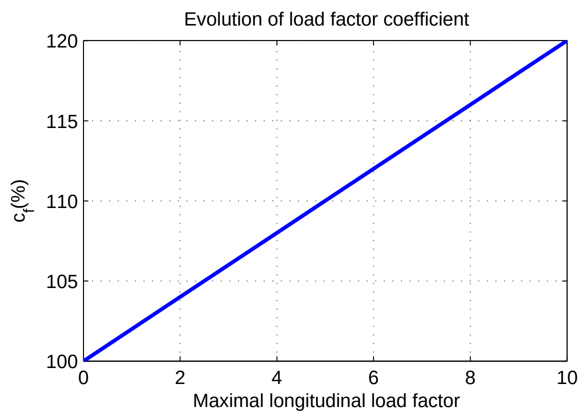


Figure 6.6: Evolution of the load factor coupling coefficient with respect to the maximal axial load factor

The total dry mass is given by the formula :

$$Md = [M_{tanks} + M_{eng} + M_{press} + M_{TP} + M_{add}].c_f \tag{6.27}$$

The dry mass calculation module has been compared with classical cryogenic stages found in ESA launch vehicles catalog [European Space Agency, 2004]. Results show that the modelization is quite valid and gives results in the same order of magnitude as the real stages (Fig. 6.7) and consequently are satisfactory for early phase design studies.

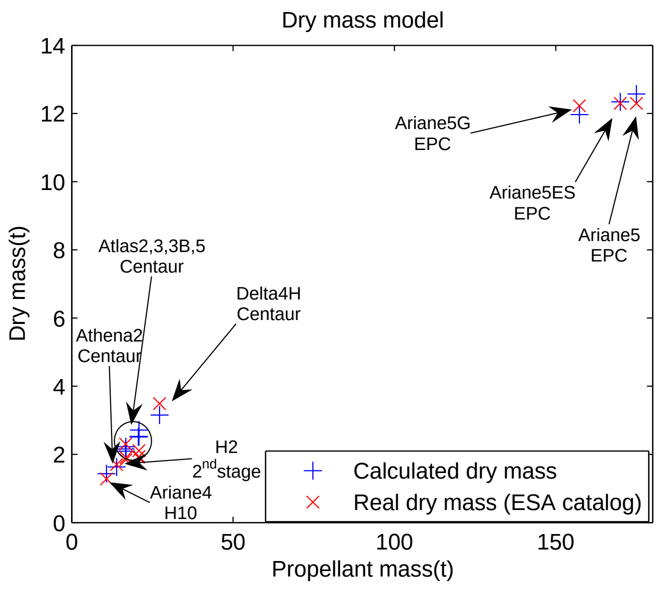


Figure 6.7: Results obtained with dry mass model

6.3 Optimization variables and disciplinary interactions

In this section, we define the optimization variables and we describe the different interactions between the considered disciplines.

6.3.1 Optimization variables

The optimization problem for a three stage launch vehicle consists of 30 optimization variables divided as follows :

- 6 for the mass budget (Mp_i, Rm_i, $i = 1, \cdots, 3$),
- 9 for the propulsion (Pc_i, Pe_i, T/W_i, $i = 1, \cdots, 3$),
- 3 for the aerodynamics (D_i, $i = 1, \cdots, 3$),
- 12 for the trajectory(θ_{ij}, $i = 1, \cdots, 3$, $j = 1, \cdots, 4$)

All of the propulsion and aerodynamics variables are used in the mass budget module to compute the dry mass. Concerning the propulsion, the selected optimization variables are the propellant mass Mp, the mixture ratio Rm, the initial thrust to weight ratio $\frac{T}{W}$, the chamber pressure Pc and the nozzle exit pressure Pe. We have chosen to handle the thrust to weight ratio instead of the mass flow rate q because the search domain of this parameter can be more easily estimated than the one of the mass flow rate. In order to simulate the trajectory, a relationship between the mass flow rate and the thrust to weight ratio can be easily determined by introducing the mass index of the considering stage. Considering the aerodynamics, the selected optimization variables are the diameters of the stages D.

6.3.2 Interactions between the different disciplines

The following N2-Chart sums up the different interactions between the disciplines for one stage (Fig. 6.8). This chart is used to describe the data flows between the different disciplines during the multidisciplinary analysis. This chart can be generalized to describe more generally the MDO methods using the same notations [Lambe, 2011].

From the diameters of the different stages, the drag coefficient is computed and transmitted to the trajectory module. In the same manner, the specific impulse is computed in the propulsion module from the mixture ratio, the chamber and nozzle exit pressures. The mass budget computes the dry mass of the stages from the diameter, the mixture ratio, the chamber and nozzle pressures, the weight on thrust ratio and the propellant mass. This module is in charge of the computation of the objective function (GLOW). Finally, from all the variables, the trajectory module simulates the trajectory and returns the state vector at the final time.

For the “mass budget” module requires the maximal load factor in order to compute the dry mass, a coupling is necessary between the trajectory and mass budget modules. This coupling induces a loop between these two disciplines and requires an estimation of the load factor to compute at first the dry mass and so begin the loop (Fixed Point Iteration). We will see later in this chapter how to handle this coupling without requiring a FPI.

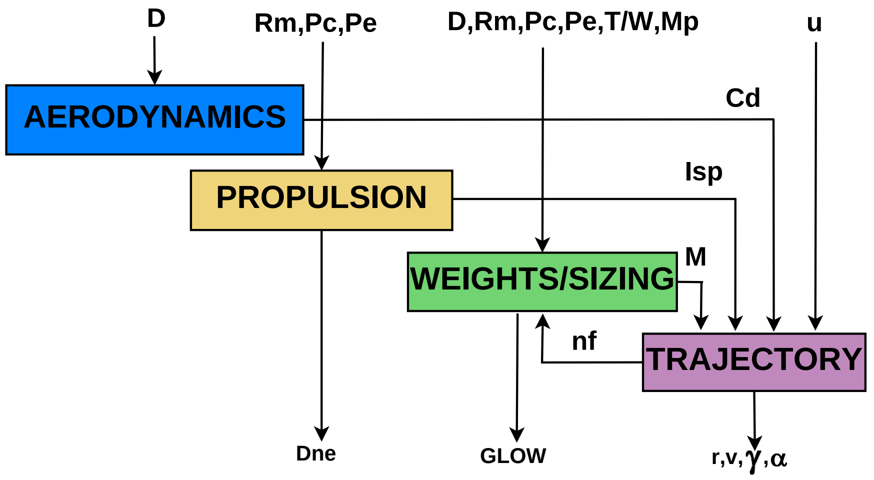


Figure 6.8: N2 Chart for one stage

6.4 MDO problem formulation

6.4.1 Initial formulation

Using the mathematical MDO notations developed previously, the original formulation of the Launch Vehicle MDO problem is the following :

$$
\begin{aligned}
\textbf{Minimize} \quad & f(z) = \sum_{i=1}^{3}(Mp_i + Md_i) + M_{upper\ composite} \\
\textbf{With respect to} \quad & z = \{Mp_i, \tfrac{T}{W}_i, Rm_i, Pc_i, Pe_i, D_i, \theta_{ij}\}, \quad i = 1, \cdots, 3 \\
\textbf{Subject to} \quad & g_1 : |\alpha_{max\ 1}| - 4^\circ \leq 0 && (6.28)\\
& g_2 : |\alpha_{max\ 2}| - 15^\circ \leq 0 && (6.29)\\
& g_{3\ldots5} : 0.4 - \frac{Pe_i}{Pa(r)} \leq 0 && (6.30)\\
& g_{6\ldots8} : Dne_i - 0.8\ D_i \leq 0 && (6.31)\\
& h_1 : r_f - r_{orbit} = 0 && (6.32)\\
& h_2 : v_f - v_{orbit} = 0 && (6.33)\\
& h_3 : \gamma_f - \gamma_{orbit} = 0 && (6.34)
\end{aligned}
$$

with θ_{ij} the different variables of the control law of the ith stage ($j = 1, \cdots, 4$).

6.4.2 MDF formulation

In order to break the FPI between the trajectory and the mass budget modules, the initial formulation has been transformed in order to perform sequentially the MDA at the subsystem level. To this end, an estimation of maximal load factor sustained during the flight ($\widehat{n_{f_{max}}}$) has been included into the optimization variable set and a coupling constraint has been added to the constraint set (Eq. 6.42). We assume that the maximal load factor occurs during the flight of the first stage, that allows to introduce only one coupling variable (and consequently one coupling constraint). This approximation is valid for expendable launch vehicles which lift off from the ground and has been verified by simulation *a posteriori* (once the optimization performed). The use of FPI is discussed in the appendix B. The resulting formulation is the following :

$$\begin{aligned}
&\textbf{Minimize} && f(z) = \sum_{i=1}^{3}(Mp_i + Md_i) + M_{upper\ composite} \\
&\textbf{With respect to} && z = \{Mp_i, \tfrac{T}{W}_i, Rm_i, Pc_i, Pe_i, D_i, \theta_{ij}\}, \quad i = 1, \cdots, 3 \\
& && y = \{\widehat{n_{f_{max}}}\} \\
&\textbf{Subject to} && g_1 : |\alpha_{max\ 1}| - 4^\circ \leq 0 && (6.35) \\
& && g_2 : |\alpha_{max\ 2}| - 15^\circ \leq 0 && (6.36) \\
& && g_{3\ldots5} : 0.4 - \tfrac{Pe_i}{Pa(r)} \leq 0 && (6.37) \\
& && g_{6\ldots8} : Dne_i - 0.8\ D_i \leq 0 && (6.38) \\
& && h_1 : r_f - r_{orbit} = 0 && (6.39) \\
& && h_2 : v_f - v_{orbit} = 0 && (6.40) \\
& && h_3 : \gamma_f - \gamma_{orbit} = 0 && (6.41) \\
& && c : \widehat{n_{f_{max}}} - max(\widehat{n_{f_{max}}}, n_{f_{max}}) = 0 && (6.42)
\end{aligned}$$

The use of the *max* in Eq. 6.42 symbolizes the fact that the configuration of the launch vehicle is still valid when the estimated maximal load factor taken into account in the weights & sizing module is greater than the real load factor observed during the trajectory simulation. In this case, the configuration of the launch vehicle is consistent even if the structure is oversized with respect to the sustained loads. We can note that this formulation involves one coupling variable (and corresponding constraint). For convenience reason, we will use this formulation to implement the MDF method in order to save computation time avoiding the FPI within the MDA.

6.5 Stage-wise decomposition formulations

6.5.1 First formulation

In the first formulation, all the subsystems have to minimize their own mass. Since a stage optimizer requires the knowledge of the upper stage masses, these ones are estimated at the system level ($\widehat{Mu_{stage_i}}$) and have to be equal to the real optimized mass ($Mu^*_{stage_i}$) at convergence (satisfaction of the coupling constraints (6.43) and (6.44)).

Remark 6.5.1. *For all our proposed formulations, we use the angle of attack $\widehat{\alpha}$ instead of the pitch angle $\widehat{\theta}$ at the system level. The pitch angle (real control law coupling variable) is given by the relation $\widehat{\theta} = \widehat{\alpha} + \widehat{\gamma}$. This allows to handle the inequality constraints on the angle of attack (Eq. 6.28,6.29) directly in the domain definition of the system level optimization variable $\widehat{\alpha}$ (bounds of the optimization variable). In the same manner, we avoid the subsystem level equality coupling constraints about the control law by using directly the system level control parameters (given by $\widehat{\gamma}+\widehat{\alpha}$) as first (and last) parameters of the parametric control law* *at the subsystem level. Moreover, as the dynamics is invariant with respect to the longitude angle (which does not intervene in the right members of the equation system [6.16–6.20]), we do not handle the coupling on the longitude in the system level optimization coupling variables set. This variable can be obtained once the optimization is performed by simply simulating the trajectory with the optimal values of the other variables. Finally, for flexibility reason, the coupling constraint c_i can be handled with using the inequality constraint $n_{f_{max}} - \widehat{n_{f_{max}}} \leq 0$.*

Optimization at the system level

At the system level, a coordinator-optimizer aims to minimize the GLOW. It handles the shared variables z_{sh} and the coupling variables y. The optimization problem at the system level is :

$$\begin{aligned}
&\textbf{Minimize} && f(z_{sh}, y) = GLOW \\
&\textbf{With respect to} && z_{sh} = \{D_2, D_3\} \\
& && y = \{\widehat{r_{f1}}, \widehat{r_{f2}}, \widehat{v_{f1}}, \widehat{v_{f2}}, \widehat{\gamma_{f1}}, \widehat{\gamma_{f2}}, \widehat{\alpha_{f1}}, \widehat{\alpha_{f2}}, \widehat{Mu_{stage_1}}, \widehat{Mu_{stage_2}}, \widehat{n_{f_{max}}}\} \\
&\textbf{Subject to} && c_{01} : \widehat{Mu_{stage_1}} - Mu^*_{stage_1} = 0 \qquad (6.43) \\
& && c_{02} : \widehat{Mu_{stage_2}} - Mu^*_{stage_2} = 0 \qquad (6.44)
\end{aligned}$$

Optimization at the subsystem level

The different stages are optimized simultaneously. The optimizers at the subsystem level have to solve the same problem, which is analog to minimize the total mass of a single-stage launch vehicle with regard to the corresponding elementary flight phase. For the i^{th} stage, the corresponding subsystem optimizer handles the variables $\bar{z}_i$ and solves the following problem :

$$\begin{aligned}
&\textbf{Given} && z_{sh}, y \\
&\textbf{Minimize} && f_i(z_{sh}, \bar{z}_i, y) = Md_{stage_i} + Mp_{stage_i} \\
&\textbf{With respect to} && \bar{z}_i = \{D_i, Mp_i, \left(\tfrac{T}{W}\right)_i, Rm_i, Pc_i, Pe_i, \theta_{ij}\} \\
&\textbf{Subject to} && c_i : \widehat{n_{f_{max}}} - max(\widehat{n_{f_{max}}}, n_{f_{max}}) = 0 \qquad (6.45) \\
& && g_{i1} : 0.4 - \tfrac{Pe_i}{Pa(r)} \leq 0 \qquad (6.46) \\
& && g_{i2} : Dne_i - 0.8 D_i \leq 0 \qquad (6.47) \\
& && g_{i3} : |\alpha_{i_{max}\,1}| - 4^\circ \leq 0 \qquad (6.48) \\
& && g_{i4} : |\alpha_{i_{max}\,2}| - 15^\circ \leq 0 \qquad (6.49) \\
& && h_{i1} : \widehat{r_{f_i}} - r_{f_i} = 0 \qquad (6.50) \\
& && h_{i2} : \widehat{v_{f_i}} - v_{f_i} = 0 \qquad (6.51) \\
& && h_{i3} : \widehat{\gamma_{f_i}} - \gamma_{f_i} = 0 \qquad (6.52)
\end{aligned}$$

D_i is taken into account only for i=1 and $n_{f_{max}}$ is the maximal load factor during the flight of the considered stage. In the subsystem optimizations, the flight phase final state distances to the system level targets are handled by equality constraints (6.50)-(6.52).

6.5.2 Second formulation

This formulation uses the classical Collaborative Optimization [Braun et al., 1996] applied to the stage-wise decomposition.

Optimization at the system level

In this formulation, the system level optimizer does not control explicitly the couplings about the different stage payload masses anymore but only ensures that at the convergence, the objective functions of the subsystems are equal to zero. Thus, three equality constraints (Eq.(6.53)-(6.55)) have to be handled at the system level and an additional variable ($\widehat{GLOW}$, estimation of the GLOW) is added to the problem.

Minimize $$f(z_{sh}, y) = \widehat{GLOW}$$

With respect to $$z_{sh} = \{D_2, D_3\}$$

$$y = \{\widehat{r_{f_1}}, \widehat{r_{f_2}}, \widehat{v_{f_1}}, \widehat{v_{f_2}}, \widehat{\gamma_{f_1}}, \widehat{\gamma_{f_2}}, \widehat{\alpha_{f_1}}, \widehat{\alpha_{f_2}}, \widehat{GLOW}, \widehat{Mu_{stage_1}}, \widehat{Mu_{stage_2}}, \widehat{n_{f_{max}}}\}$$

Subject to

$$c_{01} : f_1^* = 0 \tag{6.53}$$

$$c_{02} : f_2^* = 0 \tag{6.54}$$

$$c_{03} : f_3^* = 0 \tag{6.55}$$

Optimization at the subsystem level

At the subsystem level, all the equality constraints h do not appear explicitly and are included in the objective function. Indeed, each stage aims at minimizing a quadratic sum between real values and targets given by the system level optimizer. In this formulation, the couplings about the different stage masses are also taken into account in the objective function.

Given $$z_{sh}, y$$

Minimize

$$f_i(z_{sh}, \bar{z}_i, y) = \left(\frac{\widehat{r_{f_i}} - r_{f_i}}{R}\right)^2 + \left(\frac{\widehat{v_{f_i}} - v_{f_i}}{V}\right)^2 + \left(\frac{\widehat{\gamma_{f_i}} - \gamma_{f_i}}{\Gamma}\right)^2 + \left(\frac{Mp_i + Md_i + \widehat{Mu_{stage_i}} - \widehat{Mu_{stage_{i-1}}}}{M}\right)^2$$

With respect to $$\bar{z}_i = \{D_i, Mp_i, \left(\frac{T}{W}\right)_i, Rm_i, Pc_i, Pe_i, \theta_{ij}\}$$

Subject to

$$c_i : \widehat{n_{f_{max}}} - max(\widehat{n_{f_{max}}}, n_{f_{max}}) = 0 \tag{6.56}$$

$$g_{i1} : 0.4 - \frac{Pe_i}{Pa(r)} \leq 0 \tag{6.57}$$

$$g_{i2} : Dne_i - 0.8D_i \leq 0 \tag{6.58}$$

$$g_{i3} : |\alpha_{i_{max}\ 1}| - 4^\circ \leq 0 \tag{6.59}$$

$$g_{i4} : |\alpha_{i_{max}\ 2}| - 15^\circ \leq 0 \tag{6.60}$$

with R, V, Γ and M different normalization weights on the objective function, $\widehat{Mu_{stage_0}} = \widehat{GLOW}$ and D_i is only considered for i=1 in Eq. (6.5.2).

6.5.3 Third formulation

The third formulation removes all the equality constraints at the system level. Moreover, the estimated mass of the different stages are also removed from the optimization problem. In this formulation, the optimizations of the different subsystems are performed sequentially from the uppermost stage to the lowermost one. The stage masses are transmitted between the different subsystems without any action of the system level optimizer, that allows to automatically satisfy the mass couplings between the stages. In that way, all the coupling constraints concerning the state vector are satisfied at the subsystem level.

Optimization at the system level

At the system level, the coupling variables about the masses are removed.

$$\begin{aligned} \textbf{Minimize} \quad & f(z_{sh}, y) = GLOW \\ \textbf{With respect to} \quad & z_{sh} = \{D_2, D_3\} \\ & y = \{\widehat{r_{f_1}}, \widehat{r_{f_2}}, \widehat{v_{f_1}}, \widehat{v_{f_2}}, \widehat{\gamma_{f_1}}, \widehat{\gamma_{f_2}}, \widehat{\alpha_{f_1}}, \widehat{\alpha_{f_2}}, \widehat{n_{f_{max}}}\} \end{aligned}$$

Optimization at the subsystem level

At the subsystem level, the optimizations of the different stages can not be performed independently anymore. The flight phase final state distances to the desired values are handled as equality constraints. The subsystem level optimization problems are the same as in the first formulation.

i=3
While $i > 0$

$$\begin{aligned} \textbf{Given} \quad & z_{sh}, y, f^*_{k,k>i} \\ \textbf{Minimize} \quad & f_i(z_{sh}, \bar{z}_i, y) = Md_{stage_i} + Mp_{stage_i} \\ \textbf{With respect to} \quad & \bar{z}_i = \{D_i, Mp_i, \left(\tfrac{T}{W}\right)_i, Rm_i, Pc_i, Pe_i, \theta_{ij}\} \\ \textbf{Subject to} \quad & c_i : \widehat{n_{f_{max}}} - max(\widehat{n_{f_{max}}}, n_{f_{max}}) = 0 && (6.61) \\ & g_{i1} : 0.4 - \frac{Pe_i}{Pa(r)} \leq 0 && (6.62) \\ & g_{i2} : Dne_i - 0.8.D_i \leq 0 && (6.63) \\ & g_{i3} : |\alpha_{i_{max}\,1}| - 4^\circ \leq 0 && (6.64) \\ & g_{i4} : |\alpha_{i_{max}\,2}| - 15^\circ \leq 0 && (6.65) \\ & h_{i1} : \widehat{r_{f_i}} - r_{f_i} = 0 && (6.66) \\ & h_{i2} : \widehat{v_{f_i}} - v_{f_i} = 0 && (6.67) \\ & h_{i3} : \widehat{\gamma_{f_i}} - \gamma_{f_i} = 0 && (6.68) \end{aligned}$$

i=i-1
where D_i is taken into account only for i=1.

6.5.4 Fourth formulation

Optimization at the system level

As seen in Chapter 5, the fourth formulation splits up the optimization variables set into two subsets corresponding to the design and the control variables. The system level optimizer is subjected to the design and coupling constraints about the convergence of the different subsystems. The problem formulation at the system level is the following :

$$\begin{array}{rc}
\textbf{Minimize} & f(z_{sh}, \bar{z}_d, y) = GLOW \\
\textbf{With respect to} & z_{sh} = \{D_2, D_3\} \\
& \bar{z}_d = \{D_1, Mp_i, \left(\frac{T}{W}\right)_i, Rm_i, Pc_i, Pe_i, \theta_{ij}\}, \quad i = 1, \cdots, 3 \\
& y = \{\widehat{r_{f1}}, \widehat{r_{f2}}, \widehat{v_{f1}}, \widehat{v_{f2}}, \widehat{\gamma_{f1}}, \widehat{\gamma_{f2}}, \widehat{\alpha_{f1}}, \widehat{\alpha_{f2}}, \widehat{n_{f_{max}}}\} \\
\textbf{Subject to} & c_{01} : f_1^* = 0 \qquad (6.69) \\
& c_{02} : f_2^* = 0 \qquad (6.70) \\
& c_{03} : f_3^* = 0 \qquad (6.71) \\
& g_{01\ldots03} : 0.4 - \frac{Pe_{1\ldots3}}{Pa_{1\ldots3}(r)} \leq 0 \qquad (6.72) \\
& g_{04\ldots06} : Dne_{1\ldots3} - 0.8.D_{1\ldots3} \leq 0 \qquad (6.73)
\end{array}$$

Optimization at the subsystem level

Each subsystem only handles the control variables to minimize the quadratic sum of normalized deviations with respect to the system level targets. The problem formulation at the subsystem level is the following :

$$\begin{array}{rc}
\textbf{Given} & z_{sh}, \bar{z}_d, y \\
\textbf{Minimize} & f_i(z_{sh}, z_{di}, z_{ci}, y) = \left(\frac{\widehat{r_{f_i}} - r_{f_i}}{R}\right)^2 + \left(\frac{\widehat{v_{f_i}} - v_{f_i}}{V}\right)^2 + \left(\frac{\widehat{\gamma_{f_i}} - \gamma_{f_i}}{\Gamma}\right)^2 \\
\textbf{With respect to} & z_{c_i} = \{\theta_{ij}\} \\
\textbf{Subject to} & c_i : \widehat{n_{f_{max}}} - max(\widehat{n_{f_{max}}}, n_{f_{max}}) = 0 \qquad (6.74) \\
& g_{i1} : |\alpha_{i_{max}\ 1}| - 4° \leq 0 \qquad (6.75) \\
& g_{i2} : |\alpha_{i_{max}\ 2}| - 15° \leq 0 \qquad (6.76)
\end{array}$$

This formulation is a hybrid formulation between the third and the second. Since the different design variables are directly handled at the system level, all the different stage masses can be exactly known by the different subsystems and consequently this formulation allows to avoid the mass coupling handling.

6.6 Conclusion

In this chapter, we have detailed the three-stage-to-orbit LVD problem. To this end, we have briefly described the different variables, the different disciplines and the interactions between them. Secondly, the initial MDO formulation of the problem has been detailed. A coupling variable (and corresponding constraint) have been introduced in order to perform sequentially the multidisciplinary analysis. The proposed MDO formulations have been explained within the framework of the three-stage-launch-vehicle application case.

The study case described in this chapter will be used in the next chapter to numerically compare the proposed MDO formulations.

Chapter 7

Comparison of the different formulations

Contents

7.1 Introduction

This chapter is devoted to the comparison of the different proposed formulations [Balesdent et al., 2010a, 2011c]. In order to perform this comparison, we have to select a reference method which will be used as reference point to evaluate the efficiency of the different formulations. In order to be representative of the papers relative to LVD in literature, we have chosen to compare the different formulations with respect to the MDF method. Indeed, as detailed in Chapter 4, the most used MDO method in LVD appears to be the MDF method. In fact, in Table 4.1, about three quarters of the surveyed papers involve a MDF method to solve the LVD problem.
In the first section of this chapter, we compare analytically the characteristics of the different formulations with respect to the MDF method. Then, we compare numerically the different formulations using the same optimization algorithms.

7.2 Analysis of the problem dimensionality and the number of constraints

The different proposed formulations do not handle the same numbers of optimization variables and constraints at the system level. This characteristic has directly an impact on the dimension of the search space and the choice of the optimization algorithms to use. In this section, we study analytically the dimension of the search spaces and the number of constraints with respect to some characteristics of the launch vehicle design problem in order to compare theoretically the different formulations and to expose their advantages and the drawbacks. The numbers in parentheses are the current values taken by the variables for the considered study case (*cf* Chapter 6).
Let us consider :

- n_e : number of stages (3),
- n_x : dimension of the state vector (5 : altitude, velocity, flight path angle, longitude and mass),
- n_c : dimension of the control law variables vector (1 : pitch angle),
- n_{pp} : number of control law discretization points per stage (4),
- n_{psh} : number of design variables per stage which play a part in the flight phases of several stages (1 : diameter of the stage),
- n_{pk} : number of design variables per stage which only play a part in the kth stage flight phase (5 : mixture ratio, chamber pressure, nozzle exit pressure, propellant mass and initial thrust to weight ratio).

The search space dimensions are given by the following relationships:

$$\begin{aligned}
n_{MDF} &= n_e.(n_{pk} + n_c.n_{pp} + n_{psh}) + 1 && (7.1)\\
n_{first\ formulation} &= (n_e - 1).(n_x - 2) + (n_e - 1).n_c + (n_e - 1).n_{psh} + 1 + (n_e - 1) && (7.2)\\
n_{second\ formulation} &= (n_e - 1).(n_x - 2) + (n_e - 1).n_c + (n_e - 1).n_{psh} + 1 + n_e && (7.3)\\
n_{third\ formulation} &= (n_e - 1).(n_x - 2) + (n_e - 1).n_c + (n_e - 1).n_{psh} + 1 + 0 && (7.4)\\
n_{fourth\ formulation} &= (n_e - 1).(n_x - 2) + (n_e - 1).n_c + n_e.n_{psh} + n_e.n_{pk} + 1 && (7.5)
\end{aligned}$$

In the same manner, the numbers of constraints at the system level are given by :

$$\begin{aligned}
n_{c_{MDF}} &= 3.n_e + (n_x - 2) + 2 && (7.6)\\
n_{c_{first\ formulation}} &= (n_e - 1) && (7.7)\\
n_{c_{second\ formulation}} &= n_e && (7.8)\\
n_{c_{third\ formulation}} &= 0 && (7.9)\\
n_{fourth\ formulation} &= n_e + 2.n_e && (7.10)
\end{aligned}$$

By decomposing the optimization problem, the search domains of each optimization process at the system and subsystem levels are considerably reduced in comparison with

MDF which involves a single optimizer to handle all the optimization variables. Indeed, in the MDF method, the dimension of the optimizer search domain concerning the current application case is 31, while the system level optimizers of the first, second, third and fourth stage-wise decomposition formulations have to handle respectively 13, 14, 11 and 27 variables. In the same manner, the MDF optimizer problem is very constrained (8 inequality and 6 equality constraints). The stage-wise decomposition formulations allow the distribution of the constraints on the system and the subsystem levels. Table 7.1 summarizes the number of constraints and variables handled by the MDF and the stage-wise decomposition formulation system level optimizers. The values in green (respectively in red) represent the best (respectively the worst) values.

	MDF	**F1**	**F2**	**F3**	**F4**
Number of variables for the three-stage configuration	31	13	14	**11**	27
Number of additional variables when adding a stage	10	6	6	**5**	9
Number of constraints for the three-stage configuration	14	4	5	**0**	9
Number of additional constraints when adding a stage	3	1	1	**0**	3

Table 7.1: Synthesis of the optimization problems complexity at the system level

The evolutions of the search domain dimensions and the numbers of constraints with respect to the number of stages are given in Figure 7.1. This figure points that the more stages the optimization problem involves, the better are the stage-wise decomposition formulations with respect to the MDF method. Globally, the third formulation is the one which presents the best characteristics. Indeed, this method does not require any constraint at the system level, involves the lowest number of coupling variables (*i.e.* the lowest search domain dimension) and ensures the coupling consistency between the different subsystems at each iteration. This characteristic allows the user to have consistent solutions even if the algorithm at the system level has not converged yet, which is not the case with using the first, second and fourth formulations.

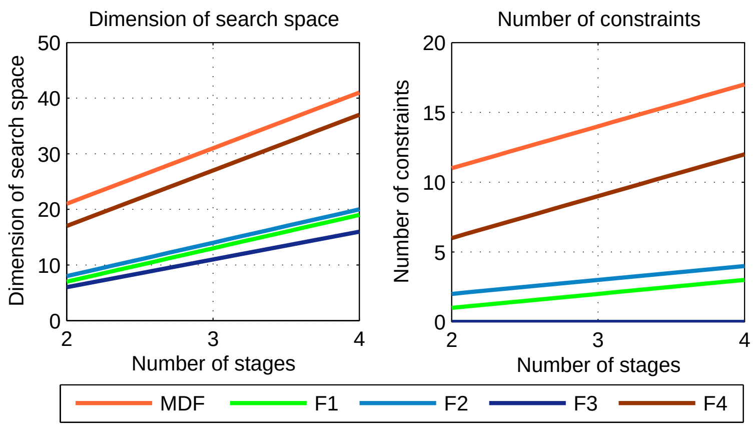


Figure 7.1: Search domain and number of constraints at the system level w.r.t. the number of stages

Table 7.2 details the complete analysis of the problem dimension at the system and subsystem levels (the values in parentheses stand for the first stage optimization which handles the diameter of the first stage as an optimization variable).

	F1	**F2**	**F3**	**F4**
Dimension of the system level optimization problem	13	14	11	27
Number of equality constraints at the system level	2	3	0	3
Number of inequality constraints at the system level	0	0	0	6
Dimension of a subsystem optimization problem	8(9)	8(9)	8(9)	3
Number of inequality constraints for a subsystem	5	5	5	3
Number of equality constraints for a subsystem	3	0	3	0
Total number of optimization variables	38	39	36	36
Total number of constraints	26	18	24	18

Table 7.2: Search space dimension and number of constraints at the system and subsystem levels

7.3 Qualitative comparison of proposed formulations

Table 7.3 qualitatively sums up the different characteristics of the proposed formulations.

	F1	**F2**	**F3**	**F4**
Absence of equality constraints at the system level	-	-	+	--
Absence of equality constraints at the subsystem level	-	+	-	+
Dimension of the coupling vector	-	- -	+	+
Convergence velocity expected	-	$\approx$	+	-
Consistency between the subsystems at each iteration	-	$\approx$	+	$\approx$
Independence of the subsystem optimizations	+	+	-	+

Table 7.3: Qualitative comparison of the different formulations

The six qualitative criteria present is this table are :

- The absence of equality constraints at the system level : this characteristic is very important because it determines the facility to reach the feasible domain and the type of optimization algorithm to use at the system level. From this point of view, the third formulation appears to be the best (no requirement of equality constraints) and the fourth formulation appears to be the worst because it involves both the coupling and design inequality constraints at the system level. The second and third formulations only involves the coupling constraints at the system level.
- The absence of (state) equality constraints at the subsystem level : this characteristic is important in order to quantify the ability of the different subsystems to converge

from any instructions prescribed by the system level. In this matter, the first and third formulations present the worst characteristics because they involve the equality coupling constraints dealing with the reach of the final flight phase state at the subsystem level. The formulations 2 and 4 allow to avoid these equality constraints, considering the reach of the final state directly in the subsystem objective functions.

- The dimension of the coupling vector : the third and fourth formulations present the lowest dimension of the coupling vector because they do not require the estimations of the stage payload masses. The second formulation presents the largest coupling vector because it includes the estimation of the GLOW in addition to the estimated payload masses.

- The expected convergence velocity : this characteristic is fundamental and depends partially on the search space dimension and equality constraints at the system level. From this point of view, the third formulation expects to have the best convergence velocity because the system level does not have any equality constraint and shows the lowest optimization variable set to handle. Contrariwise, the fourth formulation may pose problems because the system level has to handle the design variables (and corresponding constraints) in addition to the shared and coupling variables.

- The consistency between the subsystems at each iteration : from this point of view, only the third formulation ensures the consistency between the subsystems at each iteration of the system level optimization. Indeed, in this formulation, the only conditions to have a feasible design is the convergence of the different subsystems. For the formulations 2 and 4, due to the fact that the subsystem level objective function minimization and the satisfaction of the system level (coupling) equality go the same way, we may expect to reach the feasible domain rather quickly in the optimization process. Contrariwise, concerning the first formulation, the feasibility of the design depends not only on the subsystem convergence (satisfaction of equality constraints) but also the system level coupling equality constraint satisfaction. Furthermore, in this formulation, the minimization of the subsystem objective functions and the satisfaction of the system level coupling constraints do not go the same way. Consequently, we may expect a rather low convergence velocity of this formulation.

- The independence of the different subsystem optimizations : in this matter, only the third formulation does not allow the parallel optimizations of the different subsystems. Indeed, due to the mass couplings which are handled directly at the subsystem level, this method involves a nonhierarchical decomposition and the optimization of the different stages cannot be performed in parallel.

Globally, the third formulation is the formulation which presents the best qualitative characteristics. The only drawbacks of this formulation are the requirement of the equality constraints at the subsystem level and the impossibility to perform in parallel the different subsystem optimizations.

7.4 Numerical comparison of the formulations

7.4.1 Context

In order to complete the previous theoretical analysis, a numerical comparison of the different formulations has to be performed. In order to be relatively independent of the tunings of the optimizers, we have chosen to perform this comparison in global search study. Indeed, the objective of this comparison is to quantify the ability of the different formulations to find feasible designs and to optimize them in the case of large search space, standard tunings on the optimization algorithms and without any *a priori* knowledge on the feasible solution domain. In that way, we have chosen to perform the comparison using a Genetic Algorithm at the system level. This algorithm is the most used evolutionary algorithm in LVD. Many examples of application of GA in case of LVD studies can be found in literature [Schoonover et al., 2000; Duranté et al., 2004, 2005; Akhtar and Linshu, 2005a; Bairstow et al., 2006; Bayley et al., 2008; Castellini et al., 2008], *etc.* A recent study [Rafique et al., 2010] shows the benefits of the use of GA with respect to other heuristics such as Particle Swarm Optimization (PSO) and Simulated Annealing (SA) in air launched vehicle in terms of quality of the optimum, even if the PSO presents better characteristics in terms of computation time. The problem considered in this study present analog characteristics (three-stage configuration, same objective function, disciplines, modelization of the trajectory, constraints, optimization variables) as the problem we want to solve, even if the launch vehicle considered in [Rafique et al., 2010] does not lift off from the ground. However, the problem solved in this study considers very low space search (in terms of bounds on the optimization variables), which is not the case in our study. A global comparative study of the most common heuristics in case of launch vehicle design can be found in [Castellini and Lavagna, 2009]. This study shows the superiority of the GA in case of non linear constraints and non convex (multi)objective function.

The comparative study is not only focused on the value of the best found design but also on the way to reach it. Indeed, the aim of this study is not to find the best tunings of the GA in order to the reach the best performances of the optimization process but to compare the effects of the problem formulation on the optimization process, and to verify that the third proposed formulation is more appropriate to solve the LVD problem. For this comparison, we suppose that :

- the user does not know an appropriate initialization of the optimization process,
- the user does not have *a priori* knowledge on the search domain (bounds of the optimization variables are very large),
- the user has not an accurate idea of the fine tunings on the GA parameters.

Consequently, we have chosen to perform the optimization in case of random initialization and using standard tunings on the GA. The bounds of the optimization variables have been chosen very large in order to reflect the non *a priori* knowledge of the user concerning the search domain (Table 7.4). The results obtained with using the MDF method and a standard gradient-based algorithm in case of large search space are discussed in Appendix B.

Variables		Lower bounds	Upper bounds
Propellant mass (t)	Mp_1	60	100
	Mp_2	10	40
	Mp_3	5	25
Mixture ratio	Rm_1	3	6
	Rm_2	3	6
	Rm_3	3	6
Thrust to weight ratio	T/W_1	1	3
	T/W_2	0.5	2.5
	T/W_3	0.1	2
Chamber pressure (bar)	Pc_1	80	120
	Pc_2	40	80
	Pc_3	20	60
Nozzle exit pressure (bar)	Pe_1	0.01	1
	Pe_2	0.001	1
	Pe_3	0.0001	1
Diameter (m)	D_1	3	5
	D_2	3	5
	D_3	3	5
Maximal load factor	$n_{f_{max}}$	4	6
Control law 1^{st} stage (°)	θ_{11}	40	60
	θ_{12}	25	45
	θ_{13}	20	40
	θ_{14}	15	35
Control law 2^{nd} stage (°)	θ_{21}	10	30
	θ_{22}	5	25
	θ_{23}	5	25
	θ_{24}	0	20
Control law 3^{rd} stage (°)	θ_{31}	-10	10
	θ_{32}	-15	5
	θ_{33}	-20	0
	θ_{34}	-25	-5
Altitude at stage separations (km)	r_{f1}	30	130
	r_{f2}	150	250
Velocity at stage separations (km/s)	v_{f1}	1	5
	v_{f2}	4	8
Flight path angle at stage separations (°)	γ_{f1}	0	50
	γ_{f2}	-10	40
Pitch angle at stage separations (°)	α_{f1}	-4	4
	α_{f2}	-15	15
Estimated mass (t)	$GLOW$	130	170
	Mu_1	40	60
	Mu_2	10	25

Table 7.4: Bounds of the optimization variables

7.4.2 Adaptation of stage-wise formulations to genetic algorithm

The standard GA is not able to formally handle the different involved constraints. In order to use the GA as the system level optimization algorithm, the proposed formulations have been modified in order to handle the constraints as penalization on the objective function [Deb, 1995; Bäck et al., 1997; Deb, 2000]. Many other methods to handle the constraints in GA (*e.g.* "repair methods" [Orvosh and Davis, 1993; Le Riche and Haftka, 1994]) can be found in literature. Different techniques to handle the constraints in GA are exposed in [Richardson et al., 1989; Michalewicz et al., 1996; Coello, 2002]. Moreover, a statistical analysis of different penalty function methods can be found in [Kuri-Morales and Gutierrez-Garcia, 2002].

As we want to only compare the effect of the formulation on the efficiency of the MDO process, we have chosen to handle the constraints by penalizing the objective function in order to be consistent with the papers found in literature which use the GA to solve the LVD problem [Duranté et al., 2004, 2005; Akhtar and Linshu, 2005a, 2006; Delattre and Mongrard, 2007; Geethaikrishnan et al., 2008; Collange et al., 2009; Castellini and Lavagna, 2009; Lemaire et al., 2010]. The weights applied to the penalized objective function can be fixed or variable in function of the number of generations [Akhtar and Linshu, 2005a; Bairstow et al., 2006].
The selected global objective function is consequently a weighted sum of the real objective (GLOW) and the constraint violations. The generic problem reformulation to use a GA as the system level optimizer is the following :

$$\begin{array}{cc} \textbf{Min.} & f(z) \\ \textbf{W.r.t.} & \text{z} \\ \textbf{S.t.} & \begin{array}{c} g(z) \leq 0 \\ h(z) = 0 \\ c(z) = 0 \end{array} \end{array} \longrightarrow \begin{array}{cc} \textbf{Min.} & f(z)+\sum_i \frac{|h_i(z)|}{H_i}+\sum_j \frac{|c_j(z)|}{C_j}+\sum_k \frac{\max(0, g_k(z))}{G_k} \\ \textbf{W.r.t.} & \text{z} \end{array}$$

where H_i, C_j and G_k are normalization coefficients on the equality, coupling and inequality constraints.
The normalization coefficients have been chosen as constants so that constraint violations of 100m on the altitude, 10m/s on the velocity, 0.1° on the angles, 100kg on the mass induce penalizations equivalent to 10 tons on the objective function, which is a relatively high value with respect to the order of magnitude of the real objective function.

In order to be consistent at the two levels of optimization process, the subsystem level objective functions have also been modified as weighted sums of the real objective functions and the satisfaction of the constraints. This modification is necessary in order to evaluate the consistency of the solution when the subsystem level constraints (especially equality constraints) cannot be reached during the subsystem optimizations. Indeed, when the subsystems are not feasible, the weighted sum formulation allows to quantify and transmit the information of no convergence to the GA optimizer which can perform its operations and can evolve its population even if the subsystems have not converged.

In order to compare the efficiency of the proposed formulations, we use the same random initializations, the same optimization algorithm and the same tunings at the system and the subsystem levels. Depending on the different used methods, the objective function can vary according to the number of constraints present in the optimization problem. The control parameters used for the Genetic Algorithm are the followings :

Population size	100
Probability of mutation	0.1
Probability of crossover	0.5
Probability of selection	0.05
Elitism	enabled
Stop criterion	10 hours

Table 7.5: Used GA parameters for the comparison

7.4.3 Results and analysis

In order to compare the different formulations with respect to the MDF method at a fixed calculation time, we take into account the three following criteria :

- the value of the objective function at the optimization process stop,
- the elapsed time to find a first feasible design from a random initialization,
- the improvement of the objective function between the first and the best feasible designs.

Due to the stochastic nature of GA, this study has been perform with 10 random initializations. The chosen stop criterion is calculation time (10 hours). Consequently, the necessary calculation time for the whole comparison is 500hours. Concerning the optimization algorithm, we use the MOGA-II algorithm [Poles, 2003] of the modeFRONTIER platform as the system level optimizer and a SQP algorithm implemented in the $fmincon$ MATLAB function as the subsystem level optimizer. Figure 7.2 shows the results obtained for the different formulations. Figure 7.3 details the result for one case (8^{th} initialization). The results are given in function of the elapsed time and the number of generations.

Quality of the best found design in a given time

One of the most representative characteristics to compare the different formulations is naturally the average best obtained design at the stopping time of the optimization process. For simplicity reasons, we note f^*_{MDF} (respectively f^*_{F1}, f^*_{F2}, f^*_{F3}, f^*_{F4}) the best design found by the MDF (respectively the first, second, third and fourth) formulations, $\bar{f}^*$ the mean value of f^* and σ_{f^*} the corresponding standard deviation.

As expected according to the theoretical analysis, the third formulation is the one which presents the best characteristics in terms of quality of the objective function. Indeed, this formulation induces both the best average value of the optimized GLOW (143.2t) and the minimal standard deviation (1.84t). The third formulation finds the best designs in eight initializations out of ten.

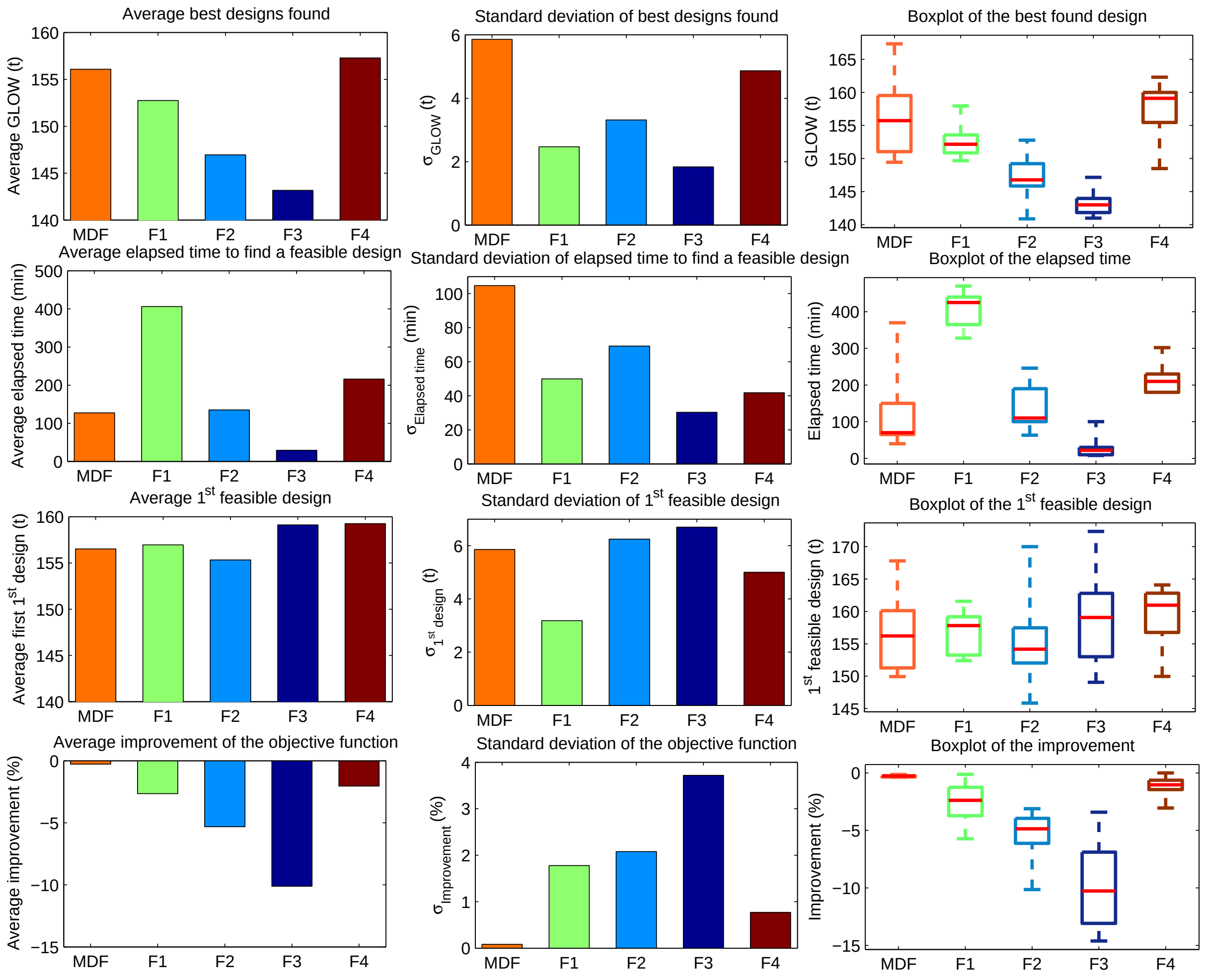


Figure 7.2: Statistical results of the comparison

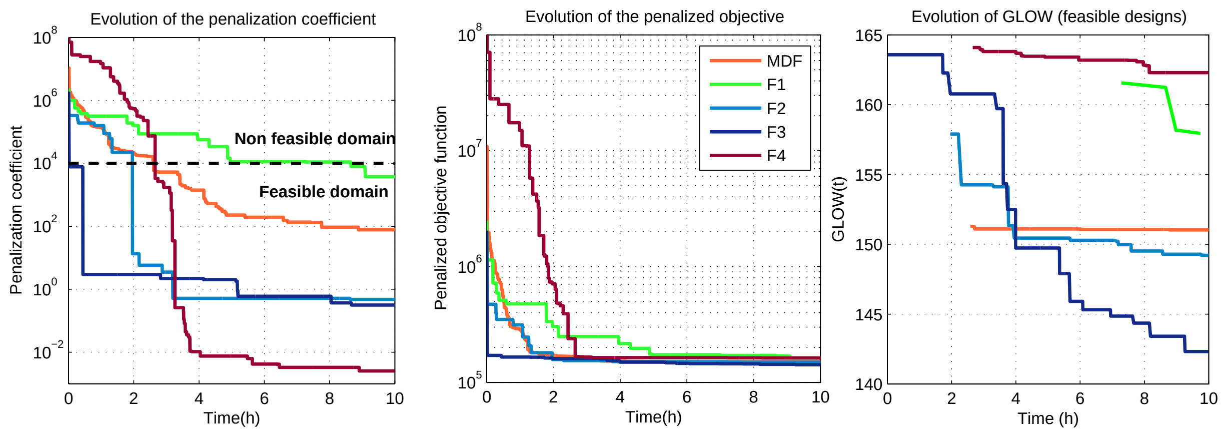
The second formulation finds the best designs in the two other cases. The average value of the optimized GLOW for the second formulation is equal to 147t and its corresponding standard deviation is equal to 3.3t. The three other formulations lead to average best designs above 150t (156t for MDF, 152t for F1 and 157.3t for F4). Globally, the first three formulations allow to obtain better results than the MDF method, both in terms of average value and standard deviation of the objective function (Eq. 7.11,7.12).

$$\overline{f^*_{F3}} < \overline{f^*_{F2}} < \overline{f^*_{F1}} < \overline{f^*_{MDF}} < \overline{f^*_{F4}} \tag{7.11}$$

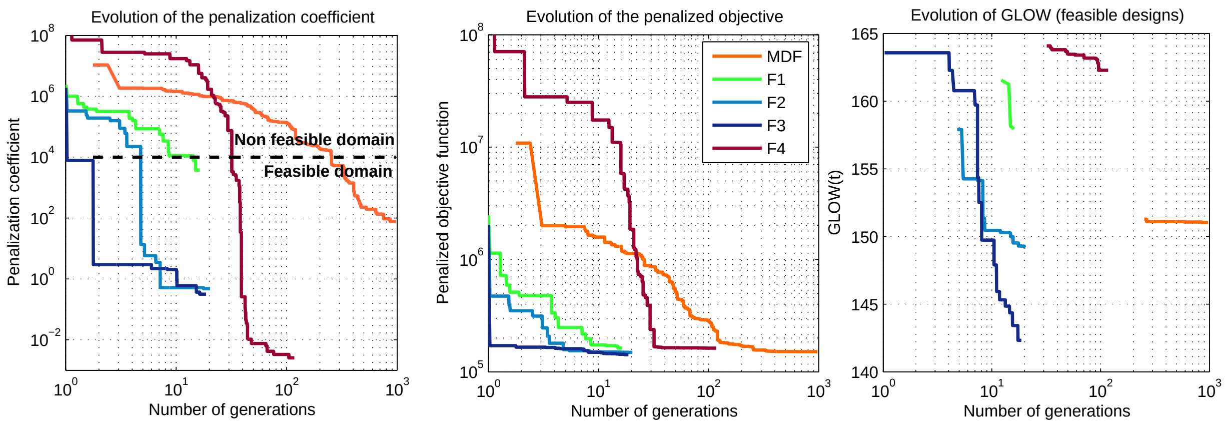
$$\sigma_{f^*_{F3}} < \sigma_{f^*_{F1}} < \sigma_{f^*_{F2}} < \sigma_{f^*_{F4}} < \sigma_{f^*_{MDF}} \tag{7.12}$$

where $\overline{f}$ stands for the average value of f and σ_f its corresponding standard deviation.

The performances of the third formulation can be explained by the fact that this formulation induces the lowest dimension of the search domain at the system level and the absence of coupling constraints at this level, that simplifies the optimizer task. Even if the second formulation involves a greater search domain dimension at the system level than the first formulation, the CO formulation used in F2 allows to satisfy more easily the coupling constraints, that allows the optimizer to perform more easily the optimization of the objective function than F1 for which the optimizer has to satisfy the coupling constraints about the equivalent payload masses of the different stages. The relatively bad results obtained for the MDF and the fourth formulations can be explained by the important number of optimization variables and constraints at the system level that makes global search very difficult and lead to relatively inconsistent results (the dispersions of the designs are the greatest with the MDF and the fourth formulations). This point will be detailed in the paragraph concerning the improvement of the objective function.



(a) Penalization coefficient, penalized and real objective functions w.r.t. elapsed time



(b) Penalization coefficient, penalized and real objective functions w.r.t. number of generations

Figure 7.3: Results obtained for the 8th random initialization

Calculation time required to find a first feasible design

For simplicity reasons we note $t_{0_{MDF}}$ (respectively $t_{0_{F1}}$, $t_{0_{F2}}$, $t_{0_{F3}}$ and $t_{0_{F4}}$) the required computation time to find a first feasible design for the MDF (respectively first, second, third and fourth) formulations. Figure 7.2 shows that the third formulation clearly outperforms the others again in terms of elapsed time to find a feasible design. Indeed, the average elapsed

time for the third formulation is only 30min while the MDF, second, fourth and first formulations require respectively 2h, 2h15, 3h35 and 6h45. Moreover, the third formulation shows the lowest standard deviation, that indicates a relative stability of the results.
This ranking of the formulations can be partly explained by the number of constraints and the handling of the coupling variables at the system level. Indeed, in the third formulation, the system level optimizer has only to coordinate the different stages with respect to the position and velocity at the stage separations (no mass coupling), all the complexity of the optimization is transferred to the different subsystems. In this matter, the feasibility of the found design is only subject to the convergence of the different subsystems (*i.e.* the penalization terms of the subsystem objective functions are below the fixed tolerances). The standard deviation of $t_{0_{F3}}$ is low comparatively to the other formulations. Indeed, this formulation allows to find a feasible design directly in the initialization in 8 cases out of ten (first generation), that explains both the average value and the standard deviation of $t_{0_{F3}}$.

The average calculation times to find a first feasible design of the MDF and the second formulation are very close. Concerning the MDF method, this result can be explained by the fact that the MDF method does not involve any optimization at the subsystem level. Indeed, for the MDF formulation, the necessary subsystem level analysis time to compute the objective function and the different constraints is very short (about 1 second) comparatively to a stage optimization (which takes between 20sec and 1min). Therefore, the GA can perform many more generations in order to decrease the penalized objective function and so to satisfy the constraints, even if these constraints are numerous and quite difficult to satisfy. Table 7.6 resumes the average number of generations required to find a feasible design. In 2h, the GA used for the MDF formulation is able to perform 230 generations whereas for the second formulation, the GA is only able to perform 4.6 generations in 2h15. This phenomenon can explain the quite good results obtained for the MDF formulation. Nevertheless, the standard deviation of this indicator is very important for this formulation, due to the fact that the time to find a feasible design is highly dependent on the initialization.

Concerning the second formulation, the average value of $t_{0_{F2}}$ can be explained by the fact that in this formulation, the convergence of the different subsystems and the satisfaction of the equality constraints of the system level go the same way. Therefore, even if GA only performs 4.6 generations during two hours, this formulation of the MDO problem allows to find a feasible design in far fewer generations than MDF. The standard deviation of $t_{0_{F2}}$ is better than the MDF formulation but is still relatively important.

The first formulation shows the worst results in terms of elapsed time to find a feasible design (6h45). This value may be understood by the fact that for this formulation, the convergence of the subsystem level and the satisfaction of the system level equality constraints do not go the same way, unlike the second formulation. The system level optimizer assumes alone the responsibility to satisfy the coupling constraints, leading to an important computation time to find a feasible design. The standard deviation of $t_{0_{F1}}$ is a little higher than the ones of the first and fourth formulations but is very low with respect to the average value of $t_{0_{MDF}}$.

Finally, the average necessary computation time to find a feasible design of the fourth method is not as good as expected. Indeed, the fourth formulation requires on average 3h35 to find a feasible design, which is much more than MDF, second and third formulations. This phenomenon is due to the important number of optimization variables and constraints at the system level. Indeed, even if in this formulation the convergence of the subsystems and the satisfaction of the system level coupling constraints go the same way, the system level optimizer has to handle all the design variables (and corresponding constraints) and not only the shared and coupling variables as in the three previous formulations. Consequently, this formulation regroups the drawbacks of having an important search domain dimension at the system level and involving optimization phases at the subsystem level instead of just an analysis phase (and consequently involving a considerable calculation time at each subsystem call).

Concerning the elapsed time to find a feasible design, the comparison of the different formulation is the following :

$$\overline{t_{0_{F3}}} < \overline{t_{0_{MDF}}} \approx \overline{t_{0_{F2}}} < \overline{t_{0_{F4}}} < \overline{t_{0_{F1}}} \tag{7.13}$$

Formulations	MDF	F1	F2	F3	F4
Required generations for feasible design	229.9	11.9	4.6	1.3	36.1

Table 7.6: Average number of generations required to find a feasible design

Improvement of the objective function during the optimization process

The improvement of the objective function during the optimization process is the difference (in percentage and absolute value) between the first found feasible design and the best design obtained at the stopping time of the optimization process. We note I_{MDF} (respectively I_{F1}, I_{F2}, I_{F3} and I_{F4}) the improvement of the MDF (respectively first, second, third and fourth) formulations. Once again, the third formulation is the formulation which leads to the best performance (Fig. 7.4). Indeed, this formulation allows to reduce the objective function to 10.1% whereas F2 (respectively F1, F4 and MDF) improve the objective of 5,3% (respectively 2.65%, 2.04% and 0.28%).

$$I_{F3} > I_{F2} > I_{F1} > I_{F4} > I_{MDF}. \tag{7.14}$$

Remark 7.4.1. *The improvement of the objective function during the optimization depends on the value of the first feasible designs. Indeed, we can imagine that a method A which finds a very good first feasible design will have more difficulty to optimize it in comparison with a method B which finds a very bad first feasible design. For this reason it is important to analyze not only the average improvement but also the average value of the first feasible design.*
The average first feasible designs found by all the methods are relatively close and are contained between 155.3t and 159.3t (Fig. 7.4), that shows that the comparison of the different formulations is globally not impacted by the value of the first feasible design and therefore is

relatively valid. Indeed, the dispersion of the average first feasible designs is relatively small in comparison with the best obtained improvements such as the ones of the second and third formulations.

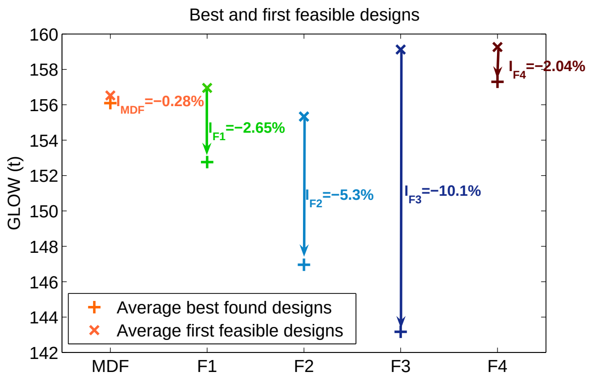


Figure 7.4: Average improvement of the objective function during the optimization

MDF appears as the worst formulation in terms of improvement of the objective function. This phenomenon can be explained by the very high search domain dimension and the fact that the MDF formulation directly involves the satisfaction of the equality constraints on the requirements of the mission at the system level. Consequently, when a design satisfying these constraints is found, the optimizer presents some difficulty to move away in order to find other designs which allow to improve the (real) objective function and without debasing the (effective) penalized objective function. This phenomenon is strengthened by the use of GA as the system level optimizer and the requirement of the penalized objective function. Decomposing the optimization process into the different stages allows to break the perturbation propagation along the whole trajectory and to entrust the different stage optimizers with the task of satisfying the mission specifications.

The same reasoning can be applied for the first proposed formulation. Indeed, the mass coupling constraints are also difficult to satisfy and the system level optimizer of the first formulation presents also some difficulties to move away from the first feasible design in order to minimize the objective function, even if the system level search domain dimension is reduced with respect to the MDF formulation. However, the average improvement of the objective function in case of the first formulation is 10 times the MDF's one. The relatively small average improvement observed for the fourth formulation can be explained in the similar fashion as for MDF by the important number of optimization variables and design constraints at the system level.

On the contrary, concerning the second and third formulations, since the satisfaction of the coupling constraints are partially or totally solved at the subsystem level, the system level optimizer is able to improve the objective function easily, that explains the high reduction of the objective function for these two formulations (especially for the third formulation). For these reasons, it is preferable to use the second or the third formulation in global search study, as we shall see in the following paragraph.

Synthesis

In the synthesis table 7.7 are summed up the different characteristics of the formulation analysis. For that purpose, different grades have been given to the formulations in function of the performance of the considered formulations with regard to the different criteria. A “1” (respectively a “5”) stands for a very good (respectively very bad) behavior of the formulation concerning the considered criterion.

	MDF	F1	F2	F3	F4
Quality of found design	4	3	2	1	5
Time to find a first feasible design	2	5	2	1	4
Improvement of the objective function	5	3	2	1	4
Total	11	11	6	3	13
Global ranking	3	3	2	1	5

Table 7.7: Qualitative synthesis of the comparison

The ranking of the formulations has been performed with respect to the different grades. Globally, this ranking points out that the second and third formulations are the most appropriate to the global search study. Indeed, these two formulations have the best grades in terms of all the considered criteria (Fig. 7.5).

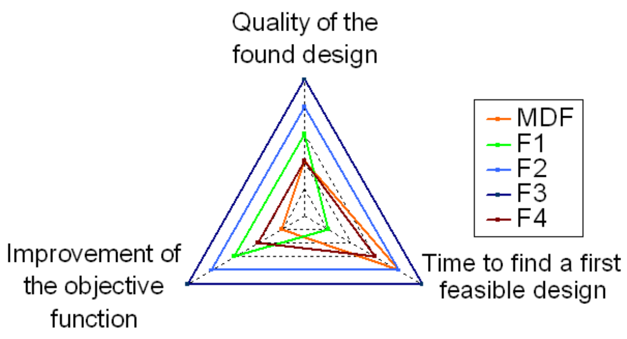


Figure 7.5: Representation of the qualitative comparison with a radar chart

In Figure 7.6 are shown the average values of the best found design and the required time to find a first feasible design for all the considered methods. The third formulation conspicuously presents the best characteristics. Indeed, this formulation leads to the best design, is able to find a first feasible design in a minimum time and leads to the best improvement of the objective function during the optimization, principally due to mass coupling handling at the subsystem level which allows to avoid the equality coupling constraints at the system level and consequently to improve the global search algorithm efficiency. For all of these reasons, we use this formulation in the next part in order to develop an optimization strategy dedicated to solve the LVD problem.

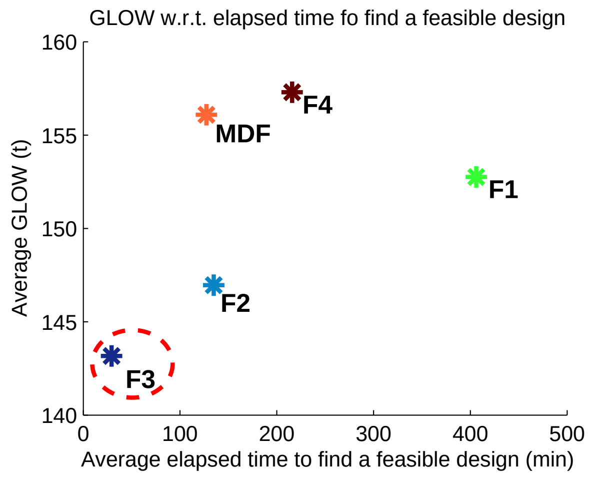


Figure 7.6: Average best found design and elapsed time to find a first feasible design

7.4.4 Limitations of the numerical comparison

The achieved numerical comparison is based on a global search study with the use of GA as the system level optimizer. In order to reduce the stochastic nature of GA, 10 optimizations from 10 random initializations have been performed and the results have been compared. Due to the fact that the necessary time to perform one initialization is 10 hours, we have not been able to perform more than 10 optimizations (the proposed comparison lasted 500 hours). Nevertheless, in order to refine the comparison, it would be helpful to generate more initializations.

Moreover, the GA depends on some tuning parameters which could impact the results (*e.g.* type of GA, probability of mutation, crossover, population size, *etc.*). A complete analysis of the effect of these different parameters was not the first goal of this study and has not been performed due to a lack of time. Indeed, an analysis which will take into account 3 different values of each parameter (*i.e.* 27 possible combinations) would take 13500 hours which is not possible in the framework of this study. For the same reasons, an analysis of the impact of the weighting factors on the penalized cost would be interesting but could not be performed in this study.

7.5 Conclusion

In this part, we have based our reasoning on the launch vehicle trajectory analysis in order to propose four MDO formulations of the LVD problem. These formulations allow to split up the initial MDO problem into the different flight phases and to reduce the initial MDO problem to the coordination of smaller ones.

The different proposed formulations differ in the way they handle the couplings between the different stages. The first proposed formulation considers each stage optimization as an independent problem equivalent to a single stage to orbit launch vehicle MDO problem, for its considered flight phase. In order to coordinate the different stages, this formulation involves at the system level the estimations of the optimized stage masses and the respective coupling constraints. The second formulation applies a collaborative optimization formulation to the stage-wise decomposition. In this formulation, the optimization problems of the different stages are transformed into the minimizations of quadratic sums between the actual subsystem outputs and the system level coupling variables. The system level optimizer does not handle the mass coupling constraints anymore but ensures the convergence (*i.e.* the feasibility) of the different subsystems. This formulation involves one additional optimization variable (and respective equality constraint) at the system level but removes all the equality constraints at the subsystem level. The third proposed formulation is analog to the first formulation but performs sequentially the optimizations of the different stages at the subsystem level. This nonhierarchical formulation allows to avoid the mass coupling variables (and corresponding constraints) at the system level and ensures the consistency of the found designs at each iteration of the optimization process. Finally, the fourth formulation decomposes the global optimization variables set into two subsets corresponding to the design and the control variable sets. This formulation involves at the system level the design variables and coupling variables (and their respective constraints) and at the subsystem level the control variables. As in the second formulation, the different subsystems minimize a quadratic sum of relative errors between the state vector at the stage separations and the coupling variables prescribed by the system level optimizer.

All the different proposed formulations have been compared to the MDF method with respect to the search space dimension and the number of constraints at the different levels of the optimization process. The third formulation appears to be the best formulation because it involves the lowest number of optimization variables, ensures the consistency of the mass coupling during the optimization process and does not require equality constraints at the system level. In order to compare numerically the different formulations, a study case has been developed. The selected problem is the optimization of a three-stage-to-orbit expendable launch vehicle. The selected objective function to minimize is the Gross-Lift-Off-Weight. The different disciplines have been detailed and the problem formulation has been described for the MDF and the proposed formulations. In the last chapter of this part, the different formulations have been numerically compared to MDF in case of a global search study. Three criteria have been selected for the comparison : the ability to find a first feasible design from random initialization, the quality of the best found design and the improvement of the objective function during the optimization process. A standard Genetic Algorithm has been used for the comparison and the different formulations have been modified in order to handle the different constraints as penalization on the objective function. Ten optimizations have been performed from the same random initializations.

Considering all the comparative criteria, the third formulation appears to be the best and allows to significantly improve the global search efficiency with respect to the MDF formulation. Indeed, this method allows to reduce in average by 4 the required computation time to find a first feasible design, finds a lighter design than MDF and allows to efficiently improve the current design all along the optimization process whereas MDF presents some difficulties to improve the objective function. Concerning the hierarchical formulations, the second formulation appears to be the best because it reaches the feasible domain more quickly than the others and leads to the best objective function at the stopping time of the optimization process.

The comparative study has shown that using an appropriate stage-wise decomposition formulation allows to greatly improve the optimization process efficiency, with respect to MDF, which is the most common used method in literature. Nevertheless, even if this formulation associated with a GA allows to find relatively quickly a feasible design and to greatly optimize it, it still presents a relatively low convergence speed. Moreover, this method, involving an optimization phase at the subsystem level may pose some problems when used with a GA which requires a lot of subsystem calls in order to optimize the objective function. For these reasons, it appears as essential to develop a dedicated optimization strategy in order to exploit the stage-wise decomposition specificities directly in the optimization algorithm, in order to have a computationally efficient optimization process. This task is the purpose of the following part. Considering the conclusions of the comparison achieved in this part, we have selected the third formulation to elaborate the algorithmic optimization strategy in Part III.

Part III

Optimization strategies dedicated to the stage-wise decomposition

Chapter 8

Optimization strategies

Contents

8.1 Introduction : needs and context

In the previous part, we have proposed four MDO formulations using the stage-wise decomposition in order to improve the efficiency of the LVD process. A theoretical and numerical comparative study performed in the case of a global optimization has shown that the third formulation appears to be the most promising approach because it allows to find a feasible design from a random initialization in a minimum calculation time, to obtain a best design at the stopping time of the GA and leads to the best improvement of the objective function. Nevertheless, this study has shown the limitations of the GA as system level optimizer. Indeed, it turns out that the GA presents a low convergence speed, even when used with the third formulation. Consequently, we have to propose a dedicated algorithm in order to converge more quickly to the optimum.

The choice of the optimization algorithms is a key point in Launch Vehicle Design. Indeed, both global and local searches are required in order to explore a large search space and to converge efficiently (*i.e.* in a low number of iterations) to an optimum. In this study, we suppose that :

- the user does not have *a priori* knowledge of a feasible solution,
- the user does not know accurately the bounds of the optimization variables,
- the user is not an expert in optimization and does not know the tunings of the optimization algorithms which allow the convergence of the algorithms,
- the optimization process has to be robust to the initialization,
- the optimization process has to be able to produce a result in a limited computation time (a few hours).

The lack of knowledge of the user about the feasible domain and the bounds of the optimization variables requires the use of a global search. Moreover, in addition to this global search, we have to use local search algorithms in order to improve the convergence speed of the optimization process, once a good estimation of the feasible domain has been found. Thus, in this study, we have taken into account the four following requirements to develop the optimization strategy :

1. the ability to quickly find feasible designs (*i.e.* satisfying the design, trajectory and coupling constraints) from random initialization,
2. the robustness to the initialization,
3. the ability to converge from very large search domains (no accurate determination of the optimization variable variation domains are required),
4. once the feasible domain is reached, the ability to efficiently converge (in a limited computation time) toward an optimum.

These requirements are very difficult to reach with an all-in-one algorithm. Indeed, a global (exploratory) algorithm will meet the three first requirements but will present some difficulty to quickly converge toward an optimum. A local search (gradient-based) will meet the fourth requirement but may pose some robustness to initialization problem and will present some difficulties to converge from random (infeasible) initialization.
This part is devoted to the proposition of an optimization strategy dedicated to the stage-wise decomposition. In the first chapter, we propose an optimization process to efficiently obtain an optimum from random initializations and in case of very large search space. In the second chapter, we analyze the numerical results obtained for the three-stage launch vehicle MDO problem for each of the optimization strategy phases and we conclude about the efficiency of the algorithms associated to the third stage-wise decomposition formulation.

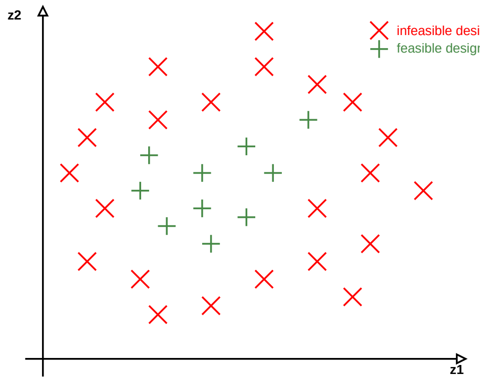
8.2 Description of the proposed optimization strategy

In order to meet all of the requirements expressed in the previous section, we propose a three-phase strategy (Fig. E.7) using the third stage-wise decomposition formulation. The philosophy of the proposed optimization strategy is the following :

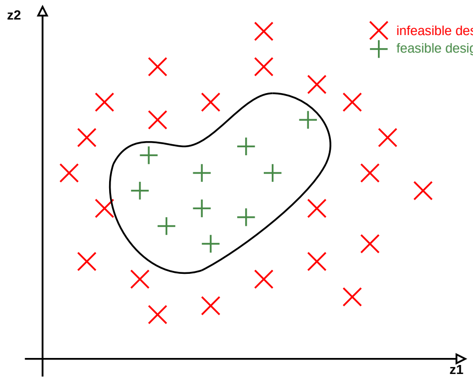
1. Without any *a priori* knowledge on the feasible domain, we explore the system level search domain to find feasible designs in order to provide an appropriate initialization to the optimization algorithm,

2. From these feasible designs, we aim to quickly find solutions in the vicinity of an optimum, in order to reduce the bounds on the optimization variables and consequently to be able to perform a local search,
3. When the search domain is reduced, we use a gradient-based algorithm in order to find an optimum.

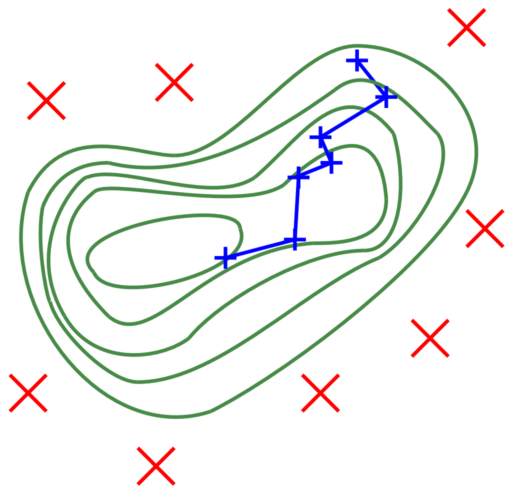
In the first phase of the proposed strategy, we exploit the flight-phase decomposition to perform a sequential exploration of the system level optimization variable set. Once feasible designs are found, a first optimization phase using the Nelder & Mead algorithm (or the Efficient Global Optimization algorithm) is performed in order to reach the vicinity of an optimum. Finally, in order to quickly converge toward an optimum, a gradient-based optimization, using Post-Optimality and Global Sensitivity Synthesis is achieved.



(a) Phase I (a) : exploration of the search space



(b) Phase I (b) : selection of the feasible designs


(c) Phase II : first coarse optimization

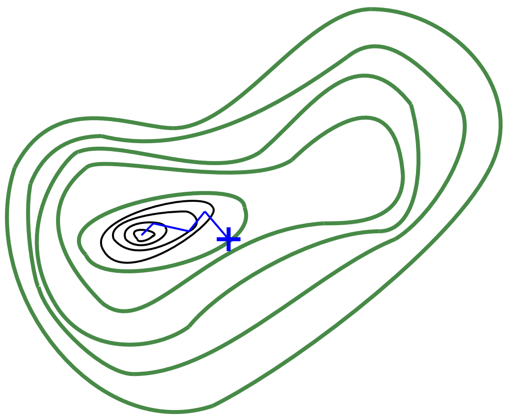

(d) Phase III : second refined optimization

Figure 8.1: Optimization strategy

Remark 8.2.1. *In this section, we only describe the algorithms used at the system level. At the subsystem level, since we have elementary problems to solve (i.e. restricted dimension, lower number of constraints, just one flight phase to simulate, etc.), we use a standard Sequential Quadratic Programming (SQP) algorithm in all the different phases.*

8.3 Phase I : Exploratory phase - initialization process

8.3.1 Exploratory phase : search of a feasible initialization at the system level

The goal of this phase is to quickly find feasible designs in case of very large search space (in terms of bounds on the system level optimization variables) and without any action of the user. This phase aims at finding automatically the initialization for the next phase of the optimization process. Here, as we use the third formulation, the term"feasible designs" means values of system optimization variables which allow the three stage optimizers to converge. The aim of this first phase is the compliance with the requirements 1 and 2.

Instead of performing a simultaneous global randomized search on all system level optimization variables, the proposed exploratory phase consists in using the stage-wise decomposition to split up the system level optimization variable set into elementary subsets and in sequentially performing the explorations of the different subset search domains (Fig. 8.2). To this end, the system level optimization variable set is subdivided into elementary subsets which correspond to the variables intervening in each of the different flight phases. Concerning the three-stage-to-orbit launch vehicle, the system level optimization variable set can be divided into two elementary subsets : $z_{sh_2} \cup y_{12}$ (variables involved in the first and second stage optimizations) and $z_{sh_3} \cup y_{23}$ (variables involved in the third stage optimization) with:

$$z_{sh_2} = \{D_2\} \tag{8.1}$$

$$z_{sh_3} = \{D_3\} \tag{8.2}$$

$$y_{12} = \{\widehat{r_{f_1}}, \widehat{v_{f_1}}, \widehat{\gamma_{f_1}}, \widehat{\alpha_{f_1}}\} \tag{8.3}$$

$$y_{23} = \{\widehat{r_{f_2}}, \widehat{v_{f_2}}, \widehat{\gamma_{f_2}}, \widehat{\alpha_{f_2}}, \widehat{n_{f_{max}}}\} \tag{8.4}$$

The principle of the exploratory phase is the following :

1. explore the optimization variables intervening in the flight phase of the third stage (z_{sh_3} and y_{23}), and select the ones ($z^\circ_{sh_3}$ and y°_{23}) for which the third stage optimizer converges,

2. given $z^\circ_{sh_3}$, y°_{23} and f°_3 (optimized mass of the third stage corresponding to $z^\circ_{sh_3}$ and y°_{23}), explore the optimization variables intervening in the flight phases of the first and second stages (z_{sh_2} and y_{12}) and select the ones ($z^\circ_{sh_2}$ and y°_{12}) for which the first and second stage optimizers converge.

Notations : We note z° (respectively y°) the values of the feasible optimization variables.

Thus, the feasible system level optimization variable sets are given by :

$$z_{sh}^{\circ} = z_{sh_2}^{\circ} \cup z_{sh_3}^{\circ} \tag{8.5}$$

$$y^{\circ} = y_{12}^{\circ} \cup y_{23}^{\circ} \tag{8.6}$$

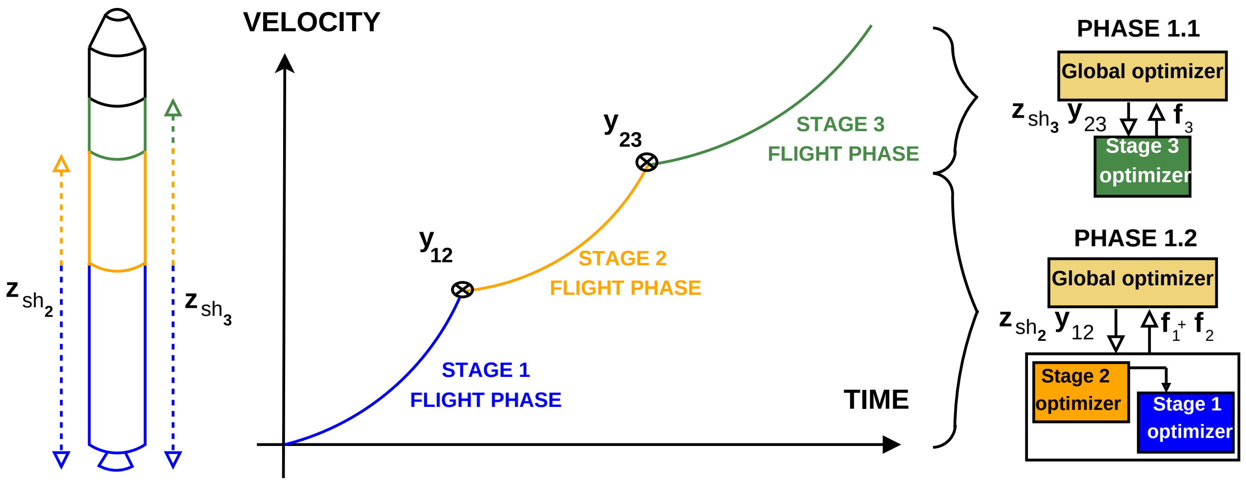


Figure 8.2: Exploration strategy

By using this method, we expect to reduce the computation cost with respect to the commonly used method (exploration of all the variables simultaneously), because the global exploration is reduced to a series of elementary explorations and the process ensures the consistency of the stage configurations between each exploration. Figure 8.2 shows the coupling handling between the different flight phases according to one component of the state vector (here the velocity).

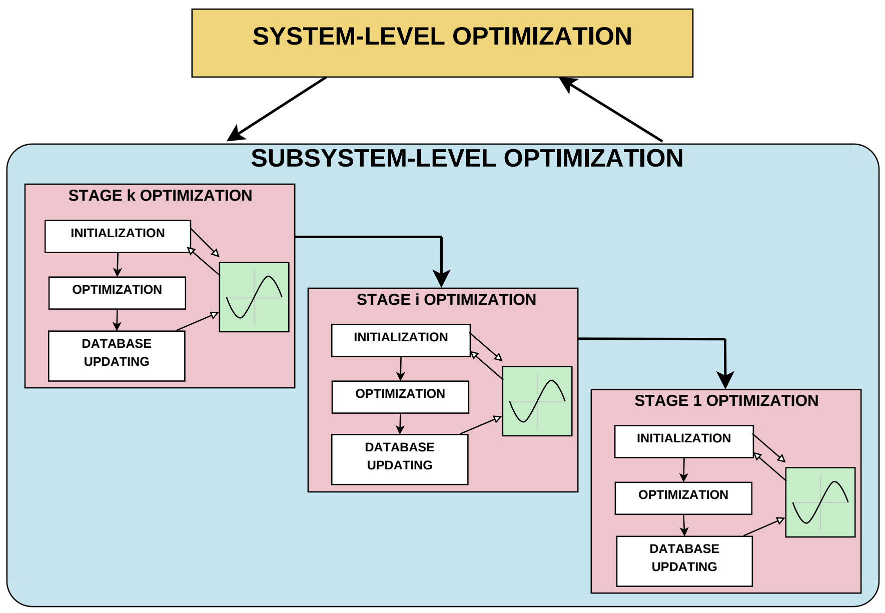
8.3.2 Initialization process at the subsystem level

The initialization problem is fundamental at the subsystem level. Indeed, from any instruction given by the system level, we have to be able to find an initialization which allows the subsystems to converge. This task is very complicated because the search domain at the system level is very important (large bounds) and some combinations of the system level instructions may be infeasible whatever the initialization at the subsystem level. In order to improve the robustness of the subsystem level with respect to the system level instructions z_{sh} and y (compliance with the requirement 3), an initialization process based on automatic database generation and interpolation has been developed.
To this end, each subsystem constitutes its own database composed of subsystem optimized variables in function of the corresponding system level instructions. For each system level iteration, the subsystem calls the initialization process in order to find the most appropriate initialization with respect to the variables prescribed by the system level optimizer. Once the subsystem converged, the resulting optimized variables and the corresponding system level instructions are added to the database. In order to improve the robustness of the optimization process, the initialization is calculated as the barycenter of the P database

nearest designs to the given system level instructions. The weights taken into account in the barycentric interpolation are the inverses of the distances to the prescribed system level variables. These weighting coefficients allow to take into account the nearness of each of the P designs into the initialization calculation. For example, for two given designs P_l and P_m, if P_l is nearer that P_m, the initialization corresponding to this design will have a higher influence on the current design initialization computation than the initialization corresponding to P_m. Let $D_j = \{([z^d_{sh_k}, y^d_k], \bar{z}^d_{jk})\}_{k \in K}$ be the current database of the jth subsystem, the initialization process of this subsystem is the following :

1. Given the system level instructions $[z_{sh}, y]$, find in the current subsystem database D_j, the P_j nearest neighbors to the system level parameters : $[z^d_{sh_i}, y^d_i], \quad i = 1, \cdots, P$ using the euclidean distance d

2. Compute the initialization according to the following relationship : $\bar{z}_{j0} = \dfrac{1}{\displaystyle\sum_{i=1}^{P_j} \frac{1}{d_i}} \displaystyle\sum_{i=1}^{P_j} \frac{1}{d_i} \bar{z}^d_{ji}$



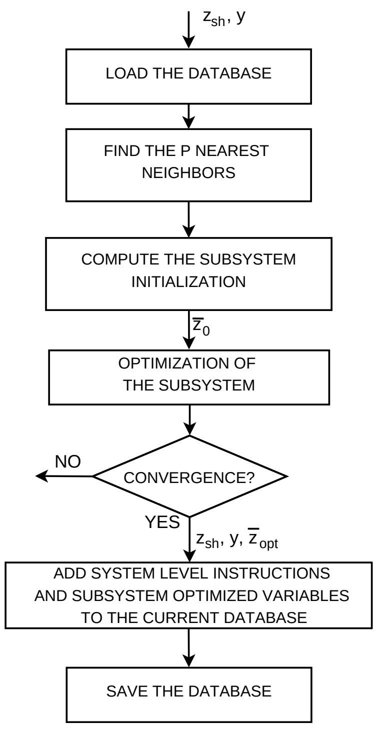

z_{sh}, y
LOAD THE DATABASE
FIND THE P NEAREST NEIGHBORS
COMPUTE THE SUBSYSTEM INITIALIZATION
$\bar{z}_0$
OPTIMIZATION OF THE SUBSYSTEM
NO
CONVERGENCE?
YES
z_{sh}, y, $\bar{z}_{opt}$
ADD SYSTEM LEVEL INSTRUCTIONS AND SUBSYSTEM OPTIMIZED VARIABLES TO THE CURRENT DATABASE
SAVE THE DATABASE

Figure 8.3: Initialization at the subsystem level

8.4 Phase II : first optimization phase

8.4.1 Phase II (a) : first optimization phase using the Nelder & Mead algorithm

From the feasible designs found in the exploration phase, we have to use an optimization algorithm which allows us to quickly find the vicinity of an optimum (compliance with the requirement 3). Indeed, because the found feasible designs are potentially very scattered all over the search space, we cannot directly use a gradient-based algorithm in the current global search space and we have to use first an algorithm which allows to reduce the bounds on the optimization variables and thus makes possible the use of a gradient-based algorithm.

To that purpose, we have chosen the Nelder & Mead algorithm [Nelder and Mead, 1965; Walters et al., 1991]. This algorithm is described in Chapter 3. This algorithm has been chosen because it does not require any tuning and is a very powerful algorithm which allows to quickly reach the vicinity of an optimum. Moreover, this algorithm is not hindered by the subsystem level optimization failures. Furthermore, it does not require any sensitivity of system level optimization variables involving finite differences on the subsystems and consequently requiring as many subsystem reoptimizations as system level optimization variables.

In this section, we suppose that the search space is sufficiently regular to use the Nelder & Mead algorithm. This assumption has been verified *a posteriori* as we shall see in the next chapter, using 10 different initializations for the Nelder-Mead algorithm and comparing the obtained designs at the convergence of the algorithm. If the search-space is not regular and presents multi-minima phenomena, the classical Nelder & Mead algorithm may not converged [Lagarias et al., 1998] and some techniques have been proposed in order to globalize the Nelder & Mead method using probabilistic restarts [Luersen et al., 2004; Luersen and Le Riche, 2004]. Moreover, the Nelder & Mead algorithm can be hybridized with a GA [Durand and Alliot, 1999; Chelouah and Siarry, 2003] in order to exploit the ability of the GA to find a global optimum and to avoid the possible collapse effect observed when Nelder & Mead is used in a highly non linear search space which presents a high number of local minima.

The Nelder & Mead algorithm requires $n + 1$ feasible designs, with n the number of the system level optimization variables. This phase aims at finding a design in the vicinity of an optimum. This design will be used as the initialization for the gradient-based method employed in the third phase.

8.4.2 Phase II (b) : first optimization phase using the Efficient Global Optimization algorithm

In case of non regular search spaces, instead of performing an hybridization of the Nelder & Mead algorithm or an hybridization of this algorithm with a GA, we have chosen to use the Efficient Global Optimization algorithm [Jones et al., 1998; Sasena, 2002] which allows us to compute a limited number of additional points by working on response surfaces of the subsystem level and to perform the optimization on these response surfaces instead of calculating the real objective function which may induce a significant computational cost.

With this technique, we expect to reduce the computation time due to the optimization. Nevertheless, this method requires a higher number of feasible designs found by the first phase in order to build the initial response surface. The number of initial feasible designs recommended in literature is 10 times the dimension [Jones et al., 1998] of the search space ($10.n$) instead of ($n + 2$) for the standard Nelder & Mead method. Nevertheless, in case of non regular search space, this method seems to be very powerful in order to find a design in the vicinity of the optimum. This technique needs the tuning of several parameters in order to present good convergence properties. These parameters quantify the smoothness of the objective function and its sensitivity with respect to the optimization variables. Consequently, this technique implies some tunings from the user in comparison with the Nelder & Mead algorithm, but presents some advantages such as :

- this algorithm is able to converge in large search space,
- this algorithm allows to save computation time because the optimization is performed using response surfaces, which is very convenient in our case because the computation of the objective function (involving the optimizations of the subsystems) takes a significant computation time,
- this method automatically gives an estimation of the approximation error made using the Kriging response surface, which is very convenient in the case of the use of response surfaces in order to quantify the confidence in the obtained solution.

In order to choose which one of the EGO or Nelder & Mead algorithms is the most appropriate in our case, these two algorithms will be numerically tested in the next chapter. If the Nelder & Mead allows to reach efficiently the vicinity of the optimum, this algorithm will be used because it requires much less initialization points (about 10%) than EGO and presents no tunings. If the Nelder & Mead leads to inconsistent results, we will use EGO to reduce the number of subsystem level optimizations during the (system level) optimization process, even if more computations are needed in the phase I.

8.5 Phase III : second optimization phase

The first optimization phase (by the Nelder & Mead algorithm or EGO) aims to find a feasible design in the vicinity of the optimum. From this design, and with reducing the search space at the vicinity of the optimum, the second optimization phase aims to find the real optimum.

8.5.1 Description of the optimization phase

The third stage-wise decomposition formulation combines two levels of optimization. Using a gradient-based at the system level requires the computation of the sensitivities of the objective function with respect to the system level variables. In our case, this computation cannot be performed analytically and consequently have to be achieved with finite differences using the following relationship (case of forward difference):

$$\frac{df}{dz_{0i}} \approx \frac{f(z_{0i}+h)-f(z_{0i})}{h} \tag{8.7}$$

with h a small deviation in one direction. This sensitivity calculation requires at least n optimizations of the subsystem level, each of them requiring as many reoptimizations as the number of stages. For example, concerning the three-stage configuration, each gradient computation requires 11 optimizations per stage (total of 33 optimizations). Consequently, performing the system level sensitivity computation using standard finite differences greatly lessens the efficiency of the optimization process and may be not applicable in our case (calculation time too important).

A possible solution to overcome this difficulty is the use of the Post-Optimality Analysis (POA) [Sobieszczanski-Sobieski et al., 1982] in order to estimate the gradient of the objective function avoiding the reoptimizations of the different subsystems. This technique allows to estimate the gradient of the objective function f from the Lagrange multipliers associated to the subsystem level 1^{st} order Karush-Kuhn-Tucker conditions [Sobieszczanski-Sobieski, 1990].

Because we have a non hierarchical process at the subsystem level (sequential optimizations of the stages), we have to use specific technique to compute the gradient of the objective function. To that purpose, we have chosen to use the Global Sensitivity Equation (GSE) [Sobieszczanski-Sobieski, 1988a] which allows to compute the system level objective function sensitivities in case of bi-level non hierarchical optimization algorithms. This technique has been recently adapted to multi-level (more than two) optimization process [Alyaqout et al., 2005]. Thanks to the Post-Optimality Analysis and the Global Sensitivity Equation, the gradient of the system level objective function can be estimated without requiring any subsystem level optimization, that allows to save computation time and to improve the optimization process efficiency. Let us describe these two techniques applied to the third formulation.

8.5.2 Expression of the system level sensitivities

Global Sensitivity Equation

Let be the system described in Figure 8.4. In this system, y_2 and y_3 stand for the optimized masses of the second and third stages (coupling variables handled at the subsystem level).

Remark 8.5.1. *For convenience sake, in this section, when h is used to represent the subsystem level equality constraints, h stands for the design, trajectory and coupling equality constraints.*

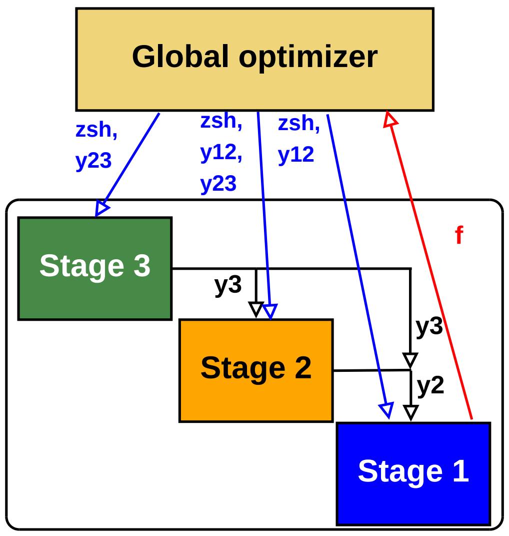


Figure 8.4: Handling of design and coupling variables

The total derivatives of the subsystem level outputs with respect to the system level variables can be written as follows :

$$\frac{dy_3}{dz_{sh}} = \frac{\partial y_3}{\partial z_{sh}} \tag{8.8}$$

$$\frac{dy_3}{dy_{12}} = \frac{\partial y_3}{\partial y_{12}} \tag{8.9}$$

$$\frac{dy_3}{dy_{23}} = \frac{\partial y_3}{\partial y_{23}} \tag{8.10}$$

$$\frac{dy_2}{dz_{sh}} = \frac{\partial y_2}{\partial z_{sh}} + \frac{\partial y_2}{\partial y_3}\frac{dy_3}{dz_{sh}} \tag{8.11}$$

$$\frac{dy_2}{dy_{12}} = \frac{\partial y_2}{\partial y_{12}} + \frac{\partial y_2}{\partial y_3}\frac{dy_3}{dy_{12}} \tag{8.12}$$

$$\frac{dy_2}{dy_{23}} = \frac{\partial y_2}{\partial y_{23}} + \frac{\partial y_2}{\partial y_3}\frac{dy_3}{dy_{23}} \tag{8.13}$$

$$\frac{df}{dz_{sh}} = \frac{\partial f}{\partial z_{sh}} + \frac{\partial f}{\partial y_2}\frac{dy_2}{dz_{sh}} + \frac{\partial f}{\partial y_3}\frac{dy_3}{dz_{sh}} \tag{8.14}$$

$$\frac{df}{dy_{12}} = \frac{\partial f}{\partial y_{12}} + \frac{\partial f}{\partial y_2}\frac{dy_2}{dy_{12}} + \frac{\partial f}{\partial y_3}\frac{dy_3}{dy_{12}} \tag{8.15}$$

$$\frac{df}{dy_{23}} = \frac{\partial f}{\partial y_{23}} + \frac{\partial f}{\partial y_2}\frac{dy_2}{dy_{23}} + \frac{\partial f}{\partial y_3}\frac{dy_3}{dy_{23}} \tag{8.16}$$

This system of equations can be reorganized with using the Global Sensitivity Equation:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{\partial y_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{\partial y_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{\partial y_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 \\ -\frac{\partial f}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} & 0 & 0 & 1 & 0 & 0 \\ 0 & -\frac{\partial f}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} & 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{\partial f}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \frac{dy_3}{dz_{sh}} \\ \frac{dy_3}{dy_{12}} \\ \frac{dy_3}{dy_{23}} \\ \frac{dy_2}{dz_{sh}} \\ \frac{dy_2}{dy_{12}} \\ \frac{dy_2}{dy_{23}} \\ \frac{df}{dz_{sh}} \\ \frac{df}{dy_{12}} \\ \frac{df}{dy_{23}} \end{bmatrix} = \begin{bmatrix} \frac{\partial y_3}{\partial z_{sh}} \\ \frac{\partial y_3}{\partial y_{12}} \\ \frac{\partial y_3}{\partial y_{23}} \\ \frac{\partial y_2}{\partial z_{sh}} \\ \frac{\partial y_2}{\partial y_{12}} \\ \frac{\partial y_2}{\partial y_{23}} \\ \frac{\partial f}{\partial z_{sh}} \\ \frac{\partial f}{\partial y_{12}} \\ \frac{\partial f}{\partial y_{23}} \end{bmatrix} \tag{8.17}$$

This equation has the form $[A].X = b$. The matrix $[A]$ is called the local sensitivity matrix, the vector b is called the local sensitivity vector and the vector X is called the sensitivity derivative vector. This vector contains the total derivatives with respect to the system level optimization variables. Finally, the sensitivity derivative vector can be obtained from the equation (E.71) by the inversion of the local sensitivity matrix ($X = [A]^{-1}.b$) :

$$\begin{bmatrix} \frac{dy_3}{dz_{sh}} \\ \frac{dy_3}{dy_{12}} \\ \frac{dy_3}{dy_{23}} \\ \frac{dy_2}{dz_{sh}} \\ \frac{dy_2}{dy_{12}} \\ \frac{dy_2}{dy_{23}} \\ \frac{df}{dz_{sh}} \\ \frac{df}{dy_{12}} \\ \frac{df}{dy_{23}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{\partial y_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{\partial y_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{\partial y_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 \\ -\frac{\partial f}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} & 0 & 0 & 1 & 0 & 0 \\ 0 & -\frac{\partial f}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} & 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{\partial f}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} & 0 & 0 & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \frac{\partial y_3}{\partial z_{sh}} \\ \frac{\partial y_3}{\partial y_{12}} \\ \frac{\partial y_3}{\partial y_{23}} \\ \frac{\partial y_2}{\partial z_{sh}} \\ \frac{\partial y_2}{\partial y_{12}} \\ \frac{\partial y_2}{\partial y_{23}} \\ \frac{\partial f}{\partial z_{sh}} \\ \frac{\partial f}{\partial y_{12}} \\ \frac{\partial f}{\partial y_{23}} \end{bmatrix} \tag{8.18}$$

The main advantage of the use of the GSE is the possibility to compute separately the stage sensitivities. It also makes possible the parallelization of the different sensitivity calculations. This calculation technique is explained more in details in Appendix C.

Post-Optimality Analysis

The Post-Optimality Analysis [Sobieszczanski-Sobieski et al., 1982] allows to estimate the gradient of the system level objective function avoiding the reoptimizations of the subsystems (which are required by traditional computation of the gradient using finite differences). This technique has been applied in case of Launch Vehicle Design [Braun et al., 1993] and other aerospace problems [Sobieszczanski-Sobieski, 1988b; Chandila et al., 2004; Erb, 2007; Martins and Chittick, 2007; Chittick and Martins, 2008] and allows to enhance the gradient of the objective function at the system level by adding informations obtained at the subsystem level. Let us describe the Post-Optimality Analysis in a simplified case (just one subsystem is considered). Given this assumption, we have at the system level the following optimization problem :

$$\begin{aligned} \textbf{Minimize} \quad & f(z_0, z_1^*) \\ \textbf{With respect to} \quad & z_0 \end{aligned}$$

with z_1^* the subsystem level optimized variables obtained during the optimization of the subsystem. At the subsystem level, the optimization problem can be expressed as follows :

$$\begin{aligned} \textbf{Minimize} \quad & f(z_0, z_1) \\ \textbf{With respect to} \quad & z_1 \\ \textbf{Subject to} \quad & e(z_0, z_1) = 0 \end{aligned} \tag{8.19}$$

For simplification reasons, let us denote by e the equality and saturated inequality constraints at the point (z_0, z_1^*) and ν the corresponding associated Lagrange multipliers vectors. If we suppose that the set of saturated inequalities (included in e) does not change with respect to a small variation of the system level variables, the system level derivatives can be expressed as follows :

$$\frac{df}{dz_0}(z_0, z_1^*) = \frac{\partial f}{\partial z_0}(z_0, z_1^*) + \nu^T \frac{\partial e}{\partial z_0}(z_0, z_1^*) \tag{8.20}$$

We can remark that the equation (8.20) allows to compute the system level derivatives directly from the results obtained after the subsystem level optimization. Indeed, for this calculation, we have just to evaluate the derivative of the constraints with respect to the system level optimization variables, without performing complete reoptimization of the considered subsystems. The different calculations allowing to obtain the equation (8.20) are explained in details in Appendix C.

Coupled use of the Post-Optimality Analysis and the Global Sensitivity Equation

Replacing the derivatives present in the equations (8.8-8.16) in function of their expression obtained by using the Post-Optimality Analysis, we have :

$$\frac{dy_3}{dz_{sh}} = \frac{\partial y_3}{\partial z_{sh}} + \nu_3^T \frac{\partial e_3}{\partial z_{sh}} \tag{8.21}$$

$$\frac{dy_3}{dy_{12}} = \frac{\partial y_3}{\partial y_{12}} + \nu_3^T \frac{\partial e_3}{\partial y_{12}} \tag{8.22}$$

$$\frac{dy_3}{dy_{23}} = \frac{\partial y_3}{\partial y_{23}} + \nu_3^T \frac{\partial e_3}{\partial y_{23}} \tag{8.23}$$

$$\frac{dy_2}{dz_{sh}} = \frac{\partial y_2}{\partial z_{sh}} + \nu_2^T \frac{\partial e_2}{\partial z_{sh}} + \left(\frac{\partial y_2}{\partial y_3} + \nu_2^T \frac{\partial e_2}{\partial y_3}\right) \frac{dy_3}{dz_{sh}} \tag{8.24}$$

$$\frac{dy_2}{dy_{12}} = \frac{\partial y_2}{\partial y_{12}} + \nu_2^T \frac{\partial e_2}{\partial y_{12}} + \left(\frac{\partial y_2}{\partial y_3} + \nu_2^T \frac{\partial e_2}{\partial y_3}\right) \frac{dy_3}{dy_{12}} \tag{8.25}$$

$$\frac{dy_2}{dy_{23}} = \frac{\partial y_2}{\partial y_{23}} + \nu_2^T \frac{\partial e_2}{\partial y_{23}} + \left(\frac{\partial y_2}{\partial y_3} + \nu_2^T \frac{\partial e_2}{\partial y_3}\right) \frac{dy_3}{dy_{23}} \tag{8.26}$$

$$\frac{df}{dz_{sh}} = \frac{\partial f}{\partial z_{sh}} + \nu_1^T \frac{\partial e_1}{\partial z_{sh}} + \left(\frac{\partial f}{\partial y_2} + \nu_1^T \frac{\partial e_1}{\partial y_2}\right) \frac{dy_2}{dz_{sh}} + \left(\frac{\partial f}{\partial y_3} + \nu_1^T \frac{\partial e_1}{\partial y_3}\right) \frac{dy_3}{dz_{sh}} \tag{8.27}$$

$$\frac{df}{dy_{12}} = \frac{\partial f}{\partial y_{12}} + \nu_1^T \frac{\partial e_1}{\partial y_{12}} + \left(\frac{\partial f}{\partial y_2} + \nu_1^T \frac{\partial e_1}{\partial y_2}\right) \frac{dy_2}{dy_{12}} + \left(\frac{\partial f}{\partial y_3} + \nu_1^T \frac{\partial e_1}{\partial y_3}\right) \frac{dy_3}{dy_{12}} \tag{8.28}$$

$$\frac{df}{dy_{23}} = \frac{\partial f}{\partial y_{23}} + \nu_1^T \frac{\partial e_1}{\partial y_{23}} + \left(\frac{\partial f}{\partial y_2} + \nu_1^T \frac{\partial e_1}{\partial y_2}\right) \frac{dy_2}{dy_{23}} + \left(\frac{\partial f}{\partial y_3} + \nu_1^T \frac{\partial e_1}{\partial y_3}\right) \frac{dy_3}{dy_{23}} \tag{8.29}$$

Thus, using the same notations as for the GSE, we have the following matrix equation :

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{\partial y_2}{\partial y_3} - \nu_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{\partial y_2}{\partial y_3} - \nu_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{\partial y_2}{\partial y_3} - \nu_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 \\ -\frac{\partial f}{\partial y_3} - \nu_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \nu_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 & 0 & 0 \\ 0 & -\frac{\partial f}{\partial y_3} - \nu_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \nu_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{\partial f}{\partial y_3} - \nu_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \nu_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{dy_3}{dz_{sh}} \\ \frac{dy_3}{dy_{12}} \\ \frac{dy_3}{dy_{23}} \\ \frac{dy_2}{dz_{sh}} \\ \frac{dy_2}{dy_{12}} \\ \frac{dy_2}{dy_{23}} \\ \frac{df}{dz_{sh}} \\ \frac{df}{dy_{12}} \\ \frac{df}{dy_{23}} \end{bmatrix} = \begin{bmatrix} \frac{\partial y_3}{\partial z_{sh}} + \nu_3^T \frac{\partial e_3}{\partial z_{zh}} \\ \frac{\partial y_3}{\partial y_{12}} + \nu_3^T \frac{\partial e_3}{\partial y_{12}} \\ \frac{\partial y_3}{\partial y_{23}} + \nu_3^T \frac{\partial e_3}{\partial y_{23}} \\ \frac{\partial y_2}{\partial z_{sh}} + \nu_2^T \frac{\partial e_2}{\partial z_{sh}} \\ \frac{\partial y_2}{\partial y_{12}} + \nu_2^T \frac{\partial e_2}{\partial y_{12}} \\ \frac{\partial y_2}{\partial y_{23}} + \nu_2^T \frac{\partial e_2}{\partial y_{23}} \\ \frac{\partial f}{\partial z_{sh}} + \nu_1^T \frac{\partial e_1}{\partial z_{zh}} \\ \frac{\partial f}{\partial y_{12}} + \nu_1^T \frac{\partial e_1}{\partial y_{12}} \\ \frac{\partial f}{\partial y_{23}} + \nu_1^T \frac{\partial e_1}{\partial y_{23}} \end{bmatrix} \tag{8.30}$$

Finally, using the Post-Optimality Analysis and the Global Sensitivity Equation, the system level derivatives vector can be expressed as follows :

$$\begin{bmatrix} \frac{dy_3}{dz_{sh}} \\ \frac{dy_3}{dy_{12}} \\ \frac{dy_3}{dy_{23}} \\ \frac{dy_2}{dz_{sh}} \\ \frac{dy_2}{dy_{12}} \\ \frac{dy_2}{dy_{23}} \\ \frac{df}{dz_{sh}} \\ \frac{df}{dy_{12}} \\ \frac{df}{dy_{23}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{\partial y_2}{\partial y_3} - \nu_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{\partial y_2}{\partial y_3} - \nu_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{\partial y_2}{\partial y_3} - \nu_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 \\ -\frac{\partial f}{\partial y_3} - \nu_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \nu_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 & 0 & 0 \\ 0 & -\frac{\partial f}{\partial y_3} - \nu_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \nu_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{\partial f}{\partial y_3} - \nu_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \nu_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} \frac{\partial y_3}{\partial z_{sh}} + \nu_3^T \frac{\partial e_3}{\partial z_{zh}} \\ \frac{\partial y_3}{\partial y_{12}} + \nu_3^T \frac{\partial e_3}{\partial y_{12}} \\ \frac{\partial y_3}{\partial y_{23}} + \nu_3^T \frac{\partial e_3}{\partial y_{23}} \\ \frac{\partial y_2}{\partial z_{sh}} + \nu_2^T \frac{\partial e_2}{\partial z_{zh}} \\ \frac{\partial y_2}{\partial y_{12}} + \nu_2^T \frac{\partial e_2}{\partial y_{12}} \\ \frac{\partial y_2}{\partial y_{23}} + \nu_2^T \frac{\partial e_2}{\partial y_{23}} \\ \frac{\partial f}{\partial z_{sh}} + \nu_1^T \frac{\partial e_1}{\partial z_{zh}} \\ \frac{\partial f}{\partial y_{12}} + \nu_1^T \frac{\partial e_1}{\partial y_{12}} \\ \frac{\partial f}{\partial y_{23}} + \nu_1^T \frac{\partial e_1}{\partial y_{23}} \end{bmatrix} \quad (8.31)$$

This equation allows us to avoid the subsystem reoptimizations to compute the gradient of the system level of the objective function and consequently allows the efficient use of bi-level gradient-based optimization process in the third phase of the proposed optimization strategy.

8.6 Convergence considerations

In this section, we aim to demonstrate that the third stage-wise decomposition formulation satisfies the same 1^{st} order KKT conditions as the AAO formulation applied to the stage-wise decomposition. In order to perform this demonstration, we proceed as follows :

- firstly, we show the equivalence between the 1^{st} order KKT conditions obtained for the first SWORD formulation and the conditions obtained with AAO,
- secondly, we show the equivalence between the 1^{st} order KKT conditions obtained respectively for the first and the third stage-wise decomposition formulations.

In this section, we only expose the main results of the demonstration. The complete demonstration of the equivalence between the AAO formulation and the third stage-wise decomposition formulation is given in Appendix D.

The AAO formulation of the optimization problem is the following :

Minimize $\quad f(z_{sh}, y, \bar{z}_1, \bar{z}_2, \bar{z}_3) = f_1(z_{sh}, y, \bar{z}_1) + f_2(z_{sh}, y, \bar{z}_2) + f_3(z_{sh}, y, \bar{z}_3)$

With respect to $\quad z_{sh}, y, \bar{z}_1, \bar{z}_2, \bar{z}_3$

Subject to

$$\begin{aligned} h_1(z_{sh}, y, \bar{z}_1) &= 0 && (8.32) \\ h_2(z_{sh}, y, \bar{z}_2) &= 0 && (8.33) \\ h_3(z_{sh}, y, \bar{z}_3) &= 0 && (8.34) \\ g_1(z_{sh}, y, \bar{z}_1) &\leq 0 && (8.35) \\ g_2(z_{sh}, y, \bar{z}_2) &\leq 0 && (8.36) \\ g_3(z_{sh}, y, \bar{z}_3) &\leq 0 && (8.37) \\ c_{01}(z_{sh}, y, \bar{z}_2) &= 0 && (8.38) \\ c_{02}(z_{sh}, y, \bar{z}_3) &= 0 && (8.39) \end{aligned}$$

The constraints c_{01} and c_{02} can be written as follows :

$$c_{01}(z_{sh}, y, \bar{z}_2) = f_2(z_{sh}, y, \bar{z}_2) - y_{m1} \tag{8.40}$$
$$c_{02}(z_{sh}, y, \bar{z}_3) = f_2(z_{sh}, y, \bar{z}_3) - y_{m2} \tag{8.41}$$

with y_{m1} and y_{m2} the two mass components of the coupling vector y.
Globally, the demonstration of the equivalence between AAO and the first formulation amounts to show that these two formulations lead to the same set of 1^{st} order KKT conditions (Eq. 8.42-8.46). Only the condition concerning the Lagrangian is given here. For the complete KKT conditions, one may consult Appendix D.

$$\frac{\partial f_1}{\partial z_{sh}} + \frac{\partial f_2}{\partial z_{sh}} + \frac{\partial f_3}{\partial z_{sh}} + \lambda_1^T \frac{\partial h_1}{\partial z_{sh}} + \lambda_2^T \frac{\partial h_2}{\partial z_{sh}} + \lambda_3^T \frac{\partial h_3}{\partial z_{sh}} + \mu_1^T \frac{\partial g_1}{\partial z_{sh}} + \mu_2^T \frac{\partial g_2}{\partial z_{sh}} + \mu_3^T \frac{\partial g_3}{\partial z_{sh}} + \nu_1 \frac{\partial c_{01}}{\partial z_{sh}} + \nu_2 \frac{\partial c_{02}}{\partial z_{sh}} = 0 \tag{8.42}$$
$$\frac{\partial f_1}{\partial y} + \frac{\partial f_2}{\partial y} + \frac{\partial f_3}{\partial y} + \lambda_1^T \frac{\partial h_1}{\partial y} + \lambda_2^T \frac{\partial h_2}{\partial y} + \Lambda_3^T \frac{\partial h_3}{\partial y} + \mu_1^T \frac{\partial g_1}{\partial y} + \mu_2^T \frac{\partial g_2}{\partial y} + \mu_3^T \frac{\partial g_3}{\partial y} + \nu_1 \frac{\partial c_{01}}{\partial y} + \nu_2 \frac{\partial c_{02}}{\partial y} = 0 \tag{8.43}$$
$$\frac{\partial f_1}{\partial \bar{z}_1} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_1} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_1} = 0 \tag{8.44}$$
$$\frac{\partial f_2}{\partial \bar{z}_2} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_2} = 0 \tag{8.45}$$
$$\frac{\partial f_3}{\partial \bar{z}_3} + \lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_3} = 0 \tag{8.46}$$

The system level equivalence between the first and the third formulations can be shown using the Implicit Function Theorem :

Theorem 8.6.1. ***Implicit Function Theorem[Kudryavtsev, 2001]***
Let be $f : D \to \mathbb{R}, D \in \mathbb{R}^n$. Let us suppose that $\exists\ \alpha = (a_1, \cdots, a_n) \in D$ and $\eta \in \mathbb{R}$ such as $f(\alpha) = \eta$ and $\frac{\partial f}{\partial x_n}(\alpha) \neq 0$,
Then the following hold :

(i) There is a function $g(x_1, \cdots, x_{n-1})$, defined near $(a_1, \cdots, a_{n-1}) \in D \cap (\mathbb{R}^{n-1} \times \{a_n\})$, such that

$$f(x_1, \cdots, x_{n-1}, g(x_1, \cdots, x_{n-1})) = \eta$$

(ii) Near α the given equation has no solutions other than the ones described in (i)

(iii) The derivative of g at $(a_1, \cdots, a_{n-1})$ is given by :

$$g'(a_1, \cdots, a_{n-1}) = \left[\frac{\partial g}{\partial x_1} \cdots \frac{\partial g}{\partial x_{n-1}}\right](a_1, \cdots, a_{n-1}) = \left[-\frac{\frac{\partial f}{\partial x_1}(\alpha)}{\frac{\partial f}{\partial x_n}(\alpha)} \cdots - \frac{\frac{\partial f}{\partial x_{n-1}}(\alpha)}{\frac{\partial f}{\partial x_n}(\alpha)}\right]$$

By applying this theorem to the first stage-wise decomposition formulation, we can show that the optimality conditions obtained for this formulation lead to the necessary optimality condition of the third formulation :

$$\frac{df}{dz_{F3}} = 0 \tag{8.47}$$

Concerning the subsystem-level, since the first and third formulations involve the same optimization problem for each stage, the subsystem level KKT conditions for the first and third formulations are the same. Consequently, we can show the equivalence between AAO, first and third formulations about the 1^{st} order KKT conditions. In other words, the optimum found using the third stage-wise decomposition formulation will also satisfy the optimality conditions of the AAO formulation, and consequently satisfy the general 1^{st} order optimality condition of the MDO problem.

8.7 Conclusion

In this chapter, we have proposed an optimization strategy dedicated to the stage-wise decomposition. This optimization strategy aims to find an optimum in a limited calculation time and without requiring *a priori* informations from the user concerning the feasible domain and the bounds on the optimization variables. This strategy aims to improve the efficiency in terms of calculation time with respect to the global search performed in the previous part with using a Genetic Algorithm. In this chapter, we have chosen optimization algorithms which require the minimal tunings in order to limit as much as possible the intervention of the user during the optimization process.

In order to improve the robustness of the optimization process with respect to the starting points, we have proposed an automatic initialization process which exploits the stage-wise decomposition in order to efficiently find feasible designs. These feasible designs are then used to correctly initialize the optimization phase. The optimization phase is composed of two phases, with a first phase which allows to quickly reach the vicinity of an optimum and a second gradient-based phase which aims to quickly converge to the optimum in a limited search space and initialized in the vicinity of the optimum. We propose to use two algorithms for the first optimization phase. Indeed, we propose to use firstly a Nelder & Mead algorithm which does not require any tuning from the user but may present some difficulties to converge in case of highly non linear search space. If this algorithm is not able to converge or leads to inconsistent results, we propose to use the Efficient Global Optimization algorithm which uses Kriging-based response surfaces in order to improve the efficiency (computation time) of the optimization process but requires additional points to compute the initial response surfaces. In the second optimization phase, we use system sensitivity analysis techniques such as Post-Optimality Analysis and Global Sensitivity Equation in order to avoid the subsystem reoptimizations during the gradient-based search and consequently to save computation time.

In order to validate the different phases of the proposed optimization strategy, a numerical analysis has to be performed in the case of the three-stage launch vehicle MDO problem defined in Chapter 6. This analysis will also allow us to analyze the efficiency of the optimization process (quality of the found optimum, ability to find feasible designs, robustness to the initialization) and to conclude about the performances of the SWORD method and the compliance with the requirements defined in the introduction of this chapter.

Chapter 9

Analysis of the optimization strategy

Contents

9.1 Introduction

This chapter is devoted to the numerical analysis of the optimization strategy proposed in the previous chapter [Balesdent et al., 2011a,b]. Before studying the results obtained for the whole process, we have to validate each of the proposed optimization phases. To this end, we study each of the different phases separately and compare the results obtained for each of them.

Concerning the first phase, we compare the results obtained by the proposed initialization process with respect to the global random search of all the optimization variables taken simultaneously in order to quantify the computational gain obtained with our proposed strategy. Concerning the second phase, Nelder & Mead (and Efficient Global Optimization) are compared to a standard Genetic Algorithm from several different initializations. From the results of this study, we will choose to use either Nelder & Mead or EGO in the global process. Concerning the third phase, we perform 15 different runs from different random feasible initializations in the case of restricted search space and we analyze the results obtained by the proposed bi-level gradient-based algorithm.

After having validated all the different phases, the last section of this chapter concerns the analysis of the whole process. To this end, we generate ten runs of the proposed strategy and we statistically compare the results obtained for the different generated initializations. The results are compared to a reference optimum which has been obtained with fine tunings on the bounds of the optimization variables and the parameters of the optimization algorithm.

9.2 Determination of the reference design

In order to compare the results obtained in the different phases of the proposed optimization strategy, we have to estimate the global optimum design. This optimum has been determined using the third formulation of the SWORD method with a significant initialization work, the tunings on the bounds of the optimization variables and on the parameters of the gradient-based optimization algorithm (we choose a SQP algorithm both at system and subsystem levels). This tuning phase has required a fine tuning adjustments on the search space definition in order to initialize the SQP algorithm in the vicinity of the optimum and to define the bounds on the optimization variables which allow this algorithm to converge. This process has taken several days, and consequently performing the optimization by this way clearly does not meet the optimization process requirements defined in the previous chapter. Nevertheless, this study is necessary to evaluate the efficiency of our proposed strategy (quality of the found optimum).
From the values of the optimized variables corresponding to the found optimum, we have calculated the KKT conditions using the MDF formulation and we have verified that the found optimum effectively satisfies the KKT conditions of the initial problem formulation.
The characteristics of the reference optimum are given in Table 9.1. Moreover, the corresponding optimized trajectory and launch vehicle geometry are given in Figures 9.1 and 9.2.

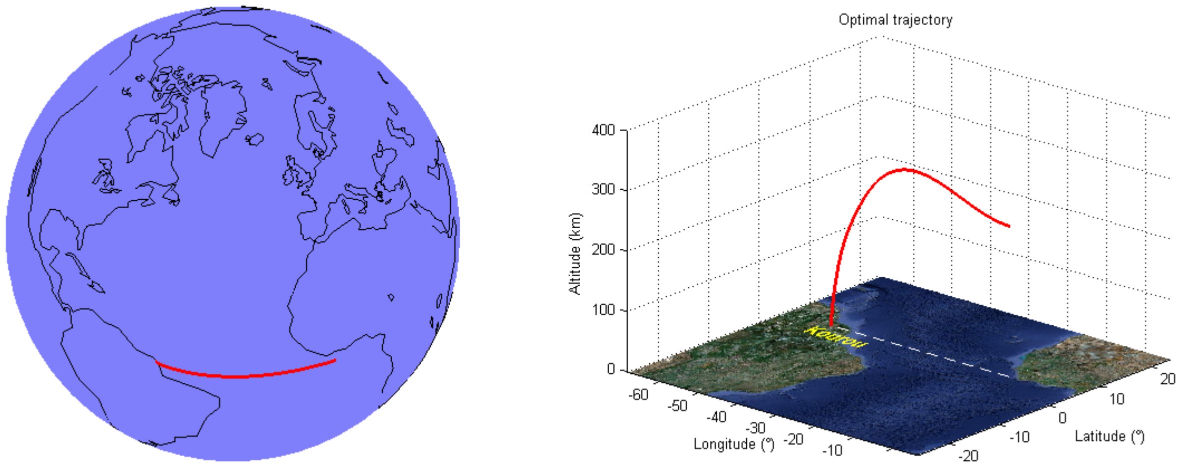


Figure 9.1: Representation of the optimal trajectory

In order to estimate the convexity of the search space around the optimum, some maps have been calculated in the vicinity of the optimum (Fig. 9.3).

Variables		Optimum
Propellant mass (t)	Mp_1	78.3
	Mp_2	26.8
	Mp_3	10.4
Mixture ratio	Rm_1	3.98
	Rm_2	4.11
	Rm_3	4.27
Thrust to weight ratio	T/W_1	1.82
	T/W_2	1.31
	T/W_3	0.49
Chamber pressure (bar)	Pc_1	120
	Pc_2	80
	Pc_3	60
Nozzle exit pressure (bar)	Pe_1	0.42
	Pe_2	$4.91.10^{-2}$
	Pe_3	$5.54.10^{-3}$
Diameter (m)	D_1	3.61
	D_2	3.61
	D_3	3.61
Maximal load factor	$n_{f_{max}}$	4.35
Control law 1st stage (°)	θ_{11}	51.0
	θ_{12}	37.9
	θ_{13}	32.4
	θ_{14}	27.1
Control law 2nd stage (°)	θ_{21}	18.6
	θ_{22}	17.2
	θ_{23}	16.3
	θ_{24}	11.8
Control law 3rd stage (°)	θ_{31}	1.00
	θ_{32}	-6.95
	θ_{33}	-12.8
	θ_{34}	-15.0
Altitude at stage separations (km)	r_{f1}	61.9
	r_{f2}	207.9
Velocity at stage separations (km/s)	v_{f1}	2.47
	v_{f2}	5.72
Flight path angle at stage separations (°)	γ_{f1}	23.7
	γ_{f2}	7.16
Angle of attack at stage separations (°)	α_{f1}	3.42
	α_{f2}	4.68
Objective function	**GLOW (t)**	**136.68**

Table 9.1: Variables for the reference optimum design

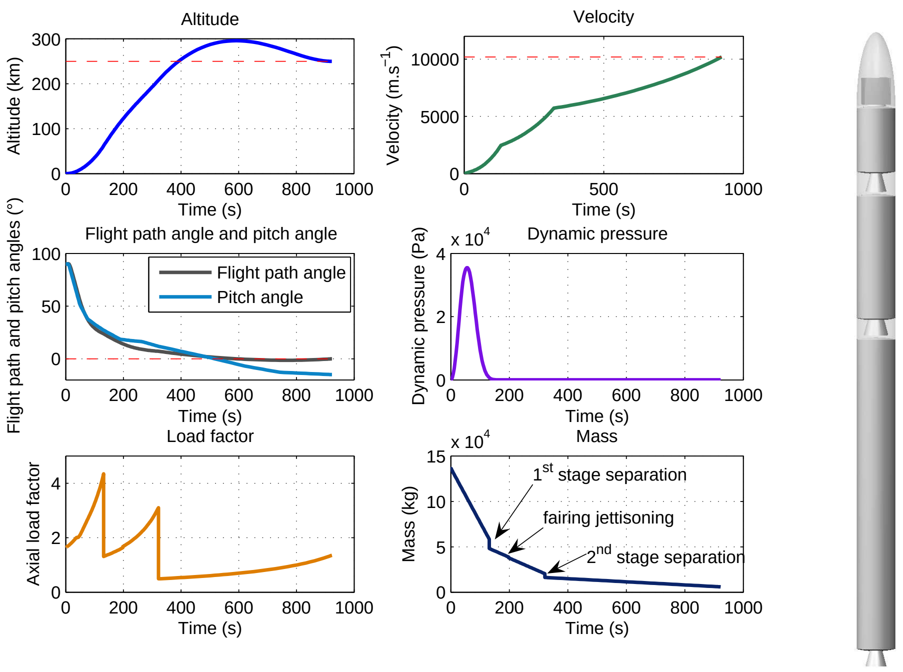


Figure 9.2: Optimal trajectory and geometry

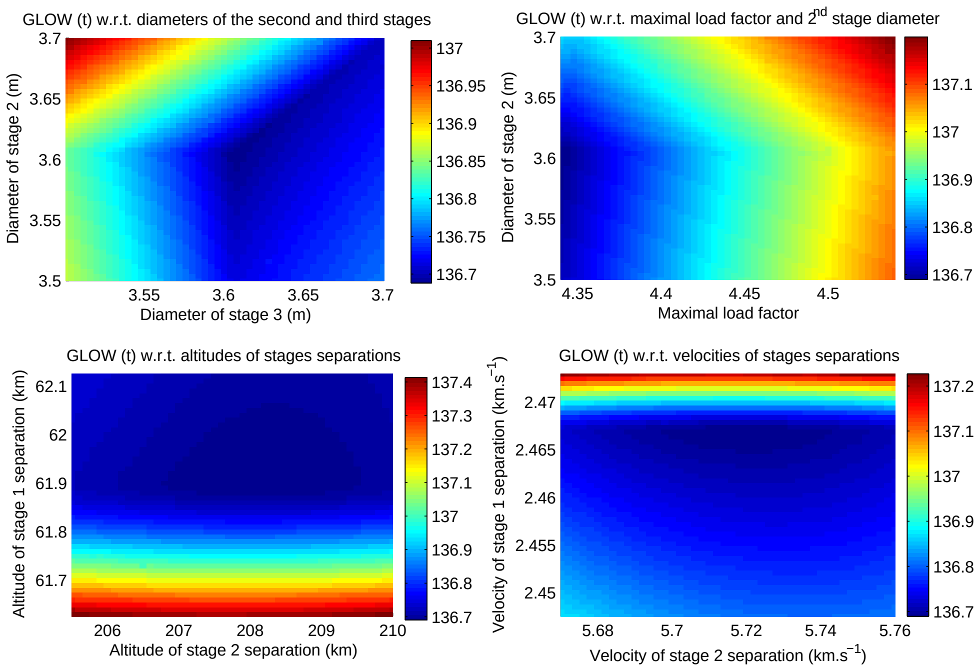


Figure 9.3: Evolution of the objective function around the reference optimum

9.3 Analysis of phase I : initialization strategy

In order to quantify the gain expected by using the elaborated initialization strategies, the proposed initialization process has been compared to the standard random search. To this end, 50000 points have been generated through a randomized process (uniform distribution) in order to statistically compare the methods (Table 9.2). The bounds on the system level optimization variables been chosen as very large (Fig. 9.4) and are the same as for the global optimization performed in Chapter 6.

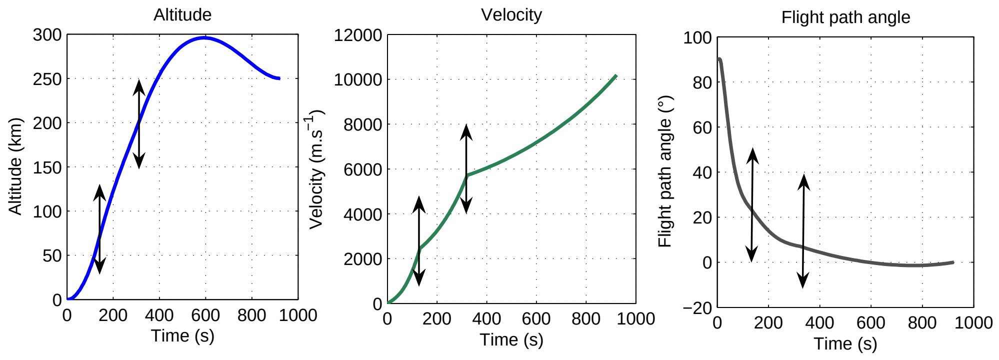


Figure 9.4: Search domain of the position and velocity coupling variables

	Random init.	Proposed init.
Average nb of generated designs required to find a feasible one	4150	244
Proportion of feasible designs in generated population	0.024%	0.41%

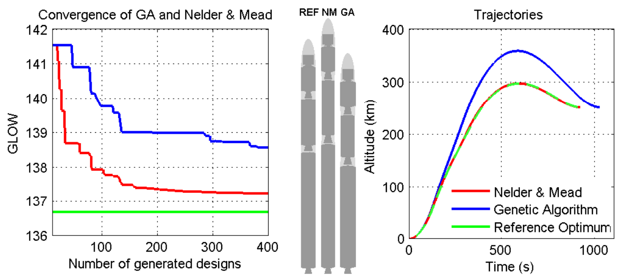
Table 9.2: Comparison between global random and proposed initialization process

Using a randomized search on all the optimization variables taken simultaneously, we have to generate in average 4150 designs to have one feasible design. By using our proposed strategy, we have to generate only 244 designs to obtain a feasible one that represents about 5 minutes of calculation time using MATLAB, 2.4Ghz DualCore Pentium/Windows XP. Consequently, even if the ratio of the feasible designs in the generated population remains quite low (0.41%), the results show that the proposed initialization process allows to significantly reduce (by a factor 17) the computational cost of the initialization. Moreover, since the subsystem databases are increased as the search of feasible designs goes on, the subsystem optimizations can be progressively executed more quickly because initialized nearer and nearer to their optimum.

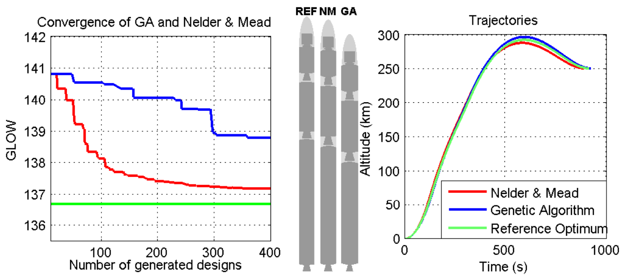
9.4 Analysis of phase II : first optimization phase

9.4.1 Comparison of Nelder & Mead and Genetic Algorithm

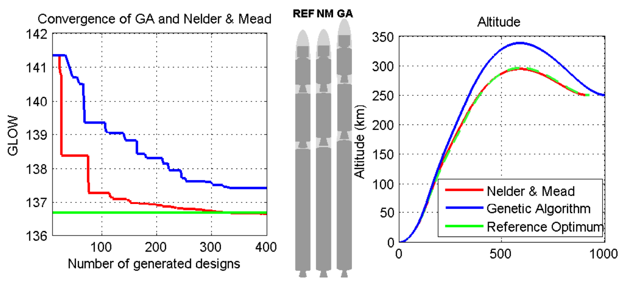
In order to validate the choice of the Nelder & Mead algorithm, we have generated different initializations with the phase I process and for each of them, we have compared the performance using the Nelder & Mead algorithm with respect to a standard GA. For this comparison, we use the same problem formulation and the same initializations. The algorithms have been stopped when 400 designs have been generated. Since the Nelder & Mead requires $(n+1)$ initialization points (where n stands for the dimension of the search space), we have to generate 12 designs using the phase I algorithm. Even if the Nelder & Mead algorithm is more adapted to relatively small search spaces, the comparison is performed in a very large search space, using the same bounds on the optimization variables as in the first phase (Table 7.4). Nevertheless, all the initialization points are feasible designs.
Performing this comparison in a large search space is a good indication to evaluate the ability of the Nelder & Mead algorithm to converge in our case or the necessity of using a more sophisticated global search algorithm (*e.g.* EGO, hybridization of Nelder & Mead and GA, globalized Nelder & Mead, *etc.*).
We have generated four complete initializations of 12 (feasible) designs, each of the initialization searches taking approximately one hour of calculation time. The results obtained by Nelder & Mead and Genetic Algorithm are compared to the reference optimum.



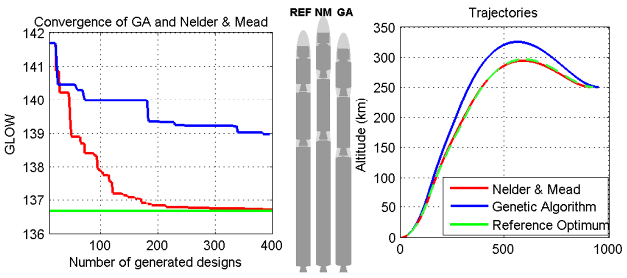
(a) Initialization 1



(b) Initialization 2



(c) Initialization 3



(d) Initialization 4

Figure 9.5: Comparison of Genetic and Nelder & Mead algorithms

Figure 9.5 summarizes the obtained results. For all the considered initializations, we can see that the Nelder & Mead algorithm obtains better results than the Genetic Algorithm. Indeed, the Nelder & Mead algorithm allows to always find a design less than 0,5t more than the reference optimum, while the GA presents difficulties to reduce the gap with respect to the reference optimum to less than 1t. Moreover, the different trajectories found by the Nelder & Mead algorithm are very close to the reference optimal trajectory, which is not the case with GA. Figure 9.5 shows the relative contributions of the geometry and the trajectory to the objective function. For example, in case of the initialization 2, the design found by the Nelder & Mead algorithm is very close to the reference optimal in terms of geometry but the trajectory does not exactly match the global optimal one. The opposite phenomenon is observed for the initialization 1. Concerning the initialization 3, both trajectory and geometry of the design found by the Nelder & Mead algorithm match the global optimum ones, that explains the reaching of the optimum.

The Nelder & Mead algorithm appears to be fairly suitable to the SWORD method because it is able to work from the current large search space (whereas a gradient-based algorithm cannot be applied). Moreover, it does not need sensitivity calculation during the optimization process and allows to quickly find a design in the vicinity of the optimum. As it is shown in Figure 9.5 concerning the first and second initializations, the Nelder & Mead algorithm may not converged to the true optimum. That reinforces the requirement of a gradient-based search in order to refine the obtained design, using the best design found by the Nelder & Mead algorithm as initialization.

The search space of the actual study case appears to be relatively regular to allow the standard Nelder & Mead algorithm to quickly find a design in the vicinity of the optimum. That is why we have chosen to select this algorithm in our global optimization strategy. Nevertheless, for other problems (*e.g.* with a more detailed modelization inducing additional nonlinearities), this algorithm may be inefficient and other global algorithms have to be considered. In the following subsection, we will test the Efficient Global Optimization algorithm which has been selected in the previous chapter.

9.4.2 Analysis of the Efficient Global Optimization algorithm

EGO has been compared in the same search space as used in the previous comparison between Nelder & Mead and Genetic Algorithm (Table 7.4) . In order to use EGO, 110 feasible designs have been generated using the phase I process. This process has taken approximately 10 hours of calculation time. Table 9.3 summarizes the tunings of the EGO algorithm for the current test :

Number of initial points	110
θ_i	10
p_i	1.75
$nuggets$	2.10^{-8}

Table 9.3: EGO parameters

Figure 9.6 shows the results obtained using EGO. We can see that EGO is able to obtain similar results than Nelder & Mead in only 80 iterations (80 calls to the subsystem level

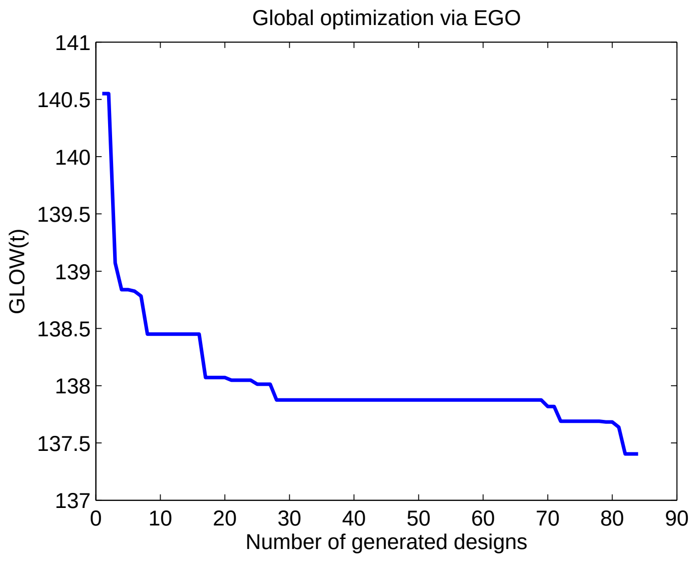


Figure 9.6: Results obtained with EGO

optimizations). Indeed, EGO requires only 20% of the number of iterations involved by the Nelder & Mead algorithm in the previous section but needs more initialization points. Even if EGO allows to obtain interesting results, we have not selected this algorithm in our optimization strategy because :

- it appears that the Nelder & Mead algorithm is sufficient in our study case,

- EGO requires more tunings (tunings on the Kriging model parameters) than the Nelder & Mead algorithm, and consequently appears to be less adapted because of the fact that one of the requirement of the optimization process is to have lesser intervention of the user (the user does not have *a priori* knowledge of the fine tunings on the optimization algorithm),

- this algorithm requires a great number of initial feasible designs in order to build the response surfaces, which induces an important calculation time (110 feasible designs take 10 hours of computation time using the phase I initialization process, which substantially debases the efficiency of the optimization process),

Some studies have been performed with less than 110 points to compute the initial response surface (25 and 55 points) but lead to inconsistent results. Nevertheless, the approximation by Kriging is a cheaper approach which is adapted in our case in order to realize the maps of the objective function around the optimum (Fig. 9.7).

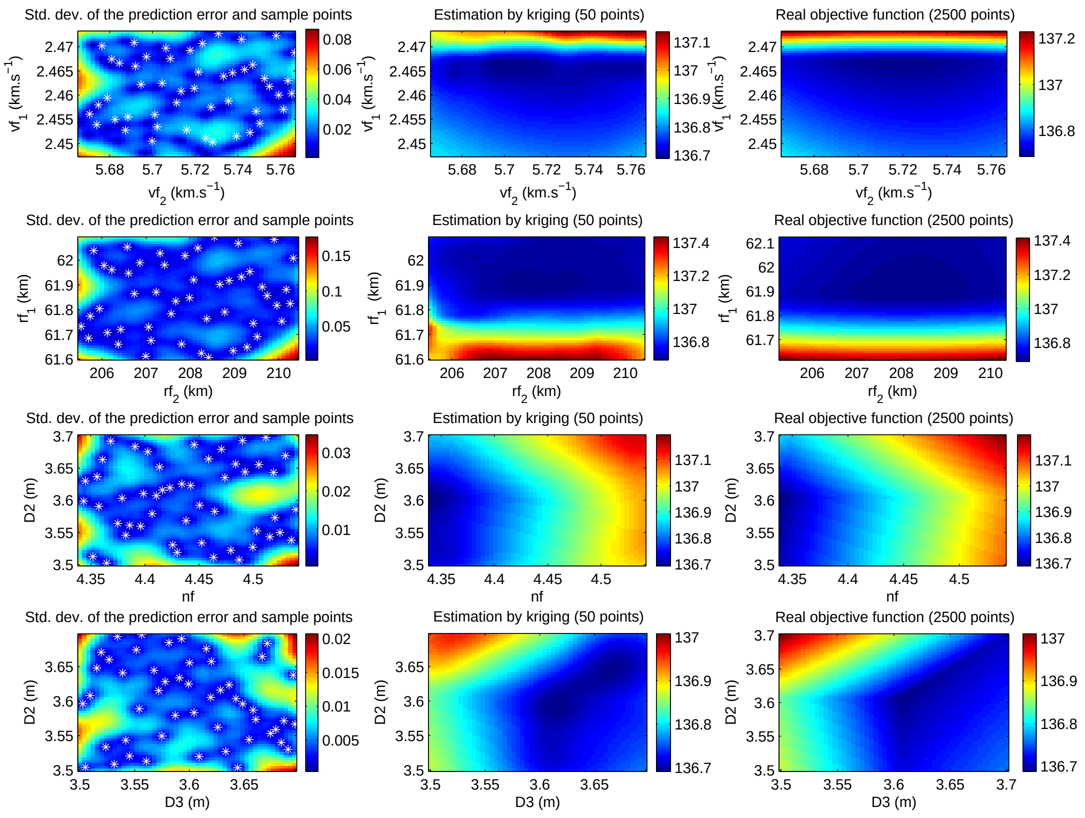


Figure 9.7: Real map (2500 points) and kriged maps (50 points) around the optimum

9.5 Analysis of phase III : second optimization phase

9.5.1 Accuracy of the gradient estimation

In order to validate the gradient estimation, we have computed the sensitivities with Post-Optimality Analysis (POA) and Global Sensitivity Equation (GSE), and using classical finite differences (Table 9.4). The coupled use of POA and GSE allows to divide the computation time of the sensitivity calculation phase by 434, for a maximal error of approximately 1%, which is very satisfactory in our case.

9.5.2 Performance of the proposed gradient-based approach

In order to validate the proposed gradient-based approach, we have generated randomly 15 initializations in the vicinity of the optimum. The optimization has been restricted to a search domain corresponding to 2% of the optimization variable dynamics around the reference optimum. Table 9.5 summarizes the results obtained for each of the initializations. Globally, the proposed algorithm allows to obtain in average a GLOW of 136,7t from an initial GLOW of 137.7t in average.

	Finite Differences (FD)	POA+GSE	Relative difference between FD and POA+GSE
$\frac{\partial f}{\partial D_2}$	$-2.42942.10^2$	$-2.42359.10^2$	0.24%
$\frac{\partial f}{\partial D_3}$	$-1.36954.10^3$	$-1.37620.10^3$	0.48%
$\frac{\partial f}{\partial nf}$	$1.87916.10^3$	$1.89293.10^3$	0.73%
$\frac{\partial f}{\partial rf_2}$	$2.40695.10^2$	$2.42533.10^2$	0.76%
$\frac{\partial f}{\partial vf_2}$	$4.04506.10^3$	$4.06896.10^3$	0.59%
$\frac{\partial f}{\partial \gamma f_2}$	$4.62184.10^2$	$4.66507.10^2$	0.93%
$\frac{\partial f}{\partial \alpha f_2}$	$2.52984.10^2$	$2.52868.10^2$	0.05%
$\frac{\partial f}{\partial rf_1}$	$2.02675.10^2$	$2.04431.10^2$	0.86%
$\frac{\partial f}{\partial vf_1}$	$-1.20073.10^3$	$-1.19255.10^3$	0.69%
$\frac{\partial f}{\partial \gamma f_1}$	$-5.05289.10^2$	$-5.02531.10^2$	0.55%
$\frac{\partial f}{\partial \alpha f_1}$	$-1.06994.10^2$	$-1.07330.10^2$	0.31%
Calculation time (s)	582	1.34	

Table 9.4: Accuracy of the gradient estimation

Indeed, the bi-level gradient-based optimization algorithm allows to find the optimum with an average error of 0.1%, which a satisfactory error rate in the early phase studies because it is lower than the accuracy of our models. Moreover, in 5 of the 15 cases, the algorithm allows to exactly reach the reference optimum. Consequently, the results obtained for this test allow to validate the proposed optimization process.

	Init.(t)	Δ_{init}	Found(t)	Δ_{opt}	Nb iter	Improvement Δ
1	137.90	0.89%	136.81	0.09%	8	-89.61%
2	137.11	0.31%	136.73	0.04%	4	-88.29%
3	137.50	0.60%	136.72	0.03%	12	-95.46%
4	137.89	0.89%	136.76	0.06%	7	-93.55%
5	137.68	0.73%	136.68	0.00%	20	-100%
6	137.19	0.37%	136.82	0.10%	3	-71.79%
7	137.58	0.66%	136.68	0.00%	7	-100%
8	137.85	0.86%	136.88	0.15%	9	-82.75%
9	138.23	1.13%	136.91	0.17%	7	-84.71%
10	138.01	0.97%	136.89	0.15%	8	-84.48%
11	137.10	0.31%	136.68	0.00%	7	-100%
12	138.24	1.14%	136.68	0.00%	14	-100%
13	137.19	0.37%	136.68	0.00%	6	-100%
14	138.81	1.56%	136.84	0.11%	4	-92.62%
15	137.46	0.57%	136.78	0.07%	6	-86.92%
Average	137.72	0.76%	136.77	0.09%	8.3	-91.35%
Std deviation	0.49	0.36%	0.09	0.09%	0.06	8.35%

Table 9.5: Results obtained for the validation of the second optimization phase

In this section, we have analyzed all the different phases of the proposed optimization strategy. This analysis has been performed taking into account the different phases separately. In the next section, we analyze the whole optimization process in order to study if the proposed optimization strategy meets the optimization requirements or not.

9.6 Analysis of the whole optimization process

In order to analyze the whole optimization process, we have chosen to perform 10 optimizations. For that purpose, 10 automatic searches of feasible designs have been performed. Then, from these designs, we have launched 10 times the Nelder & Mead algorithm and finally, from the designs found by this algorithm, we have launched the bi-level gradient-based optimization process. Let us describe the results obtained for the three phases.
Table 9.6 summarizes the results obtained in the phase I. In this table are represented the best design of each initialization (each initialization regroups 12 feasible designs), the number of generated designs required to find 12 feasible designs and the percentage of success. The results present in this table are consistent with respect to the analysis of the phase I exposed previously. Figure 9.8 summarizes all the designs found during the phase I (in terms of relative difference with respect to the reference optimum). All the feasible designs obtained after phase I are represented in the left colored bars. This phase takes approximately one hour.

N° init.	Best design	Generated designs	Success rate
1	140,87t	2820	0.42%
2	145,07t	3145	0.38%
3	141,22t	3532	0.33%
4	140,13t	2472	0.48%
5	140,22t	1678	0.71%
6	143,72t	4417	0.27%
7	139,47t	2318	0.51%
8	140,87t	3274	0.37%
9	142,88t	3533	0.34%
10	142,84t	2668	0.45%

Table 9.6: Results obtained for the phase I

The results obtained during the phase II are summarized in Table 9.7. In this table are indicated the best design found during the phase I, the best design found by the phase II algorithm and the number of expended points during the process. The results obtained in this phase are represented in Figure 9.8 (middle bars). This process takes between one and two hours and achieves the main part of the optimization.

N° init.	Best design of init	Best design found by N&M	Nb of iterations
1	140,87t	137,58t	322
2	145,07t	137,08t	377
3	141,22t	137,04t	200
4	140,13t	137,03t	237
5	140,22t	137,18t	160
6	143,72t	136,86t	261
7	139,47t	137,14t	288
8	140,87t	138,11t	310
9	142,88t	137,46t	260
10	142,84t	137,04t	314

Table 9.7: Results obtained for the phase II

Finally, from the best design found by the phase II process, the gradient-based bi-level optimization process performs the optimization in the estimated vicinity of the optimum. The results obtained during the phase III are given in Table 9.8. In this table are indicated the best designs, the relative differences with respect to the reference optimum and the numbers of system level iterations. The numbers of subsystem level optimizations are given in parenthesis. These numbers are higher than the numbers of iterations because the optimization algorithm at the system level involves a step size calculation which is performed at each iteration and requires a few calculations of the objective function and consequently a few subsystem level optimizations. The results obtained after the third phase are represented as the right bars in Figure 9.8. Table 9.9 summarizes the results obtained by the whole process.

N° init.	Init	Best design found	Diff. vs opt.	Nb of iterations (subsystem opt.)
1	137,58t	136,91t	0.16%	25(98)
2	137,08t	136,89t	0.15%	7(19)
3	137,04t	136,70t	0.01%	11(98)
4	137,03t	136,94t	0.19%	5(75)
5	136,97t	136,90t	0.16%	17(58)
6	136,86t	136,81t	0.10%	8(88)
7	137,14t	136,94t	0.19%	9(94)
8	138,11t	136,94t	0.19%	22(128)
9	137,649t	136,83t	0.11%	8(35)
10	137,04t	136,78t	0.07%	10(98)

Table 9.8: Results obtained for phase III

N° init.	Diff. w.r.t. opt.	Reduction of gap w.r.t. best initial design	Total computation time
1	0.16%	-94.51%	4h07
2	0.15%	-97.50%	3h00
3	0.01%	-99.67%	3h42
4	0.19%	-93.35%	3h10
5	0.16%	-93.72%	3h16
6	0.10%	-98.18%	4h35
7	0.19%	-90.57%	4h30
8	0.19%	-93.87%	4h54
9	0.11%	-97.63%	3h42
10	0.07%	-98.34%	4h16
Average	**0.14%**	**-95.76%**	**3h55**
Standard deviation	**0.06%**	**2.89%**	**39min**

Table 9.9: Summary of the results

Figure 9.8 presents the evolution of the relative difference (in percentage) between the best found design and the reference optimum (which is obtained with fine tunings on the optimizers and design variable variation domains). The different divisions of colored left bars represent the 12 feasible designs obtained after phase I, which are used to initialize the first optimization phase. In all cases, the optimization process converges with a maximal relative gap of 0.2% between the found design and the reference optimum. Indeed, the relative distance between the found design and the reference optimum is reduced by 96% (Fig. 9.8) on average in a calculation time of approximately 4 hours using MATLAB, 2.4GHz DualCore Pentium/Windows XP. Figure 9.9 illustrates the phase III algorithm behavior for the third run. We can see that the algorithm converges to the optimum in 11 iterations (calculation time : approx. 90min).

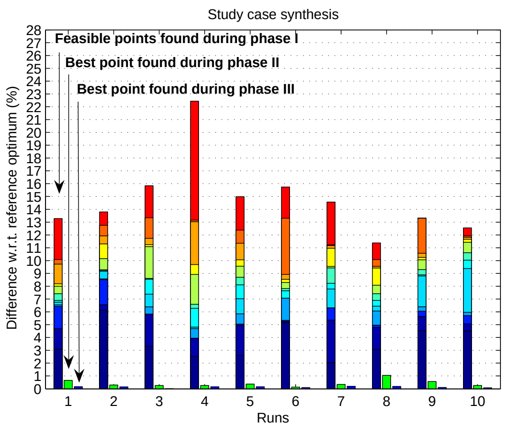


Figure 9.8: Synthesis of the results obtained for the different phases

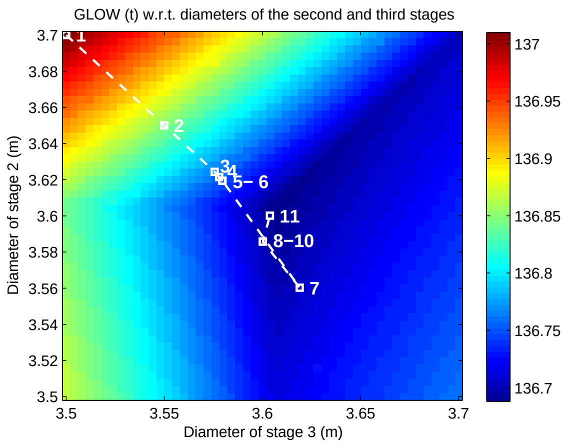


Figure 9.9: Example of phase III algorithm's behavior

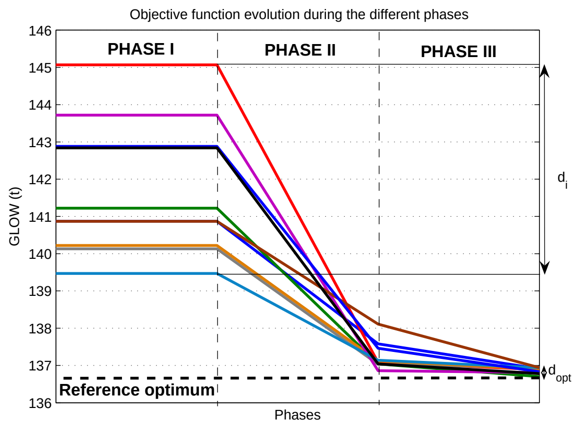


Figure 9.10: Evolution of the objective function during the optimization process

Maximum distance between the different initializations d_i (best designs of phase I)	5.59t
Maximum distance between the found optima d_{opt}	0.25t
d_{opt}/d_i	4.44%

Table 9.10: Dispersion of the found designs

In order to quantify the robustness of the algorithm, defined here as the ability to converge from large search space, we have compared the dispersion of the 10 found designs relatively to the one of the initializations found after phase I (Tab. 9.10 and Fig. 9.10). The results show that the proposed strategy presents satisfactory robustness properties because the dispersion of the found designs is just 4.44% of the initialization one.

Remark 9.6.1. *Even if the EGO algorithm has not been selected for the proposed optimization strategy, we have performed the bi-level gradient-based optimization from the design found by the EGO algorithm (Fig. 9.6). The results indicate that the initialization provided by EGO allows the convergence of the gradient-based algorithm with a relative error of 0.1% to the optimum, which is in the same order of magnitude as the results found when the gradient-based algorithm is initialized by Nelder & Mead (Fig. 9.11).*

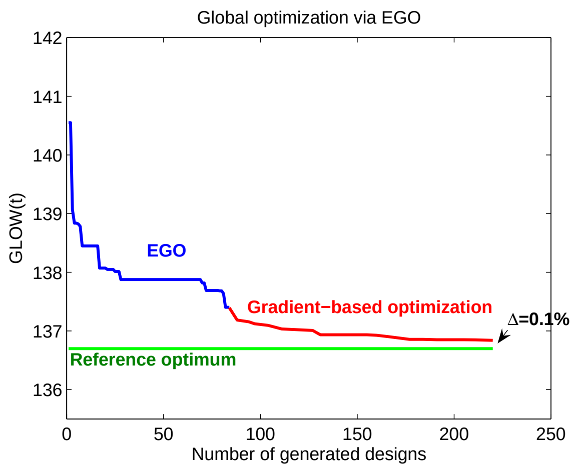


Figure 9.11: Evolution of the objective function during the optimization using EGO and gradient-based algorithm

9.7 Optimization process efficiency

To conclude about the efficiency of the proposed strategy, let us analyze the results obtained by using the proposed optimization process with respect to the optimization requirements expressed in the previous chapter :

1. "ability to quickly find feasible designs from random initialization" :
   The search of feasible designs is performed in a dedicated phase. The feasible designs are computed without any action of the user. The numerical results show that the proposed process is able to find a feasible design in approximately 5 minutes. Consequently, the initialization phase takes approximately one hour, instead of more than 10 hours with a global random search. From this point of view, the proposed process is relatively efficient and allows to meet the requirement.

2. "robustness to the initialization" :
   In order to quantify the robustness of the optimization process, we have performed ten runs of the optimization process. Even if the feasible designs found after the phase I are scattered all along the search space, the results found after phase III are very close. Indeed, the dispersion of the found designs is just 4.44% of the dispersion of the feasible designs. This characteristic is a good indicator to evaluate the robustness of the optimization process and shows that the proposed strategy meets the robustness (stability) requirement fairly well.

3. "ability to converge from very large search domains" :
   One of the principal requirement for our proposed strategy is the ability to work in a very large search space. From this point of view, the proposed algorithm allows to initially work on very large search space and reduces the search space during the optimization process in order to more efficiently converge toward the optimum. Thus, this decomposition of the optimization process in three phases in which each phase is devoted to a particular task allows to meet the large search space convergence ability requirement.

4. "once the feasible domain is reached, the ability to efficiently converge toward an optimum":
   The decomposition of the optimization process into two phases (Nelder & Mead or EGO and bi-level gradient-based algorithms) allows to exploit the advantages of each of the used optimization algorithms and consequently allows to converge with a maximal difference of 0.2% with respect to the reference optimum. This optimization is performed into approximately 3 hours which is quite satisfactory, taking into account the required time by a classical global search as performed in Chapter 7.

9.8 Conclusion

The works presented in this part concern the elaboration of an optimization strategy dedicated to the SWORD method (and more particularly the third formulation of the SWORD method). For this purpose, we have created an optimization process which allows to perform in a limited calculation time (a few hours) the optimization in case of a large search space without *a priori* knowledge on the optimization variable variation domains and with limiting at the most the action of the user. These different requirements have brought us to perform the optimization and exploration tasks separately. Indeed, decomposing these two tasks makes possible the use – or the creation if necessary – of dedicated algorithms for each of them and consequently improves the efficiency of the whole process.

Since the initial search space is very large and the feasible domain cannot be known in advance, we have chosen to develop an exploration strategy based on a series of the explorations of the system level variables relating to the different stages. Thus, this exploratory search allows to save computational time with respect to the exploration of all the system level optimization variables taken simultaneously and therefore appears to be a computationally efficient way to find a consistent (feasible) initialization for the optimization phase.

Moreover, we have chosen to split up the optimization phase into two phases. This allows to improve the flexibility of the optimization process and to leave the user free to use in the first optimization phase a global optimization algorithm if the search space is too highly non linear and presents multiple optima or if the classical Nelder & Mead algorithm, which is used by default, is not able to find consistent results. Once the first phase optimization is achieved and the vicinity of an optimum (expected as global) is reached, we have chosen to use a gradient-based algorithm in which we have replaced the classical finite difference sensitivity calculation by an estimation using the Post-Optimality Analysis, which allows to save a great computational time at the cost of reasonable approximation errors.

Each of the phases of the proposed optimization strategy has been separately tested in order to validate its behavior and a global analysis of the whole proposed optimization strategy has been performed. In order to quantify the robustness of the process and to have a statistical idea of the strategy comportment, we have performed 10 optimizations, from random initializations found by the exploratory algorithm. The results show that the optimization process allows to get closer to less than 0.2% to the reference optimum, in a calculation time less than 4 hours. These results are relatively satisfactory taking into account that this process is intended to be used in early phase studies in which simplified (and therefore not accurate) models are used (the relative difference between the obtained optima and the reference optimum is below the precision of the models).

Conclusion and perspectives

This thesis is focused on the development of MDO methodologies devoted to preliminary Launch Vehicle Design studies. Our contribution is the development of new MDO formulations, called SWORD (Stage-Wise decomposition for Optimal Rocket Design), exploiting a transversal decomposition of the design process according to the different flight phases of the launch vehicle. The SWORD formulations allow to distribute the problem complexity among the system and the subsystem optimization levels and to transform the initial global optimization problem into the coordination of a series of elementary subproblems which can be solved more easily.

From the analysis of the application of the different existing MDO methods to LVD, we have proposed a new formulation of the LVD MDO problem focused on the trajectory optimization and reflecting the physical architecture of launch vehicles. To this end, we have developed four MDO formulations and have compared them with the MDF method in case of global search study. The results of this study have shown that two of the proposed formulations (collaborative and non hierarchical) stand out from all the others and allow to improve the efficiency of the optimization process with respect to the commonly-used method in LVD (MDF), in terms of calculation time to find a feasible design from random initialization and quality of the found design for a given computation time. Therefore, with an appropriate formulation, the stage-wise decomposition makes the global search algorithms more efficient to solve the LVD problem from random initializations and without requiring specific knowledge of the optimization variable variation domains. Nevertheless, this study has shown that the use of a global search alone is not sufficient to obtain an optimum in an acceptable calculation time (a few hours). Therefore, the development of a specific optimization strategy is necessary.

In the second part of this thesis, we have proposed an optimization strategy dedicated to the stage-wise decomposition. To this end, we have used the non hierarchical SWORD formulation in order to propose an optimization strategy which aims to quickly find feasible designs, to converge to an optimum in a limited calculation time, to perform the optimization on a very large search space (in terms of optimization variable variation domains) and to improve the robustness to the initialization. In order to meet these requirements, we have proposed a three-phase optimization strategy.

The first phase is inspired by the non hierarchical SWORD formulation and consists of a sequential exploration of the different system level optimization variable subsets intervening in the different flight phases of the launch vehicle. This method allows to considerably reduce the computation time to find feasible designs with respect to a global random exploration of all the optimization variables at the same time.

From the feasible designs found during the first phase, we have chosen to perform the optimization in two steps. The first step involves a gradient-free optimization using the Nelder & Mead algorithm (or the Efficient Global Optimization algorithm if the search space is highly non linear) in order to quickly reach the vicinity of an optimum. Finally, from the best design found during the first optimization step, the second step consists in performing a gradient-based optimization at the system level, in a limited search space in the vicinity of the optimum. In order to reduce the computation time of this phase, we have replaced the classical finite difference computation of the gradient of the system level objective function by an estimation which does not require any time-costly subsystem re-optimizations. This estimation is performed using dedicated calculation techniques which exploit the convergence informations given by the subsystems (Post Optimality Analysis and Global Sensitivity Equation).

A global analysis of the whole process (formulation and optimization strategy) has then been achieved in order to evaluate the performance of the SWORD method. The results show that the optimization process allows to get closer to less than 0.2% to the reference optimum, in a calculation time less than 4 hours, that is satisfactory for early design studies in comparison with classical approaches which take more than 10 hours when performed in such a large search space. This analysis has validated the proposed optimization strategy and has shown that the use of the stage-wise decomposition is valuable in order to solve in limited calculation time the LVD problem without requiring *a priori* knowledge of the user.

In order to improve the proposed MDO method, some extensions of the works presented in this thesis can be proposed. The described design process is currently devoted to early design studies in which simplified models are used. It would be interesting to test the proposed method with more complicated disciplinary models, involving additional couplings.

Moreover, in this thesis, we have compared the SWORD methods with MDF and not with other MDO methods such as BLISS, ATC, CSSO, *etc.* The comparison of the SWORD methods with these MDO methods would be very interesting but requires to build more detailed disciplinary models in order to be able to apply the MDO methods present in literature.

Additionally, as the SWORD method is principally devoted to early design studies in which simplified models are used, it would be valuable to incorporate the handling of model uncertainties in the current design process in order to ensure the consistency of the design during the following phases of the design process.

Furthermore, the optimal control method involved in this thesis is a direct method. It would be interesting to study in future works other trajectory optimization methods (*e.g.* indirect method) in order to choose the most adapted to the design process.

In addition, the SWORD method has been tested on the design problem of classical expendable launch vehicles with sequential staging (no booster). It would be very helpful to transpose and test the SWORD formulations to more complex launch vehicle design problems such as the launch vehicles with parallel staging or the reusable launch vehicles. In the same manner, the current design process is only able to handle continuous optimization variables. The introduction of discrete variables would be valuable and would allow to consider in the design process other characteristics of launch vehicles such as the number of stages, *etc.*

Finally, in this thesis, we have applied the SWORD method to single objective problems. It would be also interesting to test our method in case of multi objective problems (*e.g.* the maximization of the payload and the minimization of the cost). This study case would allow us to design a family of optimal launch vehicles and to be able to propose several choices of launch vehicles for different missions.

Appendix A

Dry mass sizing module

The dry mass sizing module aims at calculating the dry mass of the launch vehicle in order to compute the objective function and to simulate the trajectory. The dry mass of each stage is the sum of the main launch vehicle components :

- tanks,
- turbopumps,
- pressurant system,
- combustion chamber,
- nozzle,
- engine.

A.1 Tanks

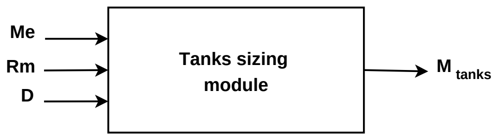


Figure A.1: Tank sizing module

Selected material : Aluminium.
Properties of aluminium :

Density(ρ_{Al})	$2800kg.m^{-3}$
Ultimate strength ($\sigma_{r_{Al}}$)	400 MPa
Yield strength ($\sigma_{e_{Al}}$)	380 MPa
Minimal thickness ($e_{min_{Al}}$)	1.5 mm

Fixed parameters :

- Safety coefficient (C_s) : 1.25
- Margin of safety for the tank's thickness (f_R) : 1.2

Inputs :

- Mp : propellant mass (kg)
- Rm : mixture ratio
- D : stage diameter (m)

Determination of the tank volume:

From Mp and Rm, we determine the propellant volumes by :

$$V_{LOX} = \frac{1}{\rho_{LOX}} \frac{Mp.Rm}{Rm+1} \tag{A.1}$$

$$V_{H2} = \frac{1}{\rho_{H2}} \frac{Mp}{Rm+1} \tag{A.2}$$

with :

- $\rho_{LOX} = 1141\ kg.m^{-3}$
- $\rho_{H2} = 71\ kg.m^{-3}$

Tank pressures : The tank pressures are given by the following expressions [Humble et al., 1995] :

$$P_{r_{LOX}} = [10^{-0.1068.log(V_{LOX})-0.258810^6.}].10^6 \tag{A.3}$$

$$P_{r_{H2}} = [10^{-0.1068.log(V_{H2})-0.258810^6}].10^6 \tag{A.4}$$

Tanks thickness :

$$e_{r_{LOX}} = C_s.\frac{P_{r_{LOX}}.D_R}{2\sigma_{r_{Al}}} \tag{A.5}$$

$$e_{r_{H2}} = C_s.\frac{P_{r_{H2}}.D_R}{2\sigma_{r_{Al}}} \tag{A.6}$$

The considered tanks are two cylinders provided with spherical bulkheads. The radius of the spherical bulkhead is given in function of the tank's diameter by the following formula :

$$R_c = \frac{D_R}{2} \tag{A.7}$$

The volume of the tank is given by :

$$V_R = \pi.\left(\frac{D_R}{2}\right)^2 L_R + \frac{4}{3}\pi\left(\frac{D_R}{2}\right)^3 \tag{A.8}$$

From Equation A.8, we can determine the length (cylinder part) in function of its radius and its volume. Therefore, the surface of the tank is given by :

$$S_R = \pi.D_R.L_R + 4\pi \left(\frac{D_R}{2}\right)^2 \tag{A.9}$$

Thus, the mass of the tank is :

$$M_{tank} = e_R.S_R.f_R.\rho_{Al} \tag{A.10}$$

By replacing L_R by its expression in function of D_R and V_R (so D and Mp from the equations A.1 and A.2) the mass of the tanks can be calculated.

A.2 Turbopumps

Inputs :

- Pc : chamber pressure (Pa)
- q : mass flow rate (kg/s)
- Rm : mixture ratio
- Mp : propellant mass (kg)

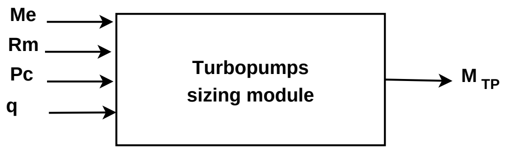


Figure A.2: Turbopumps sizing module

The mass flow rate of LOX and LH2 are given by :

$$q_{LOX} = q\frac{Rm}{1+Rm} \tag{A.11}$$

$$q_{H2} = q\frac{1}{1+Rm} \tag{A.12}$$

From the difference of pressure between the tanks and the combustion chamber, the mass flow rate, the density of the fluid and the efficiency of the pump ($\eta_P = 75\%$), we determine the necessary power of the pump :

$$PW_{LOX/H2} = \frac{q_{LOX/H2}(Pc - Pr_{LOX/H2})}{\rho_{LOX/H2}.\eta_P} \tag{A.13}$$

Then, the mass of the turbopump is given by [Humble et al., 1995] :

$$M_{TP_{LOX/H2}} = 1.5\left(\frac{PW_{LOX/H2}}{2N\pi/60}\right)^{0.6} \tag{A.14}$$

with N the rotational speed of the pump (12400 tr/min).

A.3 Pressurant system

Inputs :

- Mp : propellant mass (kg)
- Rm : mixture ratio

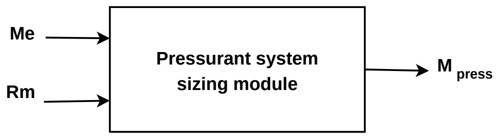


Figure A.3: Pressurant system sizing module

Data :

- Initial gas temperature (T_0) : 300k
- Pressurant gas : helium He
- Initial pressure in the pressurant tank (P_0) : 200 bars
- Final pressure in the pressurant tank (P_f) : $120\% max(P_{r_{LOX}}, P_{r_{H2}})$
- Ratio of specific heats for the pressurant gas (γ_{He}) : 1.67
- Molecular mass of the pressurant gas (M_{He}) : 0.004 kg/mol
- Mass density of the pressurant tank material (ρ_{capa}) : 1570 $kg.m^{-3}$
- Tensile strength of pressurant tank material (σ_r) : 1270 MPa
- Safety coefficient Cs : 2
- Thickness of the liner (density ρ_{liner} : 2800 kg/m3) (e_{liner}) : 1mm

The volume of the pressurant tank is given by :

$$V_0 = \frac{max(P_{r_{LOX}}, P_{r_{H2}})\gamma_{He}.V_R}{P_0 - P_f} \tag{A.15}$$

The mass of the pressurant gas is calculated by :

$$m_{He} = \frac{M_{He}.P_0.V_0}{R.T_0} \tag{A.16}$$

with R=8.314 ISU, the pressurant gas constant.
The inner radius of the pressurant tank is given by :

$$R_{int} = \left(\frac{3.V_0}{4\pi}\right)^{\frac{1}{3}} \tag{A.17}$$

We define the thickness of the tough material by :

$$e_r = \frac{Cs.P_0.R_{int}}{2.\sigma_r} \tag{A.18}$$

The total thickness is :

$$e_{tot} = e_{liner} + e_r \tag{A.19}$$

The outer radius of the pressurant tank is given by :

$$R_{ext} = e_{tot} + R_{int} \tag{A.20}$$

Finally, the mass of the pressurant tank (overestimated by 10%) is :

$$m_c = 1.1\frac{4}{3}\pi(((R_{ext}^3 - (R_{int} + e_{liner})^3)\rho_{capa} + ((R_{int} + e_{liner})^3 - R_{int}^3)\rho_{liner}) \tag{A.21}$$

Then the total mass of the pressurant system is given by :

$$M_{press} = m_{He} + m_c \tag{A.22}$$

A.4 Combustion chamber

Inputs :

- Pc : chamber pressure (Pa)
- Rm : mixture ratio
- q : mass flow rate (kg/s)

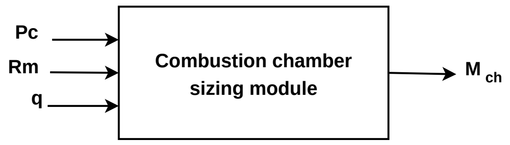


Figure A.4: Combustion chamber sizing module

The isentropic coefficient (γ), the flame temperature (T_c), and the molecular mass of combustion (M) are interpolated from classical curves found in [Humble et al., 1995] (fig. A.5) The cross-sectional area of nozzle throat is given by :

$$A_c = \frac{q\sqrt{\gamma.R.T_c}}{\eta_c.Pc.\gamma\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{2\gamma-2}}} \tag{A.23}$$

with η_c the combustion efficiency (0.98), R the gas constant $\frac{8314}{M}$.
The throat's diameter is :

$$D_c = 2\sqrt{\frac{A_c}{\pi}} \tag{A.24}$$

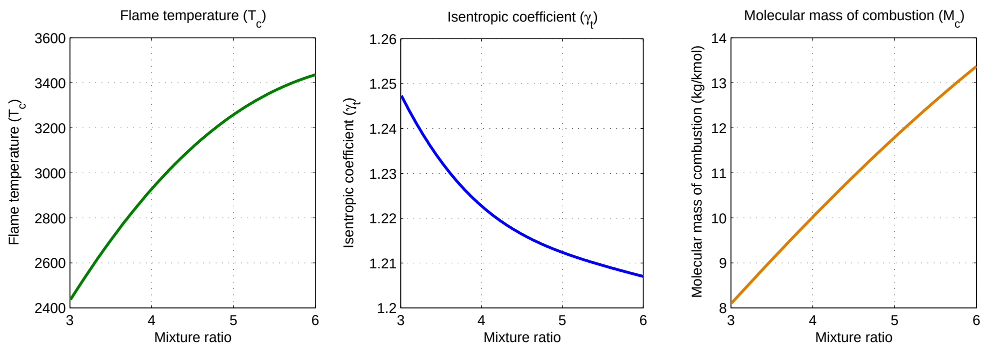


Figure A.5: Propulsive parameters

We have the following relationship between the cross-sectional area of nozzle throat and the throat's diameter :

$$A_{ch} = (8(100D_c)^{(-0.6)} + 1.25)\pi\frac{D_c^2}{4} \tag{A.25}$$

The diameter of the chamber is given by :

$$D_{ch} = 2\sqrt{\frac{A_{ch}}{\pi}} \tag{A.26}$$

The length of the chamber is given by :

$$L_{ch} = L^*\left(\frac{D_c}{D_{ch}}\right)^2 \tag{A.27}$$

with L^* the characteristic length (1.02 for the couple LOX/LH2). The thickness of the chamber is given by :

$$e_{ch} = Pc.f_p\frac{D_{ch}}{2\sigma_{ch}} \tag{A.28}$$

with $\sigma_{ch} = 310MPa$ the tensile strength of the chamber material, $f_p = 2$ a safety coefficient and $\rho_{ch} = 8000kg.m^{-3}$ the density of the chamber material.
Finally, the mass of the chamber is given by :

$$M_{ch} = (2.A_{ch} + \pi D_{ch}L_{ch})e_{ch}\rho_{ch} \tag{A.29}$$

A.5 Nozzle

Inputs :

- Pc : chamber pressure (Pa)
- Pe : exit pressure (Pa)
- Rm : mixture ratio

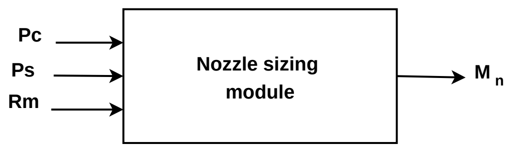


Figure A.6: Nozzle sizing module

The nozzle expansion ratio is given by :

$$\epsilon = \frac{\left(\frac{2}{\gamma+1}\right)^{\frac{1}{\gamma-1}} \cdot \frac{P_c}{P_s}^{\frac{1}{\gamma}}}{\sqrt{\left(\frac{\gamma+1}{\gamma-1}\right) \cdot \left(1 - \left(\frac{P_s}{P_c}\right)^{\frac{\gamma-1}{\gamma}}\right)}} \tag{A.30}$$

The nozzle throat area is :

$$A_n = A_c \epsilon \tag{A.31}$$

Then, the nozzle exit diameter is given by :

$$D_n = 2\sqrt{\frac{A_n}{\pi}} \tag{A.32}$$

The conical nozzle length (with a nozzle cone half angle of 15°) is :

$$LN = \frac{D_n - D_c}{2.\tan(15\pi/180)} \tag{A.33}$$

Thus, the nozzle length is :

$$L_{div} = LN.\lambda \tag{A.34}$$

The thickness of the nozzle inner structure is given by :

$$e_n = \frac{0,5.Pc.D_c.f_n}{2\sigma_{r_n}} \tag{A.35}$$

with f_n a safety coefficient ($f_n = 2$) and $\sigma_{r_n} = 310MPa$ the ultimate strength of the nozzle material. Finally, the nozzle mass is given by (with $\rho_n = 8000kg.m^{-3}$) :

$$M_n = \frac{\pi \rho_n.e_n.L_{div}(D_n + D_c)}{2} \tag{A.36}$$

A.6 Engine

The engine is calculated from the masses of the nozzle and the combustion chamber by the following relationship [Humble et al., 1995] :

$$M_{eng} = \frac{M_n + M_{ch}}{\xi} \tag{A.37}$$

with :

- $\xi = 0.2$ si $Pc \leq 20$ bars
- $\xi = \frac{0.2}{30}Pc + (0.6 - 50\frac{0.2}{30})$ if 20bars$\leq Pc \leq$ 50bars
- $\xi = 0.4$ else

A.7 Offset mass for models adjustment

At this level, the dry mass model underestimates the real dry mass of the existing cryogenic stages found in ESA Launch Vehicle Catalogue [European Space Agency, 2004]. In order to be consistent with the existing launch vehicles stages, an additional mass has to be included in the actual dry mass model. An estimation of this mass has been achieved, considering that at first order, this mass only depends of the stage's propellant mass :

$$M_{add} = Mp(-2, 3.10^{-7}\ Mp + 0, 07) \tag{A.38}$$

Appendix B

MDF robustness study and considerations about the Fixed Point Iteration

This appendix is devoted to two short studies which detail the robustness of the MDF method and some considerations about the use of the Fixed Point Iteration at the subsystem level.

B.1 Robustness of the MDF method

In order to evaluate the robustness of the MDF method and its ability to find the reference optimum in case of large search space, we have generated 250 randomized initializations in the search space defined in Table 7.4. We use a classical SQP algorithm as the optimizer.

The results obtained are summarized in Table B.1.

Number of initializations	250
Number of achieved optimizations	25
Success rate	10%
Average optimum	148.48t
Standard deviation	9t
Average relative difference with respect to the reference optimum (Chapter 9)	8.63%

Table B.1: Results of the MDF robustness study

This table shows that the coupled use of MDF and a gradient-based algorithm poses important problems in terms of the design process behavior (only 10% of success) and in terms of the robustness to the initialization. Indeed, the average found optimum is very far from the reference optimum (+8,63%). The results of this study motivate the use of a global search algorithm when the MDF is employed, and the need of a dedicated method (formulation and optimization algorithm) in order to efficiently solve the LVD problem.

B.2 Considerations about the use of the Fixed Point Iteration

In order to validate the use of an additional variable standing for the maximal axial load factor and the decomposition of the Fixed Point Iteration between the trajectory and the mass budget modules, we have compared the results obtained with and without FPI. To this end, from a very restrained search space and an initialization which allows the MDF method to converge to the reference optimum, we have compared the results obtained with different initializations of the FPI to the results obtained using the additional coupling variable (consequently without FPI). The initialization of the FPI concerns the maximal axial load factor (nf_0). All the results are summarized in Table B.2.

	FPI ($nf_0 = 4$)	FPI ($nf_0 = 4.5$)	FPI ($nf_0 = 5$)	No FPI
Calculation time (s)	367	457	682	288
Value of the optimum (t)	137.66	136.68	137.12	136.68
Nb of trajectory simulations	7826	9896	14941	6213
Nb of iterations	116	152	231	179
Nb of trajectory simulations per iteration	67	65	65	34

Table B.2: Results of the Fixed Point Iteration study

The results of this study point out that the use of the FPI may not converge if it is not correctly initialized. Indeed, only the FPI with nf_0 initialized to 4.5 allows the optimization to converge to the reference optimum. Moreover, the FPI requires a significant number of additional trajectory simulations that increases the calculation time of the global process. Consequently, the results of this study motivate the splitting up of the FPI and the use of the coupling variable $\widehat{n_{f_{max}}}$ in the design process.

Appendix C

Global Sensitivity Equation and Post-Optimality Analysis

C.1 Global Sensitivity Equation

The Global Sensitivity Equation (GSE) [Sobieszczanski-Sobieski, 1988a] is a computation technique used to calculate the derivatives in case of coupled system. This technique has been used in launch vehicle design [Olds, 1992, 1994] and results show that this technique appears to work well for the system level sensitivity analysis. Let us describe the use of the GSE in a general case. For that purpose, let us consider the system described in Figure C.1.

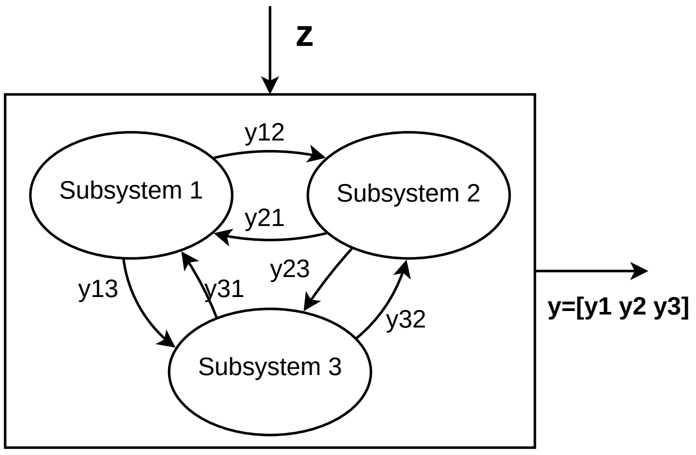


Figure C.1: Coupled system

In Figure C.1, y_i stands for $[y_{i1}, y_{i2}, y_{i3}]$. By calculating the differential form of the different variables $y_{ij_{j=1,2}}$, we have the generic form :

$$dy_i = \sum_{k \neq i} \frac{\partial y_i}{\partial y_k} dy_k + \frac{\partial y_i}{\partial z} dz \tag{C.1}$$

and the total derivative :

$$\frac{dy_i}{dz} = \sum_{k \neq i} \frac{\partial y_i}{\partial y_k} \frac{dy_k}{dz} + \frac{\partial y_i}{\partial z} \tag{C.2}$$

This equation reflects the fact that the total variation in y_i is the sum of the changes of each of the vectors $y_k \neq i$ multiplied by their effect on y_1. If we regroup all these equations, we can form the Global Sensitivity Equation :

$$\begin{bmatrix} I & -\frac{\partial y_1}{\partial y_2} & -\frac{\partial y_1}{\partial y_3} \\ -\frac{\partial y_2}{\partial y_1} & I & -\frac{\partial y_2}{\partial y_3} \\ -\frac{\partial y_3}{\partial y_1} & -\frac{\partial y_3}{\partial y_2} & I \end{bmatrix} \cdot \begin{bmatrix} \frac{dy_1}{dz} \\ \frac{dy_2}{dz} \\ \frac{dy_3}{dz} \end{bmatrix} = \begin{bmatrix} \frac{\partial y_1}{\partial z} \\ \frac{\partial y_2}{\partial z} \\ \frac{\partial y_3}{\partial z} \end{bmatrix} \tag{C.3}$$

The right side term of this equation is the local sensitivity vector and contains the local derivatives of the subsystem outputs with respect to the design variables z. The left side matrix is the global sensitivity matrix and contains the sensitivities of each subsystem output with respect to its inputs. Finally, the left side vector is the global sensitivity vector, which we want to compute. We can notice than y_i are vectors and consequently the terms $-\frac{\partial y_i}{\partial y_j}$ are matrices. We obtain the total derivative vector by using inversion matrix techniques :

$$\begin{bmatrix} \frac{dy_1}{dz} \\ \frac{dy_2}{dz} \\ \frac{dy_3}{dz} \end{bmatrix} = \begin{bmatrix} I & -\frac{\partial y_1}{\partial y_2} & -\frac{\partial y_1}{\partial y_3} \\ -\frac{\partial y_2}{\partial y_1} & I & -\frac{\partial y_2}{\partial y_3} \\ -\frac{\partial y_3}{\partial y_1} & -\frac{\partial y_3}{\partial y_2} & I \end{bmatrix}^{-1} \cdot \begin{bmatrix} \frac{\partial y_1}{\partial z} \\ \frac{\partial y_2}{\partial z} \\ \frac{\partial y_3}{\partial z} \end{bmatrix} \tag{C.4}$$

C.2 Post-Optimality Analysis

The Post-Optimality Analysis [Sobieszczanski-Sobieski et al., 1982] allows to estimate the gradient of the system level avoiding the reoptimizations of the subsystems (which is required by traditional computation of the gradient using finite differences). Let us describe the Post-Optimality Analysis in a simplified case (just one subsystem is considered). Given this assumption, we have at the system level the following optimization problem :

Minimize $f(z_0, z_1^*)$
With respect to z_0

with z_1^* the subsystem level optimized variables obtained during the optimization of the subsystem. At the subsystem level, the optimization problem is expressed as follows :

Minimize $f(z_0, z_1)$
With respect to z_1

Subject to

$$g(z_0, z_1) \leq 0 \tag{C.5}$$

$$h(z_0, z_1) = 0 \tag{C.6}$$

The derivative of f with respect to z_0 is given by :

$$\frac{df}{dz_0} = \frac{\partial f}{\partial z_0} + \frac{\partial f}{\partial z_1}\frac{\partial z_1^*}{\partial z_0} \tag{C.7}$$

In the following, we assume that the optimization problem is a regular convex problem. Let us suppose that the dimension of the inequality constraints vector is m. The first order Karush-Kuhn-Tucker optimality conditions for the subsystem are expressed through the following system of equations :

$$\begin{aligned}
\frac{\partial f}{\partial z_1}(z_0, z_1^*) + \lambda^T \frac{\partial h}{\partial z_1}(z_0, z_1^*) + \mu^T \frac{\partial g}{\partial z_1}(z_0, z_1^*) &= 0 && (C.8)\\
h(z_0, z_1^*) &= 0 && (C.9)\\
g(z_0, z_1^*) &\leq 0 && (C.10)\\
\mu &\geq 0 && (C.11)\\
\mu_j . g_j(z_0, z_1^*) &= 0 \quad 1 \leq j \leq m && (C.12)
\end{aligned}$$

For simplification reasons, let us denote by e the equality and saturated inequality constraints at the point (z_0, z_1^*) and ν the corresponding associated Lagrange multipliers vectors. Thus, Equation (C.8) can be expressed as follows :

$$\frac{\partial f}{\partial z_1}(z_0, z_1^*) + \nu^T \frac{\partial e}{\partial z_1}(z_0, z_1^*) = 0 \tag{C.13}$$

If we multiply this equation at the right by $\frac{\partial z_1}{\partial z_0}(z_0, z_1^*)$, we obtain :

$$\frac{\partial f}{\partial z_1}.\frac{\partial z_1}{\partial z_0}(z_0, z_1^*) + \nu^T \frac{\partial e}{\partial z_1}.\frac{\partial z_1}{\partial z_0}(z_0, z_1^*) = 0 \tag{C.14}$$

If we assume that the set of saturated inequality constraints remains unchanged with respect to a small variation of z_0, we can write :

$$\frac{de}{dz_0}(z_0, z_1^*) = \frac{\partial e}{\partial z_0}(z_0, z_1^*) + \frac{\partial e}{\partial z_1}\frac{\partial z_1}{\partial z_0}(z_0, z_1^*) = 0 \tag{C.15}$$

Finally, if we replace in Equation (C.13) $\frac{\partial e}{\partial z_1}\frac{\partial z_1}{\partial z_0}(z_0, z_1^*)$ and $\frac{\partial f}{\partial z_1}\frac{\partial z_1}{\partial z_0}(z_0, z_1^*)$ by their expressions found in equations (C.14-C.15), we obtain :

$$\frac{df}{dz_0}(z_0, z_1^*) = \frac{\partial f}{\partial z_0}(z_0, z_1^*) + \nu^T \frac{\partial e}{\partial z_0}(z_0, z_1^*) \tag{C.16}$$

The Post-Optimality Analysis allows to estimate (at the 1^{st} order) the gradient of the system level objective function, which sometimes is not accurate enough in case of very non linear search space. One of the drawbacks of this technique is the requirement that the active set of inequality constraints has to remain active after the Post-Optimality Analysis. Some techniques, using cumulative response surfaces approximations have been proposed [Chandila et al., 2004] in order to avoid this requirement. In our problem, we assume that the search space is sufficiently regular to apply the Post-Optimality Analysis. This assumption has been verified by numerical comparison between the gradient obtained by using the Post-Optimality Analysis and the gradient obtained by using the finite differences (Chapter 9).

Appendix D

Demonstration of optimality conditions

In this appendix, we aim to demonstrate that the third stage-wise decomposition formulation satisfies the same 1^{st} order KKT conditions as the AAO formulation applied to the stage-wise decomposition. To this end, we firstly show the equivalence between the KKT conditions obtained by the first formulation and the conditions obtained with AAO. Then we show the equivalence between the first and third formulations using the Implicit Function Theorem.

D.1 KKT conditions for the AAO formulation

In this section, we consider the AAO formulation applied to the same optimization problem as the problem involved in the first stage-wise decomposition formulation. The AAO formulation consists in handling all the optimization variables $(z_{sh}, y, \bar{z}_1, \bar{z}_2, \bar{z}_3)$ with a single optimizer. This optimizer is subjected to all the coupling, design and trajectory constraints. The involved variables and constraints are detailed in Chapter 6. The optimization problem is the following :

Minimize $f(z_{sh}, y, \bar{z}_1, \bar{z}_2, \bar{z}_3) = f_1(z_{sh}, y, \bar{z}_1) + f_2(z_{sh}, y, \bar{z}_2) + f_3(z_{sh}, y, \bar{z}_3)$

With respect to $z_{sh}, y, \bar{z}_1, \bar{z}_2, \bar{z}_3$

Subject to

$$
\begin{aligned}
h_1(z_{sh}, y, \bar{z}_1) &= 0 && \text{(D.1)}\\
h_2(z_{sh}, y, \bar{z}_2) &= 0 && \text{(D.2)}\\
h_3(z_{sh}, y, \bar{z}_3) &= 0 && \text{(D.3)}\\
g_1(z_{sh}, y, \bar{z}_1) &\leq 0 && \text{(D.4)}\\
g_2(z_{sh}, y, \bar{z}_2) &\leq 0 && \text{(D.5)}\\
g_3(z_{sh}, y, \bar{z}_3) &\leq 0 && \text{(D.6)}\\
c_{01}(z_{sh}, y, \bar{z}_2) &= 0 && \text{(D.7)}\\
c_{02}(z_{sh}, y, \bar{z}_3) &= 0 && \text{(D.8)}
\end{aligned}
$$

The constraints c_{01} and c_{02} can be written as follows :

$$\begin{aligned} c_{01}(z_{sh}, y, \bar{z}_2) &= f_2(z_{sh}, y, \bar{z}_2) - y_1 && \text{(D.9)} \\ c_{02}(z_{sh}, y, \bar{z}_3) &= f_3(z_{sh}, y, \bar{z}_3) - y_2 && \text{(D.10)} \end{aligned}$$

with y_1 and y_2 the particular components of the coupling vector y which refer to the optimized stage masses. All the functions and constraints are computed at the subsystem level and sent to the system level which is in charge of the optimization and satisfaction of all the constraints. We can notice that the constraints h and g are vectors of constraints whereas c_{01} and c_{02} are scalar constraints.
The global KKT conditions on the Lagrangian of the objective function for the AAO formulation are the followings :

$$\frac{\partial f_1}{\partial z_{sh}} + \frac{\partial f_2}{\partial z_{sh}} + \frac{\partial f_3}{\partial z_{sh}} + \lambda_1^T \frac{\partial h_1}{\partial z_{sh}} + \lambda_2^T \frac{\partial h_2}{\partial z_{sh}} + \lambda_3^T \frac{\partial h_3}{\partial z_{sh}} + \mu_1^T \frac{\partial g_1}{\partial z_{sh}} + \mu_2^T \frac{\partial g_2}{\partial z_{sh}} + \mu_3^T \frac{\partial g_3}{\partial z_{sh}} + \nu_1 \frac{\partial c_{01}}{\partial z_{sh}} + \nu_2 \frac{\partial c_{02}}{\partial z_{sh}} = 0 \quad \text{(D.11)}$$

$$\frac{\partial f_1}{\partial y} + \frac{\partial f_2}{\partial y} + \frac{\partial f_3}{\partial y} + \lambda_1^T \frac{\partial h_1}{\partial y} + \lambda_2^T \frac{\partial h_2}{\partial y} + \lambda_3^T \frac{\partial h_3}{\partial y} + \mu_1^T \frac{\partial g_1}{\partial y} + \mu_2^T \frac{\partial g_2}{\partial y} + \mu_3^T \frac{\partial g_3}{\partial y} + \nu_1 \frac{\partial c_{01}}{\partial y} + \nu_2 \frac{\partial c_{02}}{\partial y} = 0 \quad \text{(D.12)}$$

$$\frac{\partial f_1}{\partial \bar{z}_1} + \frac{\partial f_2}{\partial \bar{z}_1} + \frac{\partial f_3}{\partial \bar{z}_1} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_1} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_1} + \lambda_3^T \frac{\partial h_3}{\partial \bar{z}_1} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_1} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_1} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_1} + \nu_1 \frac{\partial c_{01}}{\partial \bar{z}_1} + \nu_2 \frac{\partial c_{02}}{\partial \bar{z}_1} = 0 \quad \text{(D.13)}$$

$$\frac{\partial f_1}{\partial \bar{z}_2} + \frac{\partial f_2}{\partial \bar{z}_2} + \frac{\partial f_3}{\partial \bar{z}_2} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_2} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + \lambda_3^T \frac{\partial h_3}{\partial \bar{z}_2} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_2} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_2} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_2} + \nu_1 \frac{\partial c_{01}}{\partial \bar{z}_2} + \nu_2 \frac{\partial c_{02}}{\partial \bar{z}_2} = 0 \quad \text{(D.14)}$$

$$\frac{\partial f_1}{\partial \bar{z}_3} + \frac{\partial f_2}{\partial \bar{z}_3} + \frac{\partial f_3}{\partial \bar{z}_3} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_3} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_3} + \lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_3} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_3} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_3} + \nu_1 \frac{\partial c_{01}}{\partial \bar{z}_3} + \nu_2 \frac{\partial c_{02}}{\partial \bar{z}_3} = 0 \quad \text{(D.15)}$$

with λ (respectively μ and ν) are the Lagrange multipliers associated to the constraints h (respectively g and c). We can notice that $\bar{z}_i$ are used only in the computation of f_i, g_i, h_i and c_{0i-1}. Therefore, $\forall i \neq j$ we have :

$$\begin{aligned} \frac{\partial f_i}{\partial \bar{z}_j} &= 0 && \text{(D.16)} \\ \frac{\partial g_i}{\partial \bar{z}_j} &= 0 && \text{(D.17)} \\ \frac{\partial h_i}{\partial \bar{z}_j} &= 0 && \text{(D.18)} \end{aligned}$$

and :

$$\begin{aligned} \frac{\partial c_{01}}{\partial \bar{z}_1} &= 0 && \text{(D.19)} \\ \frac{\partial c_{01}}{\partial \bar{z}_3} &= 0 && \text{(D.20)} \\ \frac{\partial c_{02}}{\partial \bar{z}_1} &= 0 && \text{(D.21)} \\ \frac{\partial c_{02}}{\partial \bar{z}_2} &= 0 && \text{(D.22)} \end{aligned}$$

Consequently, the KKT conditions for the Lagrangian of the AAO formulation can be written as follows :

$$\frac{\partial f_1}{\partial z_{sh}} + \frac{\partial f_2}{\partial z_{sh}} + \frac{\partial f_3}{\partial z_{sh}} + \lambda_1^T \frac{\partial h_1}{\partial z_{sh}} + \lambda_2^T \frac{\partial h_2}{\partial z_{sh}} + \lambda_3^T \frac{\partial h_3}{\partial z_{sh}} + \mu_1^T \frac{\partial g_1}{\partial z_{sh}} + \mu_2^T \frac{\partial g_2}{\partial z_{sh}} + \mu_3^T \frac{\partial g_3}{\partial z_{sh}} + \nu_1 \frac{\partial c_{01}}{\partial z_{sh}} + \nu_2 \frac{\partial c_{02}}{\partial z_{sh}} = 0 \quad (D.23)$$

$$\frac{\partial f_1}{\partial y} + \frac{\partial f_2}{\partial y} + \frac{\partial f_3}{\partial y} + \lambda_1^T \frac{\partial h_1}{\partial y} + \lambda_2^T \frac{\partial h_2}{\partial y} + \lambda_3^T \frac{\partial h_3}{\partial y} + \mu_1^T \frac{\partial g_1}{\partial y} + \mu_2^T \frac{\partial g_2}{\partial y} + \mu_3^T \frac{\partial g_3}{\partial y} + \nu_1 \frac{\partial c_{01}}{\partial y} + \nu_2 \frac{\partial c_{02}}{\partial y} = 0 \quad (D.24)$$

$$\frac{\partial f_1}{\partial \bar{z}_1} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_1} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_1} = 0 \quad (D.25)$$

$$\frac{\partial f_2}{\partial \bar{z}_2} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_2} + \nu_1 \frac{\partial c_{01}}{\partial \bar{z}_2} = 0 \quad (D.26)$$

$$\frac{\partial f_3}{\partial \bar{z}_3} + \lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_3} + \nu_2 \frac{\partial c_{02}}{\partial \bar{z}_3} = 0 \quad (D.27)$$

D.2 KKT conditions for the first formulation

Using the first formulation, four optimizers are employed. The constraints are distributed according to the different optimizers.

System level :

The optimization problem at the system level is the following :

Minimize $f^*(z_{sh}, y) = f_1^*(z_{sh}, y) + f_2^*(z_{sh}, y) + f_3^*(z_{sh}, y)$

With respect to z_{sh}, y

Subject to

$$c_{01}^*(z_{sh}, y) = 0 \quad (D.28)$$

$$c_{02}^*(z_{sh}, y) = 0 \quad (D.29)$$

(D.30)

Here, $f^*(z_{sh}, y) = f(z_{sh}, y, \bar{z}_i^*(z_{sh}, y))$ means that f is the result of the subsystem optimizations (calculation of $\bar{z}_i^*$), and

$$c_{01}^*(z_{sh}, y) = f_2^*(z_{sh}, y) - y_1 \quad (D.31)$$

$$c_{02}^*(z_{sh}, y) = f_3^*(z_{sh}, y) - y_2 \quad (D.32)$$

Subsystem level

The optimization problem for the ith subsystem is the following :

Given z_{sh}, y

Minimize $f_i(z_{sh}, y, \bar{z}_i)$

With respect to $\bar{z}_i$

Subject to

$$h_i(z_{sh}, y, \bar{z}_i) = 0 \quad (D.33)$$

$$g_i(z_{sh}, y, \bar{z}_i) \leq 0 \quad (D.34)$$

The KKT conditions about the Lagrangians of the system and subsystem levels are the followings :

System level :

$$\frac{\partial f_1^*}{\partial z_{sh}} + \frac{\partial f_2^*}{\partial z_{sh}} + \frac{\partial f_3^*}{\partial z_{sh}} + \nu_1 \frac{\partial c_1^*}{\partial z_{sh}} + \nu_2 \frac{\partial c_2^*}{\partial z_{sh}} = 0 \tag{D.35}$$

$$\frac{\partial f_1^*}{\partial y} + \frac{\partial f_2^*}{\partial y} + \frac{\partial f_3^*}{\partial y} + \nu_1 \frac{\partial c_1^*}{\partial y} + \nu_2 \frac{\partial c_2^*}{\partial y} = 0 \tag{D.36}$$

Subsystem level :

$$\frac{\partial f_1}{\partial \bar{z}_1} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_1} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_1} = 0 \tag{D.37}$$

$$\frac{\partial f_2}{\partial \bar{z}_2} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_2} = 0 \tag{D.38}$$

$$\frac{\partial f_3}{\partial \bar{z}_3} + \lambda_1^T \frac{\partial h_3}{\partial \bar{z}_3} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_3} = 0 \tag{D.39}$$

At the system-level, from the equations (8.20) given by the Post-Optimality Analysis, we have:

$$\frac{\partial f_i^*}{\partial z_{sh}} = \frac{\partial f_i}{\partial z_{sh}} + \lambda_i^T \frac{\partial h_i}{\partial z_{sh}} + \mu_i^T \frac{\partial g_i}{\partial z_{sh}} \tag{D.40}$$

$$\frac{\partial f_i^*}{\partial y} = \frac{\partial f_i}{\partial y} + \lambda_i^T \frac{\partial h_i}{\partial y} + \mu_i^T \frac{\partial g_i}{\partial y} \tag{D.41}$$

$$\frac{\partial c_{01}^*}{\partial z_{sh}} = \frac{\partial c_{01}}{\partial z_{sh}} + \lambda_2^T \frac{\partial h_2}{\partial z_{sh}} + \mu_2^T \frac{\partial g_2}{\partial z_{sh}} \tag{D.42}$$

$$\frac{\partial c_{01}^*}{\partial y} = \frac{\partial c_{01}}{\partial y} + \lambda_2^T \frac{\partial h_2}{\partial y} + \mu_2^T \frac{\partial g_2}{\partial y} \tag{D.43}$$

$$\frac{\partial c_{02}^*}{\partial z_{sh}} = \frac{\partial c_{02}}{\partial z_{sh}} + \lambda_3^T \frac{\partial h_3}{\partial z_{sh}} + \mu_3^T \frac{\partial g_3}{\partial z_{sh}} \tag{D.44}$$

$$\frac{\partial c_{02}^*}{\partial y} = \frac{\partial c_{02}}{\partial y} + \lambda_3^T \frac{\partial h_3}{\partial y} + \mu_3^T \frac{\partial g_3}{\partial y} \tag{D.45}$$

Notations : $\frac{\partial f_i^*}{\partial z}$ stands for the partial derivative of f_i with respect to z involving the reoptimization of the subsystem i (additional computation of the $\bar{z}_i^*$) , whereas $\frac{\partial f_i}{\partial z}$ stands for the "standard" partial derivative of f_i with respect to z (not involving the recomputation of the $\bar{z}_i^*$).
Combining the previous equations with the KKT conditions of the system and subsystem

levels, we obtain :

$$\frac{\partial f_1}{\partial z_{sh}} + \lambda_1^T \frac{\partial h_1}{\partial z_{sh}} + \mu_1^T \frac{\partial g_1}{\partial z_{sh}} + \frac{\partial f_2}{\partial z_{sh}} + \lambda_2^T \frac{\partial h_2}{\partial z_{sh}} + \mu_2^T \frac{\partial g_2}{\partial z_{sh}} + \frac{\partial f_3}{\partial z_{sh}} + \lambda_3^T \frac{\partial h_3}{\partial z_{sh}} \tag{D.46}$$

$$+\mu_3^T \frac{\partial g_3}{\partial z_{sh}} + \nu_1 \left(\frac{\partial c_{01}}{\partial z_{sh}} + \lambda_2^T \frac{\partial h_2}{\partial z_{sh}} + \mu_2^T \frac{\partial g_2}{\partial z_{sh}} \right) + \nu_2 \left(\frac{\partial c_{02}}{\partial z_{sh}} + \lambda_3^T \frac{\partial h_3}{\partial z_{sh}} + \mu_3^T \frac{\partial g_3}{\partial z_{sh}} \right) = 0$$

$$\frac{\partial f_1}{\partial y} + \lambda_1^T \frac{\partial h_1}{\partial y} + \mu_1^T \frac{\partial g_1}{\partial y} + \frac{\partial f_2}{\partial y} + \lambda_2^T \frac{\partial h_2}{\partial y} + \mu_2^T \frac{\partial g_2}{\partial y} + \frac{\partial f_3}{\partial z_{sh}} + \lambda_3^T \frac{\partial h_3}{\partial y} \tag{D.47}$$

$$+\mu_3^T \frac{\partial g_3}{\partial y} + \nu_1 \left(\frac{\partial c_{01}}{\partial y} + \lambda_2^T \frac{\partial h_2}{\partial y} + \mu_2^T \frac{\partial g_2}{\partial y} \right) + \nu_2 \left(\frac{\partial c_{02}}{\partial y} + \lambda_3^T \frac{\partial h_3}{\partial y} + \mu_3^T \frac{\partial g_3}{\partial y} \right) = 0$$

$$\frac{\partial f_1}{\partial \bar{z}_1} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_1} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_1} = 0 \tag{D.48}$$

$$\frac{\partial f_2}{\partial \bar{z}_2} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_2} = 0 \tag{D.49}$$

$$\frac{\partial f_3}{\partial \bar{z}_3} + \lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_3} = 0 \tag{D.50}$$

Let be :

$$\Lambda_2 = (1+\nu_1)\lambda_2 \tag{D.51}$$

$$\Lambda_3 = (1+\nu_2)\lambda_3 \tag{D.52}$$

$$M_2 = (1+\nu_1)\mu_2 \tag{D.53}$$

$$M_3 = (1+\nu_2)\mu_3 \tag{D.54}$$

The KKT conditions can be rewritten as follows :

$$\frac{\partial f_1}{\partial z_{sh}} + \frac{\partial f_2}{\partial z_{sh}} + \frac{\partial f_3}{\partial z_{sh}} + \lambda_1^T \frac{\partial h_1}{\partial z_{sh}} + \Lambda_2^T \frac{\partial h_2}{\partial z_{sh}} + \Lambda_3^T \frac{\partial h_3}{\partial z_{sh}} + \mu_1^T \frac{\partial g_1}{\partial z_{sh}} + M_2^T \frac{\partial g_2}{\partial z_{sh}} + M_3^T \frac{\partial g_3}{\partial z_{sh}} + \nu_1 \frac{\partial c_{01}}{\partial z_{sh}} + \nu_2 \frac{\partial c_{02}}{\partial z_{sh}} = 0 \tag{D.55}$$

$$\frac{\partial f_1}{\partial y} + \frac{\partial f_2}{\partial y} + \frac{\partial f_3}{\partial y} + \lambda_1^T \frac{\partial h_1}{\partial y} + \Lambda_2^T \frac{\partial h_2}{\partial y} + \Lambda_3^T \frac{\partial h_3}{\partial y} + \mu_1^T \frac{\partial g_1}{\partial y} + M_2^T \frac{\partial g_2}{\partial y} + M_3^T \frac{\partial g_3}{\partial y} + \nu_1 \frac{\partial c_{01}}{\partial y} + \nu_2 \frac{\partial c_{02}}{\partial y} = 0 \tag{D.56}$$

$$\frac{\partial f_1}{\partial \bar{z}_1} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_1} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_1} = 0 \tag{D.57}$$

$$\frac{\partial f_2}{\partial \bar{z}_2} + \lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_2} = 0 \tag{D.58}$$

$$\frac{\partial f_3}{\partial \bar{z}_3} + \lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_3} = 0 \tag{D.59}$$

In order to show the equivalence with respect to the KKT conditions found for the AAO formulation, it remains to be seen that the equations (D.57-D.59) are equivalent to the followings :

$$\frac{\partial f_1}{\partial \bar{z}_1} + \lambda_1^T \frac{\partial h_1}{\partial \bar{z}_1} + \mu_1^T \frac{\partial g_1}{\partial \bar{z}_1} = 0 \tag{D.60}$$

$$\frac{\partial f_2}{\partial \bar{z}_2} + \Lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + M_2^T \frac{\partial g_2}{\partial \bar{z}_2} + \nu_1 \frac{\partial c_{01}}{\partial \bar{z}_2} = 0 \tag{D.61}$$

$$\frac{\partial f_3}{\partial \bar{z}_3} + \Lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + M_3^T \frac{\partial g_3}{\partial \bar{z}_3} + \nu_2 \frac{\partial c_{02}}{\partial \bar{z}_3} = 0 \tag{D.62}$$

If we replace in the equations (D.58) and (D.59) λ_2, μ_2, λ_3, μ_3 by their expressions in function of Λ_2, M_2, Λ_3, M_3, ν_1 and ν_2 (Eq. D.51-D.54), we have :

$$\frac{\partial f_2}{\partial \bar{z}_2} + \Lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + M_2^T \frac{\partial g_2}{\partial \bar{z}_2} - \nu_1 \left(\lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + \mu_2^T \frac{\partial g_2}{\partial \bar{z}_2} \right) = 0 \quad \text{(D.63)}$$

$$\frac{\partial f_3}{\partial \bar{z}_3} + \Lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + M_3^T \frac{\partial g_3}{\partial \bar{z}_3} - \nu_2 \left(\lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + \mu_3^T \frac{\partial g_3}{\partial \bar{z}_3} \right) = 0 \quad \text{(D.64)}$$

We can remark that the terms into parentheses correspond to $-\frac{\partial f_2}{\partial \bar{z}_2}$ and $-\frac{\partial f_3}{\partial \bar{z}_3}$ (Eq. D.38-D.39). Moreover, from the expressions of c_{01} and c_{02} (Eq. D.31-D.32), we have :

$$\frac{\partial c_{01}}{\partial \bar{z}_2} = \frac{\partial f_2}{\partial \bar{z}_2} \quad \text{(D.65)}$$

$$\frac{\partial c_{02}}{\partial \bar{z}_3} = \frac{\partial f_3}{\partial \bar{z}_3} \quad \text{(D.66)}$$

Finally, we have :

$$\frac{\partial f_2}{\partial \bar{z}_2} + \Lambda_2^T \frac{\partial h_2}{\partial \bar{z}_2} + M_2^T \frac{\partial g_2}{\partial \bar{z}_2} + \nu_1 \frac{\partial c_{01}}{\partial \bar{z}_2} = 0 \quad \text{(D.67)}$$

$$\frac{\partial f_3}{\partial \bar{z}_3} + \Lambda_3^T \frac{\partial h_3}{\partial \bar{z}_3} + M_3^T \frac{\partial g_3}{\partial \bar{z}_3} + \nu_2 \frac{\partial c_{02}}{\partial \bar{z}_3} = 0 \quad \text{(D.68)}$$

We have shown the equivalence between the expressions of the KKT conditions about the Lagrangian between the AAO and the first formulation. For the additional conditions, we have :

$$M_{2i}.g_{2i} = 0 \quad \text{since} \quad \mu_{2i}.g_{2i} = 0 \quad \forall i$$

$$M_{3j}.g_{3j} = 0 \quad \text{since} \quad \mu_{2j}.g_{2j} = 0 \quad \forall j$$

In order to be consistent with respect to the KKT conditions of AAO, we have to show that:

$$M_{2i} \geq 0 \quad \forall i \quad \text{(D.69)}$$

$$M_{3j} \geq 0 \quad \forall j \quad \text{(D.70)}$$

These Lagrange multipliers depend on ν_1 and ν_2 which can take *a priori* any values as Lagrange multipliers associated to equality constraints. In order to have M_2 and M_3 positive, ν_1 and ν_2 must be greater than -1. Let us see if this condition is verified for the launch vehicle design optimization problem.
Let us express the system level KKT conditions concerning the variables y_1 and y_2 (representing the estimated masses of the stages). We have :

$$\frac{\partial f_1^*}{\partial y_1} + \frac{\partial f_2^*}{\partial y_1} + \frac{\partial f_3^*}{\partial y_1} + \nu_1 \left(\frac{\partial f_2^*}{\partial y_1} - \frac{\partial y_1}{\partial y_1} \right) + \nu_2 \left(\frac{\partial f_3^*}{\partial y_1} - \frac{\partial y_2}{\partial y_1} \right) = 0 \quad \text{(D.71)}$$

$$\frac{\partial f_1^*}{\partial y_2} + \frac{\partial f_2^*}{\partial y_2} + \frac{\partial f_3^*}{\partial y_2} + \nu_1 \left(\frac{\partial f_2^*}{\partial y_2} - \frac{\partial y_2}{\partial y_1} \right) + \nu_2 \left(\frac{\partial f_3^*}{\partial y_2} - \frac{\partial y_2}{\partial y_2} \right) = 0 \quad \text{(D.72)}$$

y_1 are involved only in the optimization of the first stage, and y_2 only in the optimization of the second stage, thus the previous equations can be simplified as follows :

$$\begin{aligned}\frac{\partial f_1^*}{\partial y_1} + \nu_1 (0-1) &= 0 \qquad \text{(D.73)}\\ \frac{\partial f_1^*}{\partial y_2} + \frac{\partial f_2^*}{\partial y_2} + \nu_1 \left(\frac{\partial f_2^*}{\partial y_2} - 0\right) + \nu_2 (0-1) &= 0 \qquad \text{(D.74)}\end{aligned}$$

Thus :

$$\begin{aligned}\nu_1 &= \frac{\partial f_1^*}{\partial y_1} \qquad \text{(D.75)}\\ \nu_2 &= \frac{\partial f_1^*}{\partial y_2} + \frac{\partial f_2^*}{\partial y_2} + \nu_1 \left(\frac{\partial f_2^*}{\partial y_2} - 0\right) \qquad \text{(D.76)}\end{aligned}$$

Assuming the fact that for a given stage at the optimum, the mass of the optimal stage will be automatically increased when its payload mass increases (otherwise it is not the optimal stage), we have $\frac{\partial f_1^*}{\partial y_1} \geq 0$ and consequently $\nu_1 \geq 0$. For the same reason, and because $\nu_1 \geq 0$, we have also $\nu_2 \geq 0$. Consequently, for our case, the conditions (D.69-D.70) are satisfied and the first formulation is equivalent to AAO applied to the stage-wise decomposition.

D.3 Equivalence between the first and the third formulations

In order to show the equivalence between the first and third formulations, we have to demonstrate that the 1^{st} order KKT conditions of the first formulation are the same as the 1^{st} optimality conditions obtained for the third formulation. In this section, for the sake of convenience, we note in this paragraph z_{F3} the system level optimization variables of the third formulation (z_{F3} regroups both z_{sh} and y_{F3}). The optimization variables at the system level for the first formulation are z_{F3} and the coupling variables corresponding to the estimated payload masses of the first and second stages : y_1 and y_2.

Considering the previous notations, the optimality conditions of the third formulation at the system level are the followings :

$$\frac{df}{dz_{F3}} = 0 \qquad \text{(D.77)}$$

Given the same notations, the optimality conditions of the first formulation at the system level are the followings :

$$\frac{\partial f}{\partial z_{F3}} + \nu_1 \frac{\partial c_1}{\partial z_{F3}} + \nu_2 \frac{\partial c_2}{\partial z_{F3}} = 0 \tag{D.78}$$
$$\frac{\partial f}{\partial y_1} + \nu_1 \frac{\partial c_1}{\partial y_1} + \nu_2 \frac{\partial c_2}{\partial y_1} = 0 \tag{D.79}$$
$$\frac{\partial f}{\partial y_2} + \nu_1 \frac{\partial c_1}{\partial y_2} + \nu_2 \frac{\partial c_2}{\partial y_2} = 0 \tag{D.80}$$
$$c_1(z_{F3}, y_1, y_2) = 0 \tag{D.81}$$
$$c_2(z_{F3}, y_2) = 0 \tag{D.82}$$

By applying the Implicit Function Theorem (Theorem 8.6.1) to the equation (D.82) at the optimum, we have:

$$\frac{dy_2}{dz_{F3}} = -\frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}} \tag{D.83}$$

By applying the Implicit Function Theorem to the equation (D.81) and replacing the term $\frac{dy_2}{dz_{F3}}$ by its expression found in the equation (D.83), we obtain :

$$\frac{dy_1}{dz_{F3}} = -\frac{\frac{\partial c_1}{\partial z_{F3}}}{\frac{\partial c_1}{\partial y_1}} + \frac{\frac{\partial c_1}{\partial y_2}}{\frac{\partial c_1}{\partial y_1}} . \frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}} \tag{D.84}$$

The following operation : (D.78) $-$ (D.79)$\frac{dy_2}{dz_{F3}}$ $-$ (D.80)$\frac{dy_1}{dz_{F3}}$ gives :

$$\frac{\partial f}{\partial z_{F3}} + \nu_1 \frac{\partial c_1}{\partial z_{F3}} + \nu_2 \frac{\partial c_2}{\partial z_{F3}} + \left(\frac{\partial f}{\partial y_1} + \nu_1 \frac{\partial c_1}{\partial y_1} + \nu_2 \frac{\partial c_2}{\partial y_1}\right)\left(-\frac{\frac{\partial c_1}{\partial z_{F3}}}{\frac{\partial c_1}{\partial y_1}} + \frac{\frac{\partial c_1}{\partial y_2}}{\frac{\partial c_1}{\partial y_1}} . \frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}}\right)$$
$$+ \left(\frac{\partial f}{\partial y_2} + \nu_1 \frac{\partial c_1}{\partial y_2} + \nu_2 \frac{\partial c_2}{\partial y_2}\right)\left(-\frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}}\right) = 0 \tag{D.85}$$

Developing the previous equation, we have :

$$\frac{\partial f}{\partial z_{F3}} + \nu_1 \cancel{\frac{\partial c_1}{\partial z_{F3}}} + \nu_2 \cancel{\frac{\partial c_2}{\partial z_{F3}}} - \frac{\partial f}{\partial y_1} \frac{\frac{\partial c_1}{\partial z_{F3}}}{\frac{\partial c_1}{\partial y_1}} + \frac{\partial f}{\partial y_1} \frac{\frac{\partial c_1}{\partial y_2}}{\frac{\partial c_1}{\partial y_1}} . \frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}} - \nu_1 \cancel{\cancel{\frac{\partial c_1}{\partial y_1}} \frac{\frac{\partial c_1}{\partial z_{F3}}}{\cancel{\frac{\partial c_1}{\partial y_1}}}}$$
$$+\nu_1 \cancel{\cancel{\frac{\partial c_1}{\partial y_1}} \frac{\frac{\partial c_1}{\partial y_2}}{\cancel{\frac{\partial c_1}{\partial y_1}}} . \frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}}} - \nu_2 \frac{\partial c_2}{\partial y_1} . \frac{\frac{\partial c_1}{\partial z_{F3}}}{\frac{\partial c_1}{\partial y_1}} + \nu_2 \frac{\partial c_2}{\partial y_1} \frac{\frac{\partial c_1}{\partial y_2}}{\frac{\partial c_1}{\partial y_1}} . \frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}} - \frac{\partial f}{\partial y_2} \frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}}$$
$$-\nu_1 \cancel{\frac{\partial c_1}{\partial y_2} \frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}}} - \nu_2 \cancel{\cancel{\frac{\partial c_2}{\partial y_2}} \frac{\frac{\partial c_2}{\partial z_{F3}}}{\cancel{\frac{\partial c_2}{\partial y_2}}}} = 0 \tag{D.86}$$

We can notice that y_1 is not involved in the calculation of c_2. Consequently, we have $\frac{\partial c_2}{\partial y_1} = 0$, and

$$\frac{\partial f}{\partial z_{F3}} - \frac{\partial f}{\partial y_1}\frac{\frac{\partial c_1}{\partial z_{F3}}}{\frac{\partial c_1}{\partial y_1}} + \frac{\partial f}{\partial y_1}\frac{\frac{\partial c_1}{\partial y_2}}{\frac{\partial c_1}{\partial y_1}}.\frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}} - \frac{\partial f}{\partial y_2}\frac{\frac{\partial c_2}{\partial z_{F3}}}{\frac{\partial c_2}{\partial y_2}} = 0$$

Thus,

$$\frac{\partial f}{\partial z_{F3}} + \frac{\partial f}{\partial y_1}\frac{dy_1}{dz_{F3}} + \frac{\partial f}{\partial y_2}\frac{dy_2}{dz_{F3}} = 0 \tag{D.87}$$

Finally, we have :

$$\frac{df}{dz_{F3}} = 0 \tag{D.88}$$

which is equivalent to the 1^{st} optimality condition of the third formulation.
Concerning the subsystem-level, since the first and third formulations involve the same optimization problem for each stage, the subsystem level KKT conditions for the first and third formulation are the same.

We have shown in this section the equivalence between the first and third formulations. Consequently, since the first formulation is equivalent to AAO, we have demonstrated that the third formulation is equivalent to AAO applied to the stage-wise decomposition.

Appendix E

Résumé étendu de la thèse

Introduction

Depuis le début du XX^{ème} siècle et les travaux de Konstantin Tsiolkovsky sur la conception de fusées multiétage, publiés en 1903 dans “L'exploration de l'espace cosmique par des engins à réaction” , la conception de lanceurs n'a cessé de croître et est devenue un domaine stratégique car permettant l'accès à l'espace. Ce secteur, en raison du développement de nouveaux lanceurs durant ces dernières années, est devenu de plus en plus compétitif. Pour cette raison, il est primordial d'être capable de concevoir des lanceurs de plus en plus performants, *i.e.* capables d'injecter en orbite des charges utiles de plus en plus importantes et ce à moindre coût, tout en garantissant un niveau de fiabilité important. Ceci requiert dans les phases d'avant projet un processus d'optimisation capable d'explorer un large domaine de recherche afin de trouver rapidement la meilleure configuration qui répond aux spécifications de la mission.

La conception de lanceurs (LVD pour Launch Vehicle Design) met en jeu de nombreuses disciplines telles l'aérodynamique, le calcul structural, la trajectoire, le dimensionnement massique, *etc.* Ces disciplines requièrent des outils adaptés et peuvent induire des décisions conflictuelles. Par exemple, le choix d'un large diamètre d'étage a des effets antagonistes sur les performances du lanceur à travers l'aérodynamique (calcul de traînée) et le dimensionnement structural. De tels objectifs conflictuels induisent un processus de conception complexe incluant la recherche de compromis multidisciplinaires. Par conséquent, en plus de maîtriser les différentes technologies disciplinaires, la LVD requiert des méthodologies de conception orientées système afin d'organiser le processus de conception de manière efficace.

La méthode classiquement utilisée en ingénierie consiste en une boucle entre les différentes tâches disciplinaires. A chaque itération de cette boucle, chaque discipline est réoptimisée selon les données transmises par la discipline précédente. Celle-ci fournit à la discipline suivante des nouvelles données, en fonction des résultats obtenus durant sa propre optimisation. Ainsi, ce processus de conception requiert plusieurs itérations pour converger, peut présenter des problèmes pour trouver (si nécessaire) les compromis entre les disciplines et peut ne pas aboutir à l'optimum global. Par conséquent, la LVD nécessite l'utilisation d'outils adaptés afin d'exploiter les couplages entre les différentes disciplines, de faciliter la recherche de compromis ainsi que d'améliorer la robustesse et l'efficacité du processus de conception.

L'optimisation multidisciplinaire (MDO pour Multidisciplinary Design Optimization), est un domaine de recherche des sciences d'ingénierie qui a pour but d'élaborer des méthodologies de conception dédiées à la résolution des problèmes multidisciplinaires complexes. A cette fin, la MDO regroupe un certain nombre de thématiques comme l'élaboration de nouvelles formulations du problème de conception, la gestion d'incertitudes dans le but d'améliorer la robustesse du processus ou encore l'élaboration de nouveaux algorithmes d'optimisation lorsque les algorithmes classiques sont inefficaces.
De nombreuses méthodes MDO peuvent être trouvées dans la littérature [Alexandrov and Hussaini, 1995; Balling and Sobieszczanski-Sobieski, 1994; Sobieszczanski-Sobieski and Haftka, 1997; Agte et al., 2009; Tosserams et al., 2009]. Ces méthodes peuvent être décomposées en deux catégories suivant la présence d'un ou plusieurs niveaux d'optimisation.
La méthode MDO la plus utilisée dans la LVD est la méthode MDF (MultiDiscipline Feasible)[Duranté et al., 2004; Tsuchiya and Mori, 2004; Bayley and Hartfield, 2007]. Cette méthode mono niveau décompose le problème de conception suivant les différentes disciplines et associe un optimiseur global (niveau système) et des outils d'analyses disciplinaires (niveau sous-système). Dans le but d'assurer la cohérence au niveau des couplages, la méthode met en jeu au niveau sous-système une analyse multidisciplinaire (MDA Multi Disciplinary Analysis). La méthode MDF facilite l'obtention de l'optimum global par rapport aux méthodes d'ingénierie classiques car elle met en jeu un optimiseur unique qui gère toutes les variables d'optimisation. De plus, cette méthode rend possible la recherche de compromis, grâce à l'utilisation de la MDA. Néanmoins, la gestion des variables de conception par un seul optimiseur induit un domaine de recherche important. Ceci peut poser un certain nombre de problèmes tels que la nécessité d'une initialisation appropriée et une bonne connaissance du domaine de variation des variables de conception afin de converger. Par conséquent, cette méthode est seulement adaptée aux problèmes de conception de moindre taille pour lesquels le domaine de recherche peut être bien connu à l'avance.

Dans le but de traiter des problèmes de conception plus importants, des méthodes MDO mettant en jeu plusieurs niveaux d'optimisation ont été proposées. Ces méthodes (*e.g.* Collaborative Optimization et ses dérivées [Braun and Kroo, 1995; Alexandrov and Lewis, 2000; DeMiguel and Murray, 2000], Bi-Level Integrated Synthesis [Sobieszczanski-Sobieski et al., 1998, 2000], *etc.*) décomposent le problème de conception selon les différentes disciplines. Celles-ci utilisent des optimiseurs disciplinaires au niveau sous-système et un optimiseur global au niveau système afin de coordonner les différentes optimisations disciplinaires et d'optimiser l'objectif global.

La conception de lanceurs est un problème MDO spécifique. En effet, celle-ci implique d'optimiser un système dynamique et combine les optimisations des variables de conception et de la trajectoire. L'optimisation de la trajectoire induit une simulation de trajectoire présentant des séparations d'étages. Cette dernière met en jeu un système d'équations différentielles ordinaires fortement non linéaires qui est complexe à intégrer. De plus, la trajectoire est soumise à des contraintes d'égalité strictes, dûes aux spécifications de la mission. Ces contraintes sont très difficiles à satisfaire et limitent considérablement le domaine de

recherche faisable *i.e.* le domaine dans lequel toutes les contraintes sont satisfaites.
Toutes les méthodes MDO appliquées à la LVD dans la littérature utilisent une décomposition du problème MDO selon les différentes disciplines. Dans cette décomposition, la trajectoire est le plus souvent considérée comme une “boîte noire ” du point de vue de l'optimiseur et est optimisée de la même manière que les autres disciplines. De ce fait, cette décomposition semble ne pas être la plus adaptée au problème spécifique de la conception de lanceurs dans lequel la trajectoire joue un rôle prédominant. Une autre décomposition du problème de conception qui exploite les couplages entre la trajectoire et l'optimisation des variables de conception apparaît être une voie d'amélioration intéressante.
Cette thèse est centrée sur l'élaboration de nouvelles méthodes MDO dédiées à la conception de lanceurs. Ces méthodes, appelées SWORD (pour Stage-Wise decomposition for Optimal Rocket Design), permettent de placer l'optimisation de trajectoire au coeur du processus d'optimisation. Les méthodes SWORD transforment le problème MDO initial en la coordination de problèmes MDO plus faciles résoudre. Des nouvelles formulations du problème de conception et les stratégies d'optimisation associées sont proposées. Les méthodes SWORD sont analysées et comparées à la méthode MDO la plus utilisée dans la littérature (MDF).

Cette thèse est organisée de la manière suivante :

La première partie dresse un état de l'art des méthodes MDO appliquées à la conception de lanceurs. Le chapitre 1 rappelle les concepts généraux relatifs à la MDO. Le second chapitre dresse un tableau des principales méthodes MDO proposées dans la litérature. Le chapitre 3 détaille les principaux algorithmes d'optimisation utilisés en MDO et les techniques utilisées en optimisation de trajectoire. Le dernier chapitre de cette partie (chapitre 4) analyse l'application des méthodes MDO dans la conception de lanceurs [Balesdent et al., 2011d] et propose quelques voies d'amélioration.

La seconde partie de la thèse est consacrée à la présentation des formulations SWORD [Balesdent et al., 2010a,b, 2011c,e]. Le chapitre 5 détaille le problème de conception de lanceurs et présente la formulation mathématique des méthodes proposées. Le chapitre 6 est consacré à la description du cas d'application : l'optimisation d'un lanceur tri-étage afin de minimiser la masse totale au décollage. Le chapitre 7 porte sur la comparaison théorique et numérique des différentes formulations proposées dans le cas d'une recherche globale.

La dernière partie de cette thèse présente la stratégie d'optimisation dédiée aux formulations SWORD décrites dans la seconde partie du manuscrit [Balesdent et al., 2011a,b]. Le chapitre 8 décrit les différentes phases de la stratégie d'optimisation proposée. Enfin, le chapitre 9 analyse les performances du processus d'optimisation complet (formulation MDO et stratégie d'optimisation). Les résultats mettent en évidence le potentiel des méthodes proposées. Nous présentons dans ce document un résumé des travaux de la thèse. Chaque partie est résumée afin d'exposer les différents problèmes abordés au cours de la thèse.

E.1 Panorama des méthodes MDO utilisées dans la conception de lanceurs

Cette section a pour but de présenter les différentes notions et méthodes utilisées dans la MDO. A cette fin, nous présentons d'abord la formulation mathématique du problème, puis nous décrivons les différents concepts de faisabilité et de gestion des différentes variables entrant en jeu dans le processus d'optimisation. Nous détaillons ensuite brièvement le problème de conception de lanceurs ainsi que les principales méthodes MDO utilisées dans la littérature. La dernière partie de cette section concerne l'analyse de l'application des méthodes MDO dans la conception de lanceurs et détaille quelques voies d'amélioration possibles.

E.1.1 Formulation d'un problème MDO et concepts généraux

Formulation mathématique du problème

La formulation générale d'un problème d'optimisation multidisciplinaire est la suivante :

Minimiser $$f(x, y, z)$$

Par rapport à $$z \in \mathcal{Z}$$

Soumis à

$$g(x, y, z) \leq 0 \tag{E.1}$$

$$h(x, y, z) = 0 \tag{E.2}$$

$$\forall i \in \{1, \cdots, n\}, \forall j \neq i, y_i = \{c_{ji}(x_j, y_j, z_j)\}_j \tag{E.3}$$

$$\forall i \in \{1, \cdots, n\}, R_i(x_i, y_i, z_i) = 0 \tag{E.4}$$

avec :

- z : variables de conception (*e.g.* propulsion, aérodynamique, structure, trajectoire, *etc.*), ces variables se décomposent en deux sous-ensembles z_{sh} et $\bar{z}_k$ suivant qu'elles sont partagées entre les disciplines (z_{sh}) ou qu'elles interviennent dans une seule discipline ($\bar{z}_k$ pour la discipline k),
- y : variables de couplage (*e.g.* facteur de charge estimé, *etc.*), utilisées pour relier les différentes disciplines,
- x : variables d'état, évoluant durant les analyses disciplinaires dans le but de trouver un équilibre au niveau des équations d'état,
- f : fonction objectif. Dans la conception de lanceurs, la fonction objectif est souvent un critère de masse (*e.g.* maximisation de la masse de charge utile, minimisation de la masse totale au décollage, *etc.*) ou un critère de coût,
- h : contraintes d'égalité (*e.g.* spécifications de la mission, *etc.*),
- g : contraintes d'inégalité (*e.g.* contraintes de pression, incidence maximale, *etc.*)
- c : fonctions de couplage, servant à calculer les variables de couplage issues des différentes disciplines (*e.g.* couplage entre la trajectoire et le calcul de structures en fonction du facteur de charge, *etc.*),
- R : fonctions de résidus, servant à évaluer la satisfaction des équations d'état.

Remarque E.1.1. *Les équations d'état peuvent être formulées de manière implicite ou explicite. Ce dernier cas permet de trouver analytiquement les valeurs des x telles que les équations* (E.4) *sont satisfaites. Dans ce cas, les équations* (E.4) *se réécrivent :*

$$x_i = X_i(y_i, z_i) \tag{E.5}$$

avec X_i les fonctions de calcul des variables d'état.

Les équations disciplinaires peuvent être traitées de deux manières : sous la forme d'analyse disciplinaire ou d'évaluation disciplinaire. Etant données y et z, l'analyse disciplinaire consiste à faire évoluer x de manière à satisfaire les équations (E.4). L'évaluation disciplinaire consiste à calculer R en fonction de x, y et z.
La différence fondamentale entre analyse et évaluation disciplinaires réside dans la gestion des variables d'état x. En effet, l'analyse disciplinaire met en œuvre un processus itératif (sauf dans le cas où les variables d'états sont définies de manière explicite) afin de résoudre les équations d'états (E.4), ce qui induit le plus souvent l'utilisation d'un solveur spécifique (*e.g.* algorithme de Newton, *etc.*). Par contre, l'évaluation disciplinaire se contente d'effectuer un simple calcul de la valeur du résidu, sans chercher à résoudre les équations d'état (E.4).

Concepts de faisabilité

Definition E.1.1. *Faisabilité disciplinaire*
Un processus est qualifié de "faisable disciplinairement " si, à chaque itération, les équations d'états de chacune des disciplines sont satisfaites : on peut trouver les variables x telles que les équations (E.4) *soient satisfaites.*

Definition E.1.2. *Faisabilité multi disciplinaire*
Un processus est qualifié de "faisable multi disciplinairement " si, à chaque itération, les équations d'états de chacune des disciplines sont satisfaites et les couplages sont cohérents : on peut trouver les variables x et y telles que les équations (E.3) *et* (E.4) *soient satisfaites.*

Definition E.1.3. *Analyse multi disciplinaire*
L'"analyse multidisciplinaire" (MDA) est un processus qui a pour but de garantir la faisabilité multidisciplinaire du système. Celle-ci consiste à trouver, pour chaque sous-système, les variables x et y telles que les équations (E.3) *et* (E.4) *soient satisfaites.*

Le tableau E.1 résume les différents concepts de faisabilité et la gestion des variables x, y et z.

Concepts	Processus utilisé au niveau sous-système	Variables gérées par les sous-systèmes	Variables gérées en dehors des sous-systèmes
Aucune faisabilité garantie	Evaluation disciplinaire		x,y,z
Faisabilité disciplinaire	Analyse disciplinaire	x	y,z
Faisabilité multidisciplinaire	MDA	x,y	z

Table E.1: Différents concepts de faisabilité et variables associées

E.1.2 Description du problème de conception de lanceurs

Les différentes disciplines entrant en jeu dans la conception de lanceurs sont l'aérodynamique, la propulsion, le calcul structural, le dimensionnement, l'estimation des coûts, le calcul de la trajectoire (calcul de performances), *etc.* Chacune des disciplines met en jeu ses propres variables et contraintes. Les variables de conception z telles que les masses, les diamètres, les variables de propulsion, *etc.* sont généralement traitées au niveau système. Les variables de trajectoire sont classiquement considérées comme des variables d'état x. Les conditions d'optimalité de la trajectoire peuvent être gérées soit au niveau sous-système ($R = 0$) soit au niveau système ($R = 0$ sont alors adjointes aux contraintes d'égalité du niveau système). Les contraintes d'égalité du problème peuvent être composées des spécifications de la mission (orbite désirée, masse utile à injecter, *etc.*) et les contraintes d'inégalité peuvent porter sur la pression de chambre maximale, le facteur de charge maximal, la pression de sortie de tuyère minimale, *etc.* Les fonctions objectif sont le plus souvent la maximisation de la masse de charge utile, la minimisation de la masse totale au décollage ou encore la minimisation d'un critère de coût.

La conception de lanceurs est un problème MDO spécifique car elle met en jeu les optimisations couplées des variables de conception et d'une loi de commande. L'optimisation de la trajectoire induit une simulation de la trajectoire avec séparations d'étages qui est complexe à intégrer (résolution numérique d'un système d'équations différentielles ordinaires fortement non linéaires). De plus, la trajectoire est soumise à des contraintes d'égalité strictes dûes aux spécifications de la mission, qui sont très difficiles à satisfaire et réduisent fortement le domaine de recherche faisable (domaine de recherche pour lequel toutes les contraintes sont satisfaites).

E.1.3 Méthodes MDO classiques

Dans cette sous-section, nous décrivons quelques méthodes MDO principales [Balesdent et al., 2011d]. Dans la littérature, on peut trouver quelques articles [Balling and Sobieszczanski-Sobieski, 1994; Alexandrov and Hussaini, 1995; Sobieszczanski-Sobieski and Haftka, 1997; Agte et al., 2009] dressant un état de l'art exhaustif des différentes méthodes MDO existantes. Le but de cette partie n'est pas d'expliquer toutes les méthodes MDO mais de seulement décrire les méthodes principales et de les comparer relativement à leur application dans la LVD. Les méthodes MDO peuvent généralement être réparties en deux catégories, suivant la présence d'un ou de plusieurs niveaux d'optimisation.

Méthodes mono niveau

On distingue principalement trois méthodes mono niveau. Celles-ci sont les méthodes "Multi-Discipline Feasible" (MDF), "Individual Discipline Feasible" (IDF) et "All At Once " (AAO).

Méthode MDF

La méthode MDF met en jeu un optimiseur au niveau système qui gère toutes les variables de conception z et est soumis aux contraintes (E.1) et (E.2). Les variables de couplage y et d'état x sont gérées au niveau sous-système par une MDA qui assure la faisabilité multidisciplinaire à chaque itération du processus d'optimisation. Le principal avantage de cette méthode réside dans le fait qu'un nombre limité de variables est géré par l'optimiseur (seulement les variables de conception z). Le principal inconvénient de cette méthode est le temps de calcul important. Celui-ci est principalement dû à la MDA qui doit être effectuée à chaque itération.

Méthode IDF

La méthode IDF permet d'éviter l'exécution de la MDA à chaque itération. Comme MDF, IDF utilise un seul optimiseur au niveau système mais emploie des analyses disciplinaires au niveau sous-système. La principale différence avec MDF et que dans IDF, l'optimiseur est aussi responsable de la gestion des variables y. Ainsi, la méthode IDF requiert des contraintes de couplage qui sont adjointes aux contraintes initiales du problème d'optimisation (E.1), (E.2) et (E.3). A chaque itération du processus d'optimisation, les sous-systèmes sont disciplinairement faisables mais la faisabilité multidisciplinaire n'est assurée qu'à la convergence de l'algorithme. La méthode IDF augmente sensiblement la taille du problème d'optimisation à résoudre au niveau système mais permet d'effectuer en parallèle les analyses disciplinaires.

Méthode AAO

La méthode AAO résout simultanément le problème d'optimisation et les équations disciplinaires. En effet, dans cette méthode, l'optimiseur gère à la fois les variables de conception z, de couplage y et d'état x. De plus, les contraintes sur les résidus sont adjointes au problème d'optimisation. Cette méthode utilise au niveau sous-système des évaluations disciplinaires. Au niveau système, l'optimiseur doit satisfaire les contraintes (E.1), (E.2), (E.3) et (E.4). Par conséquent, pour cette méthode, les faisabilités disciplinaires et multidisciplinaires ne sont satisfaites qu'à la convergence de l'algorithme.
Le tableau E.2 [Cramer et al., 1994] résume différentes caractéristiques des méthodes MDF, IDF et AAO.

	AAO	**IDF**	**MDF**
Utilisation des outils d'analyse traditionels	Non	Oui	Oui, couplage requis
Faisabilité disciplinaire de la solution	Seulement à la convergence	A chaque itération	A chaque itération
Faisabilité multidisciplinaire de la solution	Seulement à la convergence	Seulement à la convergence	A chaque itération
Variables gérées par l'optimiseur	x, y, z	y, z	z
Vitesse de convergence espérée	Rapide	Moyenne	Lente

Table E.2: Comparaison qualitative des méthodes mono niveau

Méthodes multi niveau

Nous détaillons ici trois méthodes multi niveau classiques : la méthode "Collaborative Optimization" (CO), la méthode "Bi-Level Integrated System Synthesis" (BLISS) et la méthode

“Analytical Target Cascading” (ATC). Pour plus de détails sur les méthodes présentées, on pourra consulter [Balesdent et al., 2011d].

Méthode CO

La méthode CO implique deux niveaux d’optimisation et traite chaque sous-système de manière séparée. Dans cette méthode, chaque sous-système optimise ses propres variables de conception afin de trouver un accord avec les autres sous-systèmes au niveau des couplages. Pour ce faire, cette méthode met en jeu des fonctions objectif spécifiques au niveau sous-système qui symbolisent la satisfaction des instructions données par le niveau système. Ces fonctions objectif sont le plus souvent des sommes quadratiques d’erreurs. Différentes formulations de la méthode CO [Braun and Kroo, 1995; Alexandrov and Lewis, 2000] ont été proposées dans la littérature. Cette méthode permet une plus grande modularité par rapport aux méthodes mono niveau mais peut présenter des problèmes numériques à la convergence lorsqu’elle est associée à un algorithme basé sur un calcul de gradients [Alexandrov and Lewis, 2000].

Méthode BLISS

La méthode BLISS est une méthode itérative basée sur des calculs de dérivées. Cette méthode met en place un schéma itératif dans lequel des calculs de sensibilités des niveaux système et sous-système sont alternativement réalisés. En effet, cette méthode optimise alternativement les contributions des variables locales et partagées à la fonction objectif globale. Cette méthode a été améliorée pour l’utilisation des méta modèles (BLISS2000 [Sobieszczanski-Sobieski et al., 2003]). BLISS, lorsqu’elle est applicable, permet de trouver rapidement l’optimum. Mais, comme toutes les méthodes basées sur des calculs de dérivées, celle-ci a besoin de limites sur les bornes des variables d’optimisation pour converger.

Méthode ATC

La méthode ATC est une méthode multi niveau hiérarchique et permet de propager différentes fonctions objectif à travers les différents niveaux d’optimisation. A chaque niveau de l’optimisation, un problème spécifique est résolu. Celui-ci consiste le plus souvent en une minimisation de somme quadratique d’erreurs par rapport aux consignes données par le niveau supérieur. Cette méthode s’avère très efficace pour résoudre des problèmes présentant une décomposition en plusieurs niveaux (système, sous systèmes, disciplines, composants, *etc.*).

E.1.4 Synthèse comparative de l’application des méthodes MDO dans la conception de lanceurs

Cas d’application et algorithmes d’optimisation

Il apparaît que la méthode MDF est la plus utilisée dans la conception de lanceurs. En effet, plus de 80% des articles recensés dans la littérature utilisent cette méthode. Néanmoins, on peut trouver quelques articles traitant des autres méthodes (BLISS, AAO, IDF, *etc.*). Globalement, les algorithmes basés sur des calculs de gradients sont les plus utilisés. Néanmoins, un bon nombre d’articles utilisent des heuristiques (la plus utilisée étant l’algorithme génétique). Nous pouvons aussi trouver quelques articles [Akhtar and Linshu, 2005a; Geethaikrishnan et al., 2008] mettant en place des méthodes hybrides (couplage gradient et heuristique).

Coût calculatoire et vitesse de convergence

Les méthodes qui induisent une MDA sont celles qui présentent un temps de calcul le plus important. En effet, la MDA est un processus très coûteux, d'autant plus lorsqu'elle est associée à un algorithme d'optimisation basé sur des calculs de dérivées à partir de différences finies.

Découpler la MDA au niveau sous-système permet de réduire fortement le coût en calculs mais introduit plus de variables et de contraintes à gérer par l'optimiseur. Ainsi, si le problème met en jeu un nombre restreint de disciplines, et par conséquent de peu variables de couplage, les méthodes IDF et AAO sont celles qui convergent le plus rapidement [Martins and Marriage, 2007]. Dès que le problème devient plus complexe, AAO devient inapplicable car trop difficile à implémenter.

Une autre manière d'améliorer l'efficacité du processus consiste à utiliser une formulation bi-niveau. De cette manière, le problème principal peut être subdivisé en sous problèmes et des optimiseurs locaux peuvent être utilisés.

Nécessité d'utilisation d'algorithmes d'optimisation particuliers

Toutes les méthodes basées sur des schémas itératifs à base de calculs de sensibilités impliquent l'utilisation d'algorithmes à base de calculs de gradients (BLISS, CSSO, *etc.*) et par conséquent nécessitent d'une part des propriétés de convexité du problème MDO et d'autre part des bornes sur les variables d'optimisation afin de converger efficacement vers un optimum. Cette caractéristique est une différence majeure avec les autres méthodes MDO (*e.g.* MDF, IDF, CO, *etc.*) qui permettent d'utiliser des heuristiques comme les algorithmes génétiques et par conséquent ne requièrent pas *a priori* de propriétés de convexité du problème à résoudre.

Applicabilité à des systèmes de grande taille

Les formulations fortement centralisées (*e.g.* IDF, AAO, *etc.*) sont difficiles à appliquer à des systèmes de grande dimension. En effet, en raison de l'insertion d'un grand nombre de variables de couplage, ces méthodes sont difficiles à implémenter dans le cas de systèmes de grande taille et présentent certaines difficultés à converger. La méthode CO semble être bien adaptée pour les systèmes présentant un nombre raisonnable de variables de couplage mais son efficacité décroît rapidement dès que leur nombre augmente. BLISS, considérant les différentes disciplines comme des boîtes noires, est applicable à des systèmes de taille importante, même si elle requiert des analyses de sensibilités des sous systèmes, car ces dernières peuvent être effectuées en parallèle.

Conclusion et voies d'amélioration

Au regard de l'étude comparative des différentes méthodes MDO appliquées à la conception de lanceurs, nous identifions plusieurs voies d'amélioration possibles :

1. l'inclusion des aspects de stabilité dans le processus d'optimisation : cette caractéristique est primordiale, notamment dans les phases d'avant projet, si l'on veut garantir que la configuration trouvée restera valide dans les autres phases du processus de conception,

2. le couplage des heuristiques et des algorithmes basés sur des calculs de gradients,

3. la considération particulière de la trajectoire au sein du processus de conception : la conception de lanceurs est spécifique car elle mêle les optimisations couplées des variables de conception et d'une trajectoire. En ce sens, la trajectoire ne peut être considérée comme une discipline traditionnelle mais doit jouer un rôle prédominant dans le processus d'optimisation. Afin de placer la trajectoire au centre du processus de conception, l'élaboration d'une autre subdivision du processus de conception semble être une voie d'amélioration intéressante. En effet, tous les exemples trouvés dans la littérature considèrent une décomposition selon les disciplines. De manière à améliorer l'adaptabilité et la flexibilité de la méthode, une décomposition en étages serait mieux adaptée.

Ce dernier point est l'objet de la prochaine section. Le second point fera l'objet de la troisième section.

E.2 Formulations MDO utilisant une décomposition en étages

E.2.1 Introduction

Dans cette section, nous présentons la décomposition du problème de conception de lanceurs selon les différents étages. Cette décomposition est analogue à celle selon les phases de vol. La décomposition en phases de vol a été utilisée pour optimiser la trajectoire des lanceurs consommables [Beltracchi, 1992] et réutilisables [Rahn and Schöttle, 1994; Ledsinger and Olds, 2002; Perrot, 2003]. Les méthodes existantes consistent à optimiser séparément les différentes phases de vol afin de décomposer le problème initial en une série de problèmes élémentaires. Toutes ces études sont principalement centrées sur l'optimisation de trajectoire (avec occasionnellement quelques variables de conception concernant les masses), mais ne considèrent pas toutes les variables de conception et les couplages disciplinaires. Dans cette thèse, nous utilisons la décomposition en phases de vol afin de réaliser l'optimisation multidisciplinaire complète du lanceur. Nous ne considérons pas seulement la trajectoire mais toutes les disciplines (*e.g.* aérodynamique, propulsion, dimensionnement massique, structure, *etc.*) présentes dans le problème MDO.

Remarque E.2.1. *Les formulations MDO utilisant la décomposition en étages ont été établies dans un cadre général au chapitre 5 de la thèse. Dans ce résumé de la thèse, par soucis de concision, ces formulations seront exposées au travers du cas applicatif qui nous intéresse.*

E.2.2 Présentation des formulations SWORD

Principe

Afin d'utiliser l'architecture des lanceurs directement dans le processus de conception, quatre nouvelles formulations du problème sont proposées. Les formulations SWORD décomposent le problème d'optimisation selon les différentes phases de vol et optimisent chaque étage séparément. Dans le but de réduire la complexité du problème de conception de lanceurs, ces formulations transforment le problème MDO global en la coordination de problèmes plus faciles à résoudre.

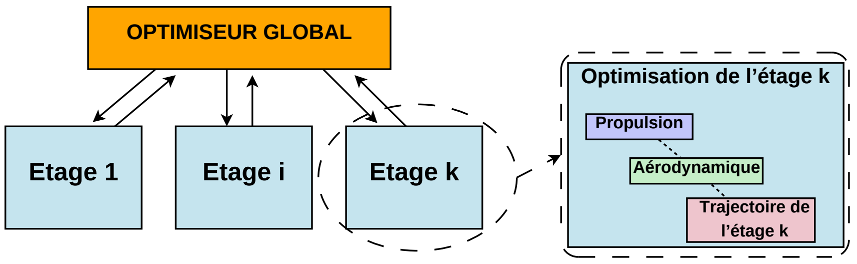


Figure E.1: Principe général de SWORD

Coordination des optimisations des étages

Afin de décomposer le problème MDO global selon les étages, nous devons insérer les variables de couplage suivantes (et leurs contraintes respectives) au problème MDO initial :

- les vecteurs d'états (position et vitesse) aux séparations d'étages : ces variables sont requises pour garantir la cohérence de la trajectoire,
- les estimations des masses d'étages : afin de simuler sa phase de vol, un étage doit connaître sa masse utile équivalente. Cette masse est la somme des masses des étages supérieurs, de la coiffe et de la charge utile.

Formulations proposées

Nous proposons dans cette thèse quatre formulations MDO bi-niveau dédiées à la conception de lanceurs. Ces formulations diffèrent dans la manière de traiter les couplages entre les différents étages. Les formulations mathématiques seront décrites directement en fonction du cas d'application dans la prochaine sous-section.

Première formulation

Dans la première formulation, chaque étage est optimisé de manière indépendante au niveau sous-système, optimise son propre objectif et est équivalent à un lanceur mono étage pour sa phase de vol élémentaire. Afin de coordonner les différents étages, l'optimiseur du niveau système gère les variables partagées, les vecteurs d'état à la séparation des étages et les masses estimées des étages supérieurs. Des contraintes d'égalité de couplage concernant les estimations des masses et les masses optimisées sont requises au niveau système.

Seconde formulation

La seconde formulation utilise l'optimisation collaborative (Collaborative Optimization [Braun and Kroo, 1995]) appliquée à la décomposition en étages. Dans cette formulation, chaque étage n'optimise pas son propre objectif mais minimise une somme quadratique d'erreurs relatives entre ses sorties et les instructions données par l'optimiseur du niveau système. Ce dernier contrôle la convergence de chaque optimisation d'étage (des contraintes d'égalité sont requises).

Troisième formulation

La troisième formulation réalise séquentiellement au niveau sous-système les optimisations des étages du dernier au premier. Dans cette formulation, chaque étage optimise sa propre fonction objectif et cherche à rejoindre le vecteur d'état défini par le niveau système. Cette formulation permet d'éviter l'utilisation des variables de couplage de masse car la cohérence massique est automatiquement assurée au niveau sous-système. Dans cette formulation, une fois que l'optimisation d'un étage est effectuée, la masse de l'étage considéré est transmise à l'étage inférieur qui peut ensuite effectuer son optimisation.

Quatrième formulation

La quatrième formulation est assez différente des trois premières. En effet, dans cette formulation, toutes les variables de conception sont gérées au niveau système. Les différents sous-systèmes gèrent uniquement les variables de trajectoire dans le but d'atteindre les vecteurs d'état définis au niveau système. Cette formulation est un compromis entre la méthode MDF (dans laquelle toutes les variables de trajectoire sont gérées au niveau système) et les trois premières formulations (dans lesquelles les variables de conception et de trajectoire sont réparties entre les différents étages).

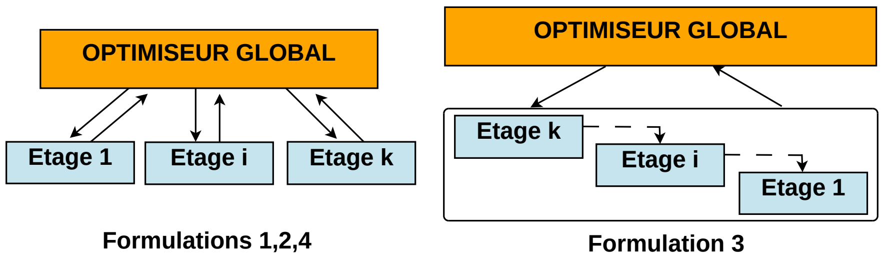


Figure E.2: Formulations SWORD : hiérarchique (gauche) et non hiérarchique (droite)

E.2.3 Présentation du cas d'application

Le problème MDO à résoudre consiste en l'optimisation d'un lanceur tri-étage en vue de minimiser sa masse totale au décollage (GLOW pour Gross-Lift-Off-Weight). La masse de charge utile à injecter en orbite de transfert géostationnaire est égale à 4 tonnes. Le lanceur est composé de 3 étages à propulsion cryotechnique. Les disciplines considérées sont la propulsion, le dimensionnement massique et géométrique, l'aérodynamique et la trajectoire (chargée du calcul de performances).

Modélisation
Afin d'éviter un temps de calcul trop important dans les analyses disciplinaires, nous avons choisi d'utiliser des modèles simplifiés qui sont généralement valables dans les phases d'avant projet. Le module de propulsion calcule l'impulsion spécifique à partir de la pression de chambre Pc, de la pression en sortie de tuyère Pe et du rapport de mélange Rm. Le module d'aérodynamique calcule le coefficient de traînée à partir du nombre de Mach. Le coefficient de portance n'est pas considéré dans cette étude. L'optimisation de trajectoire est réalisée par une méthode directe. La loi de commande (assiette θ) est définie à partir de fonctions paramétriques linéaires par morceaux. Finalement, le module de dimensionnement massique et géométrique calcule la masse sèche du lanceur. Cette masse est la somme des différents composants des étages (turbopompes, réservoirs, système de pressurisation, *etc.*). Nous considérons un couplage entre trajectoire et établissement des masses à travers le facteur de charge longitudinal maximal (nf). (Fig. E.3).

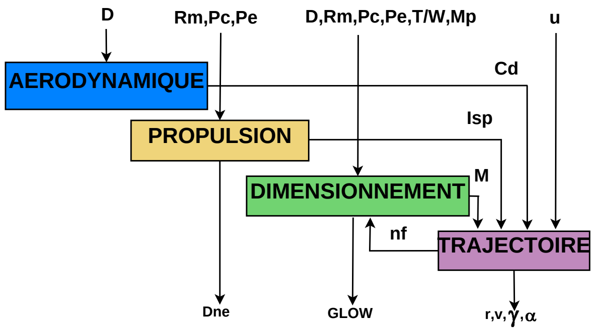


Figure E.3: Diagramme N2

Formulation mathématique du problème

La formulation du problème d'optimisation (MDF) est la suivante :

Minimiser $$f(z_d, y) = GLOW = \sum_{i=1}^{3}(Mp_i + Md_i) + M_{supérieure} \tag{E.6}$$

Par rapport à
$$\begin{aligned} z_{c_i} &= \{\theta_{ij}\} \quad i = 1, \cdots, 3, j = 1, \cdots, n_c \\ z_{d_i} &= \{D_i, Mp_i, Pc_i, Pe_i, Rm_i, T/W_i\} \quad i = 1, \cdots, 3 \\ y &= \{\widehat{n}_{f_{max}}\} \end{aligned}$$

Soumis à
$$h_1(z_c, z_d, y) : r_f - r_{orb} = 0 \tag{E.7}$$
$$h_2(z_c, z_d, y) : v_f - v_{orb} = 0 \tag{E.8}$$
$$h_3(z_c, z_d, y) : \gamma_f - \gamma_{orb} = 0 \tag{E.9}$$
$$c(z_c, z_d, y) : \widehat{n}_{f_{max}} - max(n_{f_{max}}, \widehat{n}_{f_{max}}) = 0 \tag{E.10}$$
$$g_{c1}(z_c, z_d, y) : |\alpha_{max1} - 4| \leq 0 \tag{E.11}$$
$$g_{c2}(z_c, z_d, y) : |\alpha_{max2} - 15| \leq 0 \tag{E.12}$$
$$g_{di1}(z_c, z_d, y) : 0.4Pa - Pe_i \leq 0 \tag{E.13}$$
$$g_{di2}(z_d) : Dne_i - 0.8D_i \leq 0 \tag{E.14}$$

avec f la fonction objectif, z_c les variables de commande, z_d les variables de conception, y les variables de couplage, h les contraintes d'égalité, g les contraintes d'inégalité, c les variables de couplage et pour l'étage i : Mp_i la masse d'ergols, Md_i la masse sèche, D_i le diamètre de l'étage, Pc_i la pression de chambre, Pe_i la pression en sortie de tuyère, Rm_i le rapport de mélange, Dne_i le diamètre de sortie de tuyère, T/W_i le rapport poussée/poids initial, θ_{ij} les variables de commande (assiette), r_f (resp. v_f, γ_f) l'altitude finale (resp. vitesse, pente finales), n_f le facteur de charge longitudinal maximal et α_{max_1} (resp. α_{max_2}) l'incidence maximale avant (resp. après) le largage de coiffe. Dans le but d'effectuer de manière séquentielle la MDA, une variable de couplage $\hat{n}_{f_{max}}$ et la contrainte de couplage associée ont été introduites dans le problème d'optimisation. Cette variable (facteur de charge longitudinal maximal estimé) garantit la cohérence du couplage entre la trajectoire et le dimensionnement massique.
Le problème MDO peut être écrit selon les notations génériques suivantes :

Minimiser $f(z_d, y)$ (E.15)
Par rapport à z_c, z_d, y

Soumis à
$$h(z_c, z_d, y) = 0 \quad \text{(E.16)}$$
$$c(z_c, z_d, y) = 0 \quad \text{(E.17)}$$
$$g_c(z_c, z_d, y) \leq 0 \quad \text{(E.18)}$$
$$g_d(z_c, z_d, y) \leq 0 \quad \text{(E.19)}$$

avec g_d (resp. g_c) les contraintes d'inégalité de conception (resp. de trajectoire).

Formulations SWORD

En utilisant les notations génériques détaillées précédemment, les différentes formulations proposées s'écrivent de la manière suivante :

Minimiser $f(z_{sh}, y_{F1})$ (E.20)
Par rapport à z_{sh}, y_{F1}
Soumis à $c_{0i_{F1}}(z_{sh}, y_{F1}) = 0$ (E.21)

Minimiser $f_i(z_{d_i}, z_{sh}, y_{F1})$ (E.22)
Par rapport à z_{d_i}, z_{c_i}
Soumis à
$$c(z_{c_i}, z_{d_i}, z_{sh}, y_{F1}) = 0 \quad \text{(E.23)}$$
$$c_i(z_{c_i}, z_{d_i}, z_{sh}, y_{F1}) = 0 \quad \text{(E.24)}$$
$$g_{ci}(z_{c_i}, z_{d_i}, z_{sh}, y_{F1}) \leq 0 \quad \text{(E.25)}$$
$$g_{di}(z_{d_i}, z_{sh}, y_{F1}) \leq 0 \quad \text{(E.26)}$$

PREMIERE FORMULATION

Minimiser $f(z_{sh}, y_{F2})$ (E.27)
Par rapport à z_{sh}, y_{F2}
Soumis à $c_{0i_{F2}}(z_{sh}, y_{F2}) = 0$ (E.28)

Minimiser $J_i(z_{c_i}, z_{d_i}, z_{sh}, y_{F2})$ (E.29)
Par rapport à z_{d_i}, z_{c_i}
Soumis à
$$c(z_{c_i}, z_{d_i}, z_{sh}, y_{F2}) = 0 \quad \text{(E.30)}$$
$$g_{ci}(z_{c_i}, z_{d_i}, z_{sh}, y_{F2}) \leq 0 \quad \text{(E.31)}$$
$$g_{di}(z_{d_i}, z_{sh}, y_{F2}) \leq 0 \quad \text{(E.32)}$$

SECONDE FORMULATION

```
Minimiser        $f(z_{sh}, y_{F3})$                      (E.33)
Par rapport à    $z_{sh}, y_{F3}$
```

```
$i = n$
Etant donné $f_{k,k>i}$
Tant que $i \neq 0$
    Minimiser        $f_i(z_{d_i}, z_{sh}, y_{F3})$                 (E.34)
    Par rapport à    $z_{d_i}, z_{c_i}$
    Soumis à         $c(z_{c_i}, z_{d_i}, z_{sh}, y_{F3}) = 0$      (E.35)
                     $c_i(z_{c_i}, z_{d_i}, z_{sh}, y_{F3}) = 0$    (E.36)
                     $g_{ci}(z_{c_i}, z_{d_i}, z_{sh}, y_{F3}) \leq 0$  (E.37)
                     $g_{di}(z_{d_i}, z_{sh}, y_{F3}) \leq 0$       (E.38)
$i \leftarrow i - 1$
```

TROISIEME FORMULATION

```
Minimiser        $f(z_d, z_{sh}, y_{F4})$                       (E.39)
Par rapport à    $z_d, z_{sh}, y_{F4}$
Soumis à         $c_{0i_{F4}}(z_d, z_{sh}, y_{F4}) = 0$         (E.40)
                 $g_{di}(z_{d_i}, z_{sh}, y_{F4}) \leq 0$       (E.41)
```

```
Minimiser        $R_i(z_{c_i} z_{d_i}, z_{sh}, y_{F4})$         (E.42)
Par rapport à    $z_{c_i}$
Soumis à         $c(z_{c_i}, z_{d_i}, z_{sh}, y_{F4}) = 0$      (E.43)
                 $g_{ci}(z_{c_i}, z_{d_i}, z_{sh}, y_{F4}) \leq 0$  (E.44)
```

QUATRIEME FORMULATION

avec :

$$c_{0i_{F1}} : \widehat{Mu}_{\text{étage } i} - Mu^*_{\text{étage } i} \tag{E.45}$$

$$c_{0i_{F2}} : J_i^* \tag{E.46}$$

$$c_{0i_{F4}} : R_i^* \tag{E.47}$$

$$c_{i_{1\ldots3}} : x_{f_{ij}} - \hat{x}_{ij}, j = 1, \cdots, 3^{1} \tag{E.48}$$

$$c : \widehat{n}_{f_{max}} - max(\widehat{n}_{f_{max}}, n_{f_{max}}) \tag{E.49}$$

$$g_{ci1} : |\alpha_{max_1} - 4°| \tag{E.50}$$

$$g_{ci2} : |\alpha_{max_2} - 15°| \tag{E.51}$$

$$g_{di1} : 0.4Pa - Pe_i \tag{E.52}$$

$$g_{di2} : Dne_i - 0.8D_i \tag{E.53}$$

$$f_i : Mp_i + Md_i \tag{E.54}$$

$$f : GLOW = \sum_{i=1}^{3} f_i^* + M_{supérieure} \tag{E.55}$$

$$J_i = \sum_{j=1}^{3} \left(\frac{x_{f_{ij}} - \widehat{x}_{ij}}{X_j} \right)^2 + \left(\frac{\widehat{Mu}_{\text{étage } i-1} - Mu^*_{\text{étage } i-1}}{M_i} \right)^2 \tag{E.56}$$

$$R_i = \sum_{j=1}^{3} \left(\frac{x_{f_{ij}} - \widehat{x}_{ij}}{X_j} \right)^2 \tag{E.57}$$

où $x_{f_i} = [rf_i, vf_i, \gamma f_i]$ dans l'équation (E.48) est le vecteur d'état. Dans l'équation (E.56), X_j, M_i sont des coefficients de normalisation et pour $i = 1$, $\widehat{Mu}_{\text{étage } i-1}$ est l'équivalent de $\widehat{GLOW}$ (masse totale au décollage). $\widehat{\chi}$ représente la valeur estimée de χ et χ^* représente la valeur optimisée de χ.

[1] c_i sont équivalent aux équations (E.16) pour i=3

Concernant les variables d'optimisation, nous avons :

$$z_{sh} = \{D_2, D_3\} \tag{E.58}$$

$$y_{F1} = \{\widehat{rf}_i, \widehat{vf}_i, \widehat{\gamma f}_i, \widehat{\alpha f}_i, \widehat{n}_{f_{max}}, \widehat{Mu}_{stage\ i}\} \quad i = 2,3 \tag{E.59}$$

$$y_{F2} = \{\widehat{rf}_i, \widehat{vf}_i, \widehat{\gamma f}_i, \widehat{\alpha f}_i, \widehat{n}_{f_{max}}, \widehat{Mu}_{stage\ i}, \widehat{GLOW}\} \quad i = 2,3 \tag{E.60}$$

$$y_{F3-4} = \{\widehat{rf}_i, \widehat{vf}_i, \widehat{\gamma f}_i, \widehat{\alpha f}_i, \widehat{n}_{f_{max}}\} \quad i = 2,3 \tag{E.61}$$

$$z_{d_i} = \{D_i, Pc_i, Pe_i, T/W_i, Mp_i, Rm_i\} \tag{E.62}$$

$$z_{c_i} = \{\theta_{ij}\} \tag{E.63}$$

$$z_d = \bigcup_{i=1}^{3} z_{d_i} \tag{E.64}$$

D_i est seulement considéré pour i=1 dans (E.62).

E.2.4 Comparaison des différentes formulations à MDF

Dimension de l'espace de recherche et nombre de contraintes

Une étude comparative portant sur la dimension de l'espace de recherche et le nombre de contraintes au niveau système a été menée. Celle-ci a permis d'identifier les formulations qui paraissent les plus avantageuses car permettant d'aboutir à un problème d'optimisation de moindre dimension et avec un nombre restreint de contraintes d'égalité au niveau système. En ce qui concerne ces deux caractéristiques, la troisième formulation paraît être la meilleure. En effet, cette formulation induit le plus petit espace de recherche (11 variables contre 13, 14, 27 et 31 pour les formulations 1, 2, 4 et MDF). De plus, cette formulation permet de s'affranchir totalement des contraintes d'égalité au niveau système, ce qui n'est pas le cas pour les autres formulations (Fig. E.4).

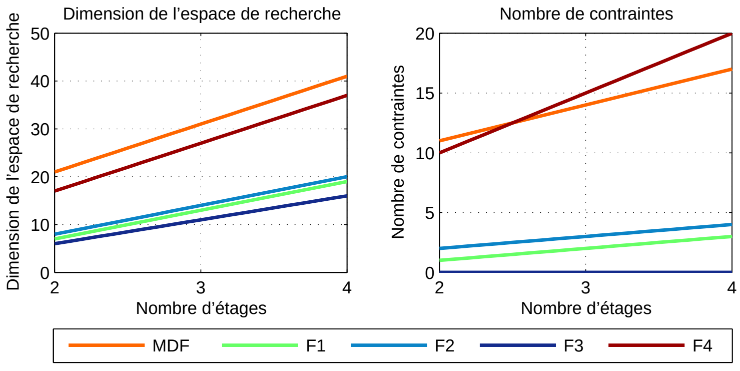


Figure E.4: Dimension de l'espace de recherche et nombre de contraintes

Comparaison numérique dans le cas d'une recherche globale

Les formulations proposées ont été comparées à la formulation MDF dans le contexte d'une recherche globale. Nous avons choisi de comparer les formulations SWORD à la formulation MDF car cette dernière est la plus utilisée dans la littérature. Les formulations ont été testées en utilisant le même algorithme d'optimisation, les mêmes initialisations et les mêmes réglages. L'algorithme d'optimisation sélectionné est l'algorithme génétique, car étant le plus répandu dans la conception de lanceurs dans le cas de recherches globales. La population initiale est composée de 100 individus. Dix tests ont été effectués à partir d'initialisations aléatoires afin de rendre la comparaison moins sensible à l'initialisation et à la part d'aléatoire de l'algorithme génétique. La comparaison entre les différentes formulations a été réalisée suivant trois critères : le temps écoulé pour trouver un premier design faisable à partir d'initialisation aléatoire, la capacité d'optimiser la fonction objectif et la qualité du meilleur point trouvé. La comparaison est effectuée en considérant un temps de calcul fixé (10 heures).

La figure E.5 indique les résultats moyens obtenus pour chacune des formulations. Le diagramme de gauche indique les meilleures solutions obtenues à l'arrêt de l'algorithme. Le diagramme central représente les temps de calcul moyens nécessaires pour trouver un premier concept faisable à partir d'une initialisation aléatoire. Finalement, le diagramme de droite indique l'amélioration de la fonction objectif durant le processus d'optimisation. La figure E.6 représente les performances des différentes formulations concernant la qualité du meilleur point obtenu et le temps nécessaire pour trouver un premier concept faisable.

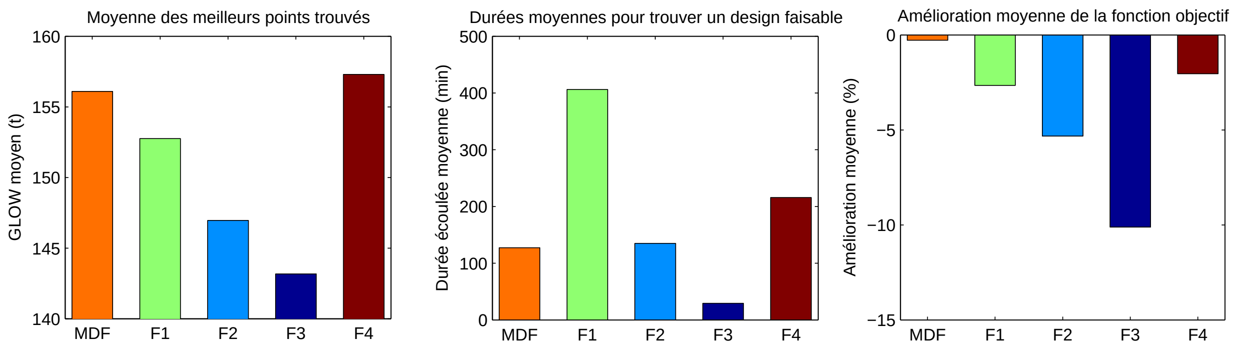


Figure E.5: Comparaison des méthodes dans le cas d'une recherche globale

La troisième formulation est celle qui aboutit aux meilleurs résultats. En effet, cette formulation permet d'obtenir la meilleure solution, est capable de trouver un premier design faisable dans le temps le plus faible et conduit à la meilleure amélioration de la fonction objectif. Ceci est principalement dû à la gestion des couplages massiques qui est effectuée au niveau sous-système, permettant d'éviter l'utilisation de contraintes d'égalité au niveau système et par conséquent permettant d'améliorer significativement l'efficacité de la recherche globale. De plus, cette méthode induit la plus faible dimension de l'espace de recherche au niveau système. Pour toutes ces raisons, nous avons choisi cette méthode afin de développer dans la prochaine section une stratégie d'optimisation dédiée à la décomposition en étages.

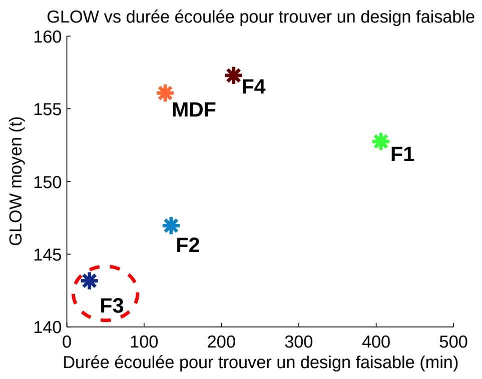


Figure E.6: Synthèse des résultats obtenus

E.2.5 Conclusion

La comparaison des méthodes a montré que l'utilisation d'une formulation SWORD appropriée permet d'améliorer significativement l'efficacité du processus d'optimisation par rapport à la méthode MDF, la plus utilisée dans la littérature. Néanmoins, même si cette formulation associée à un algorithme génétique permet de trouver relativement rapidement un design faisable et de l'améliorer significativement, le processus d'optimisation ainsi formé présente une vitesse de convergence relativement faible. De plus, cette formulation, incluant une phase d'optimisation au niveau sous système, peut poser des problèmes de temps de calcul lorsqu'elle est utilisée avec un algorithme génétique qui requiert par essence un grand nombre d'appels aux sous systèmes afin d'optimiser la fonction objectif. Pour ces raisons, il apparaît essentiel de développer une stratégie d'optimisation afin d'exploiter les spécificités de la décomposition en étages directement dans le processus d'optimisation. Ceci est l'objet de la prochaine section.

Remarque E.2.2. *Au vu des résultats obtenus dans cette section, nous utiliserons la troisième formulation SWORD dans la section suivante afin d'établir la stratégie d'optimisation.*

E.3 Stratégie d'optimisation dédiée à la décomposition en étages

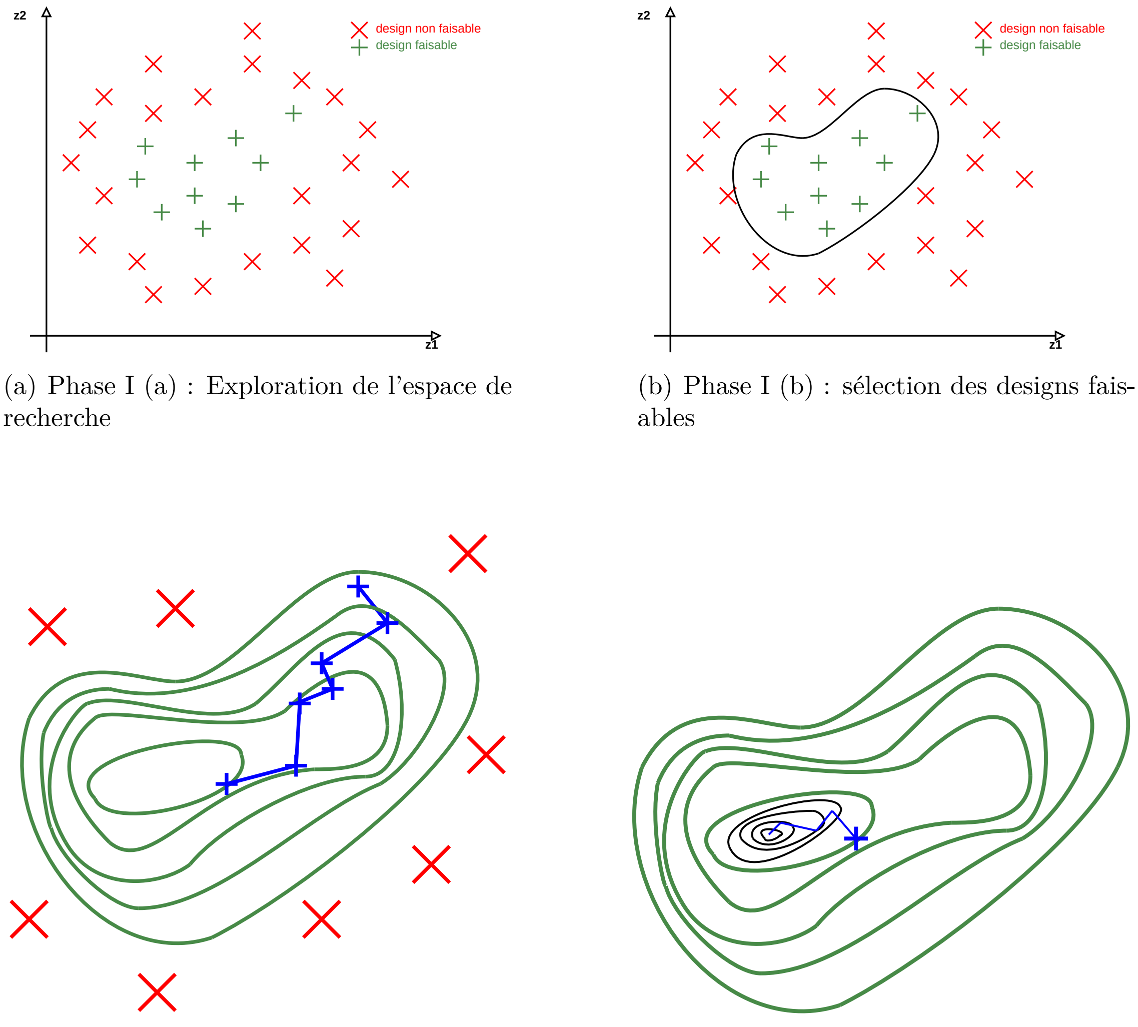
E.3.1 Identification des besoins

Le choix de la stratégie d'optimisation est un point clé dans le processus de conception de lanceurs. En effet, à la fois les recherches globales et locales sont requises afin d'explorer un large espace de recherche (bornes sur les variables d'optimisation) et de converger efficacement vers un optimum. Nous avons pris en compte les cinq exigences suivantes pour l'élaboration de la stratégie d'optimisation :

1. trouver rapidement des designs faisables (*i.e.* satisfaisant les contraintes de conception et de couplage), à partir d'initialisation aléatoire,
2. une fois le domaine de recherche faisable atteint, converger rapidement vers un optimum,
3. être robuste vis-à-vis de l'initialisation (stabilité des résultats),
4. converger sur des domaines de recherche très importants,
5. limiter au maximum l'intervention de l'utilisateur.

Afin de satisfaire toutes ces exigences, nous proposons une stratégie d'optimisation en 3 phases. Dans la première phase, nous exploitons la décomposition en étages pour effectuer une exploration séquentielle des différentes variables du niveau système. Une fois des designs faisables trouvés, une première étape d'optimisation (phase II) est effectuée en utilisant l'algorithme de Nelder & Mead (ou l'algorithme EGO si l'espace de recherche présente des optima locaux), dans le but d'atteindre un point proche de l'optimum. Dans la seconde étape de l'optimisation (phase III), nous effectuons une optimisation par un algorithme à base de calculs de gradients pour converger rapidement vers l'optimum.

Remarque E.3.1. *Dans cette partie, nous ne décrivons que les algorithmes utilisés au niveau système. En ce qui concerne le niveau sous-système, puisque les problèmes sont relativement faciles à résoudre, nous utilisons un algorithme SQP (Sequential Quadratic Programming) standard dans toutes les phases du processus d'optimisation.*



(a) Phase I (a) : Exploration de l'espace de recherche

(b) Phase I (b) : sélection des designs faisables

(c) Phase II : première optimisation

(d) Phase III : seconde optimisation

Figure E.7: Stratégie d'optimisation

E.3.2 Description de la stratégie d'optimisation

Phase I : exploration

La phase d'exploration consiste à utiliser la décomposition en étages afin de subdiviser l'ensemble des variables d'optimisation du niveau système (Fig. E.8) et de réduire l'exploration globale à une suite d'explorations élémentaires. A cette fin, l'ensemble des variables d'optimisation du niveau système est divisé en sous-ensembles élémentaires regroupant les variables intervenant dans chacune des phases de vol. Par exemple, pour la configuration triétage du lanceur, les variables d'optimisation du niveau système peuvent être réparties en deux sous-ensembles $z_{sh_2} \cup y_{12}$ et $z_{sh_3} \cup y_{23}$ où :

$$z_{sh_2} = \{D_2\} \tag{E.65}$$

$$z_{sh_3} = \{D_3\} \tag{E.66}$$

$$y_{12} = \{\widehat{r}_{f1}, \widehat{v}_{f1}, \widehat{\gamma}_{f1}, \widehat{\alpha}_{f1}\} \tag{E.67}$$

$$y_{23} = \{\widehat{r}_{f2}, \widehat{v}_{f2}, \widehat{\gamma}_{f2}, \widehat{\alpha}_{f2}, \widehat{n}_{f_{max}}\} \tag{E.68}$$

Le principe de l'exploration est le suivant :

1. explorer le sous-ensemble composé des variables d'optimisation intervenant dans la phase de vol du troisième étage (z_{sh_3} et y_{23}), et garder les valeurs pour lesquelles l'optimisation du troisième étage converge ($z^\circ_{sh_3}$ et y°_{23}),

2. étant données $z^\circ_{sh_3}$, y°_{23} et f°_3 (masse optimisée du troisième étage), explorer le sous-ensemble composé des variables d'optimisation intervenant dans les phases de vol des premier et second étages (z_{sh_2} et y_{12}) et garder les valeurs pour lesquelles les optimisations de ces étages convergent.

Ainsi, l'ensemble des variables faisables du niveau système est donné par :

$$z^\circ_{sh} = z^\circ_{sh_2} \cup z^\circ_{sh_3} \tag{E.69}$$

$$y^\circ = y^\circ_{12} \cup y^\circ_{23} \tag{E.70}$$

Cette méthode d'exploration permet de réduire le coût calculatoire par rapport à une exploration aléatoire standard (exploration simultanée de toutes les variables du niveau système), car l'exploration globale est réduite à une série d'explorations élémentaires et le processus mis en place assure la cohérence des configurations des étages entre les différentes phases d'exploration.

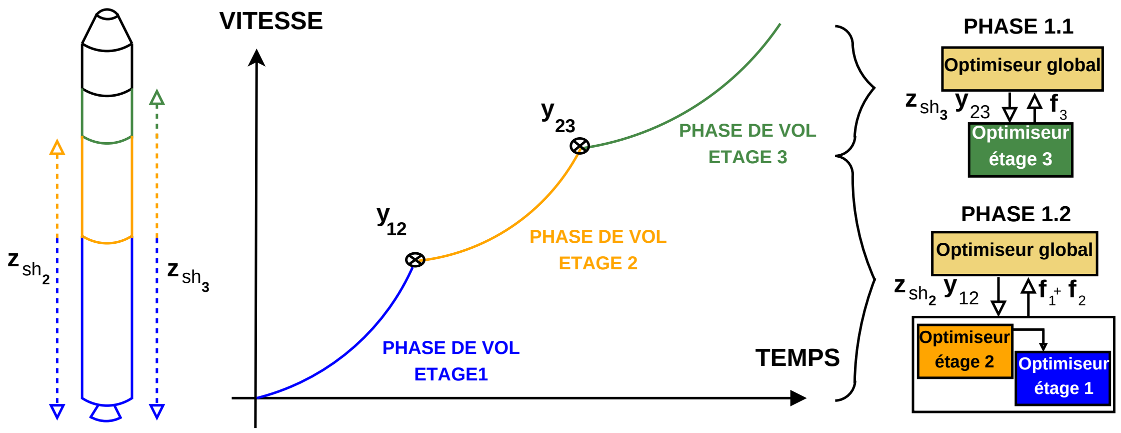


Figure E.8: Stratégie d'exploration

Phase II : première étape de l'optimisation

A partir des points faisables trouvés lors de la phase d'exploration, une première phase d'optimisation à partir d'algorithmes ne nécessitant pas de calculs de dérivées est réalisée. Pour ce faire, nous utilisons l'algorithme de Nelder & Mead [Nelder and Mead, 1965] ou l'algorithme EGO [Jones, 2001] si l'espace de recherche est trop irrégulier (présence d'optima locaux). Nous avons choisi d'utiliser en priorité l'algorithme de Nelder & Mead car celui-ci ne nécessite pas de réglages particuliers de l'utilisateur. Cet algorithme requiert de trouver $n+1$ points faisables, où n représente la dimension de l'espace de recherche au niveau système. Cette phase a pour but de trouver un point dans le voisinage de l'optimum. Ce point sera utilisé comme initialisation pour l'algorithme utilisé dans la phase 3.

Phase III : seconde étape de l'optimisation

Puisque la formulation SWORD proposée comporte deux niveaux d'optimisation, un algorithme de type gradient au niveau système requiert des calculs de dérivées qui nécessitent autant de réoptimisations des sous-systèmes qu'il y a de variables de conception au niveau système (calcul par différences finies). Ceci induit un coût en calculs important et réduit fortement l'efficacité du processus d'optimisation. Une solution possible pour surmonter cette difficulté consiste à utiliser l'analyse post-optimale (POA pour Post-Optimality Analysis) afin d'estimer le gradient de la fonction objectif à partir des informations de convergence données par le niveau sous-système. Puisque les optimisations des différents sous-systèmes sont effectuées de manière séquentielle, nous utilisons l'équation de sensibilité globale (GSE pour Global Sensitivity Equation) au niveau système. L'analyse post-optimale permet d'estimer le gradient de la fonction objectif au niveau système f à partir des multiplicateurs de Lagrange associés aux conditions de Karush-Kuhn-Tucker des différents sous-systèmes [Sobieszczanski-Sobieski, 1990; Braun et al., 1993]. Au moyen de ces deux techniques d'optimisation, le gradient de la fonction objectif du niveau système est estimé sans avoir recours à des réoptimisations de sous-systèmes (Eq. E.71).

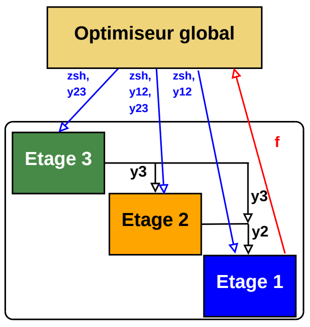


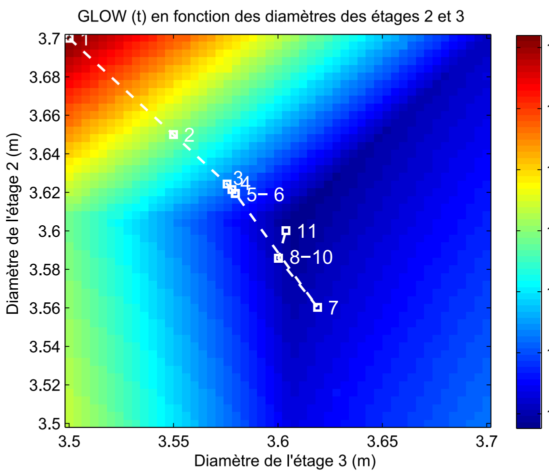
Figure E.9: Processus d'optimisation en phase 3

$$
\begin{bmatrix} \frac{dy_3}{dz_{sh}} \\ \frac{dy_3}{dy_{23}} \\ \frac{dy_3}{dy_{12}} \\ \frac{dy_2}{dz_{sh}} \\ \frac{dy_2}{dy_{23}} \\ \frac{dy_2}{dy_{12}} \\ \frac{df}{dz_{sh}} \\ \frac{df}{dy_{23}} \\ \frac{df}{dy_{12}} \end{bmatrix}
=
\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-\frac{\partial y_2}{\partial y_3} - \lambda_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{\partial y_2}{\partial y_3} - \lambda_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{\partial y_2}{\partial y_3} - \lambda_2^T \frac{\partial e_2}{\partial y_3} & 0 & 0 & 1 & 0 & 0 & 0 \\
-\frac{\partial f}{\partial y_3} - \lambda_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \lambda_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 & 0 & 0 \\
0 & -\frac{\partial f}{\partial y_3} - \lambda_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \lambda_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1 & 0 \\
0 & 0 & -\frac{\partial f}{\partial y_3} - \lambda_1^T \frac{\partial e_1}{\partial y_3} & 0 & 0 & -\frac{\partial f}{\partial y_2} - \lambda_1^T \frac{\partial e_1}{\partial y_2} & 0 & 0 & 1
\end{bmatrix}^{-1}
\cdot
\begin{bmatrix} \frac{\partial y_3}{\partial z_{sh}} + \lambda_3 \frac{\partial e_3}{\partial z_{zh}} \\ \frac{\partial y_3}{\partial y_{23}} + \lambda_3 \frac{\partial e_3}{\partial y_{23}} \\ \frac{\partial y_3}{\partial y_{12}} + \lambda_3 \frac{\partial e_3}{\partial y_{12}} \\ \frac{\partial y_2}{\partial z_{sh}} + \lambda_2 \frac{\partial e_2}{\partial z_{zh}} \\ \frac{\partial y_2}{\partial y_{23}} + \lambda_2 \frac{\partial e_2}{\partial y_{23}} \\ \frac{\partial y_2}{\partial y_{12}} + \lambda_2 \frac{\partial e_2}{\partial y_{12}} \\ \frac{\partial f}{\partial z_{sh}} + \lambda_1 \frac{\partial e_1}{\partial z_{zh}} \\ \frac{\partial f}{\partial y_{23}} + \lambda_1 \frac{\partial e_1}{\partial y_{23}} \\ \frac{\partial f}{\partial y_{12}} + \lambda_1 \frac{\partial e_1}{\partial y_{12}} \end{bmatrix}
\tag{E.71}
$$

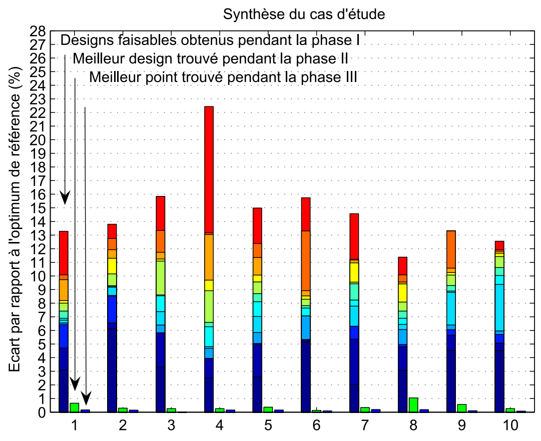
où λ_i sont les multiplicateurs de Lagrange associés au i^{ème} étage et e_i regroupent les contraintes d'égalité et celles d'inégalité saturées.

E.3.3 Analyse des performances de la stratégie d'optimisation

Afin d'évaluer les performances et la robustesse du processus d'optimisation, nous avons choisi d'effectuer 10 optimisations différentes, à partir de 10 recherches de points faisables réalisées par la phase I. Les domaines de recherche des variables du niveau système ont été choisis très importants de manière à estimer l'efficacité de la méthode sans connaissance *a priori* de l'utilisateur (exigence n° 3). A partir des designs faisables trouvés, nous avons réalisé 10 premières optimisations à l'aide de l'algorithme de Nelder & Mead puis nous avons achevé l'optimisation par l'algorithme basé sur des calculs de gradients.



(a) Evolution de la fonction objectif pendant la phase III



(b) Synthèse des 10 cas

Figure E.10: Comportement du processus d'optimisation

Dans tous les cas, le processus d'optimisation converge avec une différence relative maximale de 0.2% par rapport à l'optimum de référence (qui est obtenu en restreignant les domaines de recherche et en réglant finement les optimiseurs, ce qui nécessite une forte intervention de l'utilisateur et requiert beaucoup de temps). En effet, l'écart relatif entre le design obtenu et l'optimum de référence est réduit en moyenne de 96% (Fig. E.10). La durée totale moyenne du processus d'optimisation est de 4 heures (MATLAB, 2.4 GHz Dual Core Pentium / Windows XP). Afin de quantifier la robustesse de l'algorithme d'optimisation, définie ici comme la capacité à converger sur un espace de recherche très important, nous avons comparé la dispersion des 10 designs obtenus à celle des initialisations (faisables) trouvées après la première phase (Tab. E.11 et Fig. E.12). Les résultats montrent que la stratégie d'optimisation proposée est robuste car la dispersion des optima trouvés est seulement 4.44% de celle des initialisations.

Ecart maximum entre les différentes initialisations (meilleurs points de la phase I) : d_i	5.59t
Ecart maximum entre les optima trouvés : d_{opt}	0.25t
d_{opt}/d_i	4.4%

Figure E.11: Comparaison des dispersions des points avant et après optimisations

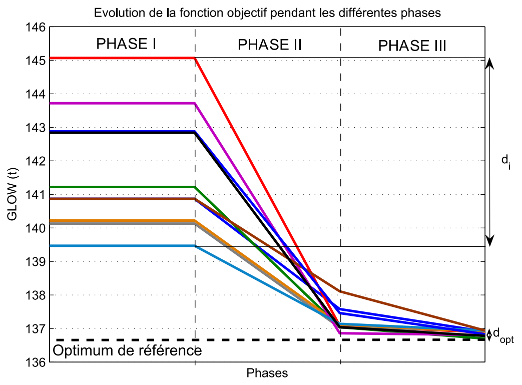


Figure E.12: Evolution de la fonction objectif pendant le processus d'optimisation

Afin de conclure sur l'efficacité de la stratégie d'optimisation proposée, nous nous proposons d'analyser les résultats obtenus par rapport aux critères formulés précédemment :

1. "trouver rapidement des designs faisables " : la recherche de solutions faisables est réalisée dans une phase spécifique. Les résultats numériques montrent que le processus proposé permet de trouver un design faisable en approximativement 5 minutes de calcul. Par conséquent, la recherche de l'initialisation de la phase II prend environ une heure, contre plus de 10 heures avec une recherche globale aléatoire. De ce point de vue, le processus proposé est relativement efficace et permet de satisfaire ce critère.

2. "être robuste vis-à-vis de l'initialisation (stabilité des résultats) " : afin de quantifier la robustesse du processus d'optimisation, nous avons réalisé 10 optimisations. Même si les designs trouvés après la première phase sont dispersés dans l'espace de recherche, les résultats obtenus après la phase III sont très proches. En effet, la dispersion des designs trouvés est seulement 4.44% de celle des designs faisables (trouvés après la phase I). Par conséquent, de ce point de vue, le processus satisfait le critère imposé.

3. "converger sur des domaines de recherche très importants" : un des principaux critères pour établir la stratégie d'optimisation est la capacité à travailler sur un espace de recherche très important (bornes sur les variables d'optimisation). De ce point de vue, le processus permet de travailler initialement sur un domaine très important et réduit l'espace de recherche durant le processus d'optimisation afin de mieux converger vers un optimum.

4. "une fois le domaine de recherche faisable atteint, converger rapidement vers un optimum" : la décomposition du processus d'optimisation en deux phases permet d'exploiter les avantages de chacun des algorithmes employés et par conséquent permet de converger avec un écart maximal de 0.2% de l'optimum de référence. Cette optimisation est réalisée en environ 3 heures, ce qui est très satisfaisant vis-à-vis du temps de calcul requis pour une recherche globale comme celle exposée dans la section précédente.

Conclusion

Cette thèse est centrée sur le développement de méthodes MDO consacrée aux études d'avant projet de conception de lanceurs. Notre contribution porte sur le développement de nouvelles formulations MDO, appelées SWORD (Stage-Wise decomposition for Optimal Rocket Design), exploitant une décomposition transversale du processus de conception selon les différentes phases de vol du lanceur. Les formulations SWORD permettent de répartir la complexité du problème entre les niveaux système et sous-système et de transformer le problème global d'optimisation en la coordination d'une série de sous-problèmes qui peuvent être résolus plus rapidement.

D'après l'étude bibliographique portant sur l'application des méthodes MDO existantes à la conception lanceurs, nous avons proposé une nouvelle formulation du problème de conception de lanceurs centrée sur l'optimisation de trajectoire et inspirée de l'architecture physique des lanceurs. A cette fin, nous avons développé quatre formulations MDO et les avons comparées à la méthode MDF dans le cas d'une recherche globale. Les résultats de cette étude ont montré que deux des formulations proposées (seconde et troisième) se détachent des autres et permettent d'améliorer l'efficacité du processus d'optimisation par rapport à la méthode MDO la plus utilisée dans la conception de lanceurs (MDF), en termes de temps de calcul pour trouver un design faisable à partir d'initialisation aléatoire et de qualité de la solution trouvée pour un temps de calcul donné. Par conséquent, avec la formulation appropriée, la décomposition en étage permet de rendre plus efficace l'utilisation d'algorithmes de recherche globale pour résoudre le problème de conception de lanceurs à partir d'initialisation aléatoire et sans requérir de connaissances spécifiques sur le domaine de variation des variables de conception. Néanmoins, cette étude a montré que l'utilisation de la recherche globale seule n'est pas suffisante pour obtenir un optimum en un temps de calcul acceptable (quelques heures). Par conséquent, le développement d'une stratégie d'optimisation spécifique est nécessaire.

Dans la seconde partie de cette thèse, nous avons proposé une stratégie d'optimisation dédiée aux formulations SWORD. A cette fin, nous avons utilisé la troisième formulation SWORD pour proposer une stratégie d'optimisation ayant pour but de rapidement trouver un point faisable, de converger vers un optimum en un temps de calcul limité, de réaliser l'optimisation sur des domaines de recherche très importants (bornes sur les variables d'optimisation) et d'améliorer la robustesse du processus vis-à-vis de l'initialisation. Afin de satisfaire ces exigences, nous avons proposé une optimisation en trois phases.

La première phase s'inspire de la troisième formulation SWORD et consiste à explorer séquentiellement les différents ensembles de variables du niveau système intervenant dans les différentes phases de vol. Cette méthode permet de réduire considérablement le temps de calcul pour trouver un design faisable par rapport à une recherche aléatoire globale.

A partir des designs faisables trouvés pendant la première phase, nous avons choisi d'effectuer l'optimisation en deux étapes. La première étape consiste en une optimisation sans calcul de gradient, utilisant l'algorithme de Nelder & Mead (ou l'algorithme Efficient Global Optimization si le problème présente beaucoup de minima locaux), dans le but de rapidement atteindre le voisinage d'un optimum. Finalement, à partir du meilleur design trouvé lors de la première étape d'optimisation, la seconde étape consiste à réaliser une optimisation à base de calculs de gradients, sur un espace de recherche limité autour de l'optimum. Afin de réduire le temps de calcul de cette phase, nous avons remplacé le calcul classique par différences finies du gradient de la fonction objectif du niveau système par une estimation exploitant les informations de convergence des sous-systèmes (Analyse Post Optimale et Equation de Sensibilité Globale).

Une analyse globale du processus complet (formulation MDO et stratégie d'optimisation) a ensuite été menée afin d'évaluer les performances de la méthode SWORD. Les résultats montrent que le processus d'optimisation permet d'atteindre l'optimum de référence à 0,2% près, en un temps de calcul inférieur à quatre heures, ce qui est satisfaisant pour les phases de conception avant projet, par comparaison avec les approches classiques qui prennent plus de dix heures de calculs lorsqu'elles prennent en compte un espace de recherche aussi important. Cette analyse a validé la stratégie d'optimisation proposée et a montré que l'utilisation de la décomposition en étages est utile afin de résoudre en temps de calcul limité le problème MDO de conception de lanceur sans requérir de connaissances *a priori* de l'utilisateur.

Afin d'améliorer la méthode MDO proposée, plusieurs extensions des travaux peuvent être proposées. Le processus de conception décrit est actuellement dédié aux phases d'avant projet dans lesquelles des modèles simplifiés sont utilisés. Il serait utile de tester la méthode MDO proposée avec des modèles disciplinaires plus détaillés, incluant davantage de couplages entre les disciplines.

De plus, dans cette thèse, nous avons comparé les méthodes SWORD avec la méthode MDF mais pas avec les autres méthodes MDO telles BLISS, ATC, CSSO, *etc.* La comparaison des méthodes SWORD avec ces méthodes serait très intéressante mais requerrait au préalable d'élaborer des modèles disciplinaires plus complexes afin de pouvoir appliquer les méthodes MDO présentes dans la littérature.

D'autre part, comme la méthode SWORD est principalement consacrée aux phases d'avant projet dans lesquelles des modèles simplifiés sont utilisés, il serait utile d'incorporer la gestion d'incertitudes à la méthode actuelle afin d'assurer la cohérence de la configuration trouvée dans les phases ultérieures du processus de conception.

Par ailleurs, la méthode de commande optimale développée dans cette thèse est une commande directe. Il serait intéressant d'étudier dans de futurs travaux d'autres méthodes d'optimisation de trajectoire (*e.g.* commande indirecte) afin de choisir laquelle est la plus adaptée au problème posé.

En outre, la méthode SWORD a été testée sur un problème de conception classique de lanceurs consommables avec étagement séquentiel (pas de boosters). Il serait intéressant de transposer et tester les formulations SWORD à des problèmes plus complexes comme les lanceurs avec étagement parallèle ou les lanceurs réutilisables. De la même manière, le processus actuel gère seulement les variables continues. L'introduction de variables discrètes serait profitable afin de considérer dans le processus de conception d'autres caractéristiques du lanceur telles le type de propulsion, le nombre d'étages, *etc.*

Enfin, dans cette thèse, nous avons appliqué la méthode SWORD à des problèmes mono objectif. Il serait très intéressant de tester notre méthode dans le cas de problèmes multi objectif (*e.g.* la maximisation de la charge utile et la minimisation du coût). Ce cas d'étude nous permettrait de concevoir une famille de lanceurs et d'être capable de proposer plusieurs choix de lanceurs possibles pour différentes missions.

Appendix F

Publications and communications

Publications in international journals

M. Balesdent, N. Bérend, Ph. Dépincé and A. Chriette. *A Survey of Mutidisciplinary Design Optimization methods in Launch Vehicle Design*, Structural and Multidisciplinary Optimization, Springer. DOI:10.1007/s00158-011-0701-4. accepted, pending publication

M. Balesdent, N. Bérend and Ph. Dépincé. *Multidisciplinary Design Optimization of a multi-stage launch vehicle using flight phases decomposition*, International Journal for Simulation and Multidisciplinary Design Optimization, EDP Sciences. accepted, pending publication

M. Balesdent, N. Bérend and Ph. Dépincé. *Optimal Design of Expendable Launch Vehicles using Stage-Wise Multidisciplinary Design Optimization Formulation*, Journal of Spacecraft and Rockets, AIAA. accepted, pending publication

Communications in international conferences

M. Balesdent, N. Bérend and Ph. Dépincé. *New MDO approaches for Launch Vehicle Design*, 4th European Conference for Aerospace Sciences, July 4 – 8 2011, St Petersburg, Russia.

M. Balesdent, N. Bérend and Ph. Dépincé. *MDO formulations and techniques adapted to multi-stage launch vehicle design*, 9th ISSMO World Congress on Structural and Multidisciplinary Optimization. June 13 – 17 2011, Shizuoka, Japan

M. Balesdent, N. Bérend and Ph. Dépincé. *Optimal Design of Expendable Launch Vehicles using Stage-Wise MDO Formulation*, 13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference. September 13 – 15 2010, Fort Worth, USA. Finalist of the "Best Student Paper Competition"

M. Balesdent, N. Bérend and Ph. Dépincé. *Multidisciplinary Design Optimization of a multi-stage launch vehicle using flight phases decomposition*, 3rd ASMDO International Conference on Multidisciplinary Design Optimization and Applications. June 21 – 23 2010, Paris, France

Bibliography

Agte, J. (2005). A Tool for Application of Bi-Level Integrated System Synthesis to Multi-disciplinary Design Optimization Problems. In *DGLR-2005-241, German Air and Space Congress, Friedrichshafen, Germany.*

Agte, J., de Weck, O., Sobieszczanski-Sobieski, J., Arendsen, P., Morris, A., and Spieck, M. (2009). MDO : assessment and direction for advancement - an opinion of one international group. *Structural Multidisciplinary Optimization, DOI10.1007/s00158-009-0381-5.*

Akhtar, S. and Linshu, H. (2005a). Simulation-based Optimization strategy for liquid fueled multi-stage space launch vehicle. In *6th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT'05). Dalian, China.*

Akhtar, S. and Linshu, H. (2005b). Support vector machine based trajectory metamodel for conceptual design of multi-stage space launch vehicle. In *Computational Intelligence and Security, vol 3801:528-535.* Springer Berlin / Heidelberg.

Akhtar, S. and Linshu, H. (2006). Support Vector Regression-driven Multidisciplinary Design Optimization for multi-stage launch vehicle considering throttling effect. In *44th AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada, USA.*

Alexandrov, N. and Hussaini, M. (1995). Multidisciplinary Design Optimization : State of the Art. *Proceedings of the ICASE / NASA Langley Workshop on Multidisciplinary Design Optimization, SIAM Proccedings Series.*

Alexandrov, N. and Lewis, R. (2000). Algorithmic Perspectives on Problem Formulations in MDO. In *8th AIAA/UASAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization. Long Beach, CA, USA.*

Allison, J. (2004). Complex system Optimization : a Comparison of Analytical Target Cascading, Collaborative Optimization and other formulations. Master's thesis, University of Michigan, USA.

Allison, J., Kokkolaras, M., Zawislak, M., and Papalambros, P. (2005). On the Use of Analytical Target Cascading and Collaborative Optimization for Complex System Design. In *6th World Congress on Structural and Multidisciplinary Optimization. 30 May-03 June, Rio de Janeiro, Brazil.*

Altus, T. (2002). A Response Surface Methodology for Bi-Level Integrated System Synthesis (BLISS). *NASA Technical Report n° NASA/CR-2002-211652. NASA Langley Research Center, USA.*

Alyaqout, S., Papalambros, P., and Ulsoy, A. (2005). Quantification and use of system coupling in decomposed design optimization problems. In *ASME International Mechanical Engineering Congress and Exposition. Orlando, FL, USA.*

Amouroux, F. (2008). Présentation des logiciels PERSEUS. *CNES Technical Report - PER-NT-3100000-4.*

Armijo, L. (1966). Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics, vol 16:1-3.*

Auroux, D., Clément, J., Hermetz, J., Masmoudi, M., and Parte, Y. (2009). *Etat de l'art et nouvelles tendances en conception collaborative. Optimisation multidisciplinaire en mécanique.* Hermes Science Publishing, ISTE Wiley. London, UK.

Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice.* Oxford University Press. UK.

Bäck, T., Fogel, D., and Michalewicz, Z. (1997). *Handbook of Evolutionary Computation.* Oxford University Press.

Bairstow, B., de Weck, O., and Sobieszczanski-Sobieski, J. (2006). Multiobjective Optimization of Two-Stage Tockets for Earth-To-Orbit Launch. In *47th AIAA/ASME/ASCE/AHS/ASC Structures, structural Dynamics and Materials Conference. Newport, RI, USA.*

Balesdent, M., Bérend, N., and Dépincé, P. (2010a). Optimal Design of Expendable Lauch Vehicles using Stage-wise MDO formulation. In *13th AIAA/ISSMO Multidisciplinary Analysis and Optimization conference. Fort Worth, Texas, USA.*

Balesdent, M., Bérend, N., and Dépincé, P. (2011a). MDO Formulations and Techniques adapted to Multi-stage Launch Vehicle Design. In *9th World Congress on Structural and Multidisciplinary Optimization, Shizuoka, Japan.*

Balesdent, M., Bérend, N., and Dépincé, P. (2011b). New MDO Approaches for Launch Vehicle Design. In *4th European Conference for AeroSpace Sciences, St. Petersburg, Russia.*

Balesdent, M., Bérend, N., and Dépincé, P. (2011c). Optimal Design of Expendable Lauch Vehicles using Stage-wise MDO formulation. *Journal of Spacecraft and Rockets, AIAA. publication pending.*

Balesdent, M., Bérend, N., Dépincé, P., and Chriette, A. (2010b). Multidisciplinary Design Optimization of a multi-stage launch vehicle using flight phases decomposition. In *3rd International Conference on Multidisciplinary Design Optimization and applications. Paris, France.*

Balesdent, M., Bérend, N., Dépincé, P., and Chriette, A. (2011d). A Survey of Multidisciplinary Design Optimization Techniques in Launch Vehicle Design. *Structural and Multidisciplinary Optimization, Springer. publication pending.*

Balesdent, M., Bérend, N., Dépincé, P., and Chriette, A. (2011e). Multidisciplinary Design Optimization of a multi-stage launch vehicle using flight phases decomposition. In *International Journal for Simulation and Multidisciplinary Design Optimization, EDP Sciences. publication pending.*

Balling, R. and Sobieszczanski-Sobieski, J. (1994). Optimization of coupled systems : a critical overview of approaches. *NASA / ICASE Report n° 94-100.*

Balling, R. and Wilkinson, C. (1997). Execution of Multidisciplinary Design Optimization approaches on common test problems. *AIAA Journal vol 35(1):178-186.*

Banach, S. (1922). Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae, vol 3:133-181.*

Bayley, D. and Hartfield, R. (2007). Design Optimization of Space Launch Vehicles for Minimum Cost Using a Genetic Algorithm. In *43rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit. Cincinnati, OH, USA.*

Bayley, D., Hartfield, R., Burkhalter, J., and Jenkins, R. (2008). Design Optimization of a Space Launch Vehicle Using a Genetic Algorithm. *Journal of Spacecraft and Rockets, vol 45(4):733-740.*

Bellman, R. (1957). *Dynamic Programming.* Princeton University Press. Princeton NJ.

Beltracchi, T. (1992). Decomposition Approach to Solving the All-Up Trajectory Optimization Problem. *Journal of Guidance, Control, and Dynamics, vol 15(3):707-716.*

Bérend, N. and Talbot, C. (1996). Overview of some optimal control methods adapted to expendable and reusable launch vehicle trajectories. *Aerospace Science and Technology, vol 10:222-232.*

Bertsekas, D. (2005). *Dynamic Programming and Optimal Control, 3rd Edition. Vol 1 & 2.* Athena Scientific.

Betts, J. (1998). Survey of Numerical Methods for Trajectory Optimization. *Journal of Guidance, Control, and Dynamics, vol 21(2):193-207.*

Boman, E. (1999). *Infeasibility and negative curvature in optimization.* PhD thesis, Stanford University, USA.

Bonnans, J., Gilbert, J., Lemaréchal, C., and Sagastizábal, C. (2006). *Numerical Optimization. Theoretical and Practical Aspects. Second Edition.* Springer.

Brauer, G., Cornick, D., and Stevenson, R. (1977). Capabilities and Applications of the Program to Optimize Simulated Trajectories (POST). *NASA Contractor Report - 2770.*

Braun, R. (1996). *Collaborative Optimization : An architecture for Large-Scale Distributed Design.* PhD thesis, Stanford University, USA.

Braun, R. and Kroo, I. (1995). Development and application of the Collaborative Optimization architecture in a multidisciplinary design environment. *Multidisciplinary Design Optimization : State of the Art, SIAM:98-116.*

Braun, R., Kroo, I., and Gage, P. (1993). Post-Optimality Analysis In Aerospace Vehicle Design. In *AIAA Aircraft Design, Systems and Operations Meeting. Monterey, CA, USA.*

Braun, R., Kroo, I., Lepsch, R., Powell, R., and Stanley, D. (1995). Comparison of two multidisciplinary optimization strategies for launch vehicle design. *Journal of Spacecraft and Rockets, vol 32(3):404-410.*

Braun, R., Moore, A., and Kroo, I. (1996). Use of the Collaborative Optimization Architecture for Launch Vehicle Design. In *6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. Bellevue, WA, USA.*

Briggs, G., Ray, T., and Milthorpe, J. (2007a). Evolutionay Algorithm use in Optimisation of a Launch Vehicle Stack Model. In *45th AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada, USA.*

Briggs, G., Ray, T., and Milthorpe, J. (2007b). Optimal design of an Australian medium launch vehicle. *Innovations in Systems and Software Engineering, vol 2(3):105-116.*

Brown, N. and Olds, J. (2005). Evaluation of Multidisciplinary Optimization (MDO) techniques applied to a reusable launch vehicle. In *43rd AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada, USA.*

Brown, N. and Olds, R. (2006). Evaluation of Multidisciplinary Optimization Techniques Applied to a Reusable Launch Vehicle. *Journal of Spacecraft and Rockets, 43:1289-1300.*

Bryson, A. (1999). *Dynamic Optimization.* Addison-Wesley Longman, Inc. Menlo-Park, California, USA.

Bryson, A. and Y-C., H. (1975). *Applied Optimal Control : Optimization, Estimation and Control, Revised Printing.* Taylor & Francis Inc. Levittown, USA.

Castellini, F. and Lavagna, M. (2009). Comparative analysis of multi-objective global optimization techniques for launchers design. In *8th World Congress on Structural and Multidisciplinary Optimization. Lisbon, Portugal.*

Castellini, F., Lavagna, M., and Erf, S. (2008). Concurrent optimization of launcher architectures and ascent trajectories with global optimization algorithms. In *59th International Astronautical Congress (IAC). Glasgow, Scotland.*

Castellini, F., Lavagna, M., Riccardi, A., and Büskens, C. (2010). Multidisciplinary Design Optimization Models and Algorithms for Space Launch Vehicles. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference. Fort-Worth, TX. USA.*

Castellini, F., Riccardi, A., Lavagna, F., and Büskens, C. (2011). Global and Local Multidisciplinary Optimization of Expendable Launch Vehicles. In *7th AIAA Multidisciplinary Design c Specialist Conference. Denver, CO. USA.*

Chachuat, B. (2006). *Non Linear and Dynamic Optimization, from Theory to Practice.* Ecole Polytechnique de Lausanne, IC-32 : Winter Semester 2006/2007.

Chandila, P., Perez, V., and Renaud, J. (2004). Post-Optimality Analysis for Multidisciplinary Systems using a Cumulative Response Surface Approximation. In *42nd AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nv, USA.*

Chelouah, R. and Siarry, P. (2003). Genetic and Nelder-Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research, vol 148:335-348.*

Chittick, I. and Martins, J. (2008). Aero-structural optimization using adjoint coupled post-optimality sensitivities. *Structural and Multidisciplinary Optimization, vol36:59-70.*

Clement, J. (2009). Optimisation multi-disciplinaire : étude théorique et application à la conception des avions en phase d'avant projet. *Institut Supérieur de l'Aéronautique et de l'Espace. Toulouse, France.*

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

Collange, G., Delattre, N., Lemaire, F., and Quinquis, I. (2009). Different approaches to multidisciplinary optimization for preliminary design of expendable launch vehicles. In *3rd European Conference for Aerospace Sciences. Versailles, France.*

Collange, G., Reynaud, S., and Hansen, N. (2010). Covariance matrix adaptation evolution strategy for multidisciplinary optimization of expendable launcher family. In *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference. Fort-Worth, TX.*

Cormier, T., Scott, A., Ledsinger, L., McCormick, D., Way, D., and Olds, R. (2000). Comparison of Collaborative Optimization to Conventional Design techniques for a Conceptual RLV. In *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. Long Beauch, CA, USA.*

Cramer, E., Dennis, J., Frank, P., Lewis, R., and Shubin, G. (1994). Problem Formulation for Multidisciplinary Optimization. *SIAM Journal of Optimization 4:754-776.*

Deb, K. (1995). *Optimization for engineering design: Algorithms and examples.* Prentice-Hall.

Deb, K. (2000). An Efficient Constraint Handling Method for Genetic Algorithms. *Computer Methods in Applied Mechanics and Engineering, vol 186(2-4):311-338.*

De Biasi, J.-M. (2002). Dossier de Définition du logiciel OPTAX 6.0.0. *Technical Repost CNES-SEGIME, SEG/MU/02-102.*

De Boor, C. (1978). *A Practical Guide to Splines.* Springer-Verlag, New-York.

Delattre, N. and Mongrard, O. (2007). The Use of Multidisciplinary Optimization for the Design of Expendable Launchers. In *2nd European Conference for AeroSpace Sciences (EUCASS). Brussels, Belgium.*

DeMiguel, A. (2001). *Two decomposition algorithms for nonconvex optimization problems with global variables.* PhD thesis, Stanford University, USA.

DeMiguel, V. and Murray, W. (2000). An Analysis of Collaborative Optimization Methods. In *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. Long Beach, CA, USA.*

DeMiguel, V. and Murray, W. (2006). A local convergence analysis of bilevel decomposition algorithms. *Optimization Engineering vol7:99-133.*

Dormand, J. and Prince, P. (1980). A family of embedded Runge-Kutta formulae. *Journal of Computational and Applied Mathematics, vol6:19-26.*

Du, X. and Chen, W. (2002). Efficient uncertainty analysis methods for multidisciplinary robust design. *AIAA Journal vol 40(3):545-552.*

Durand, N. and Alliot, J.-M. (1999). A Combined Nelder-Mead simplex and genetic algorithm. In *Genetic and Evolutionary Computation Conference. Orlando, Fl, USA.*

Duranté, N., Dufour, A., and Pain, V. (2004). Multidisciplinary Analysis and Optimisation Approach for the Design of Expendable Launchers. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference. Albany, New-York, USA.*

Duranté, N., Dufour, A., Ponçon, M., and Paulat, J. (2005). Integrated Multidisciplinary Optimisation of Expendable Launcher Design. In *1st European Conference for Aerospace Sciences. Moscow, Russia.*

Eiben, A., Hinterding, R., and Michalewicz, Z. (1999). Parameter Control in Evolutionary Algorithms. *IEEE Transactions on Evolutionary Algorithms, vol 3(2):124-141.*

Erb, S. (2007). *Development of Post-Optimality Analysis Algorithm for Optimal Control Problems.* PhD thesis, University of Stuttgart.

European Space Agency (2004). *ESA Launch Vehicle Catalogue, Revision 15.*

Filatyev, A. and Golikov, A. (2008). The complex Approach to Optimization of STS parameters on the Maximum Principle basis. In *59th International Astronautical Congress (IAC). Glasgow, Scotland.*

Filatyev, A., Golikov, A., Shanygin, A., and Voityshen, V. (2009). The local distributed criteria method for multidisciplinary optimization of launchers parameters. *Acta Astronautica 65:584-590.*

Fletcher, R. (1987). *Practical Methods of Optimization, 2nd edition.* John Wiley & Sons, Ltd.

Fraser-Andrews, G. (1999). A Multiple-Shooting Technique for Optimal Control. *Journal of Optimization theory and Applications, vol 102(2):299-313.*

Gang, C., ans W. Zi-ming, X. M., and Si-Lu, C. (2005). Multidisciplinary Design Optimization of RLV Reentry Trajectory. In *13th AIAA/CIRA International Space Planes and Hypersonic Systems and Technologies. Capua, Italy.*

Gath, P. and Well, K. (2001). Trajectory Optimization Using A Combination of Direct Multiple Shooting and Collocation. In *AIAA Guidance, Navigation, and Control Conference and Exhibit. Montreal, Canada. AIAA 2001-4047.*

Geethaikrishnan, C., Mujumbar, P., Sudhakar, K., and Adimurthy, V. (2008). Genetic Algorithm Guided Gradient Search for Launch Vehicle Trajectory Optimization. In *International Conference on Aerospace Science and Technology. Bangalore, India.*

Geethaikrishnan, C., Mujumdar, P., Sudhakar, K., and Adimurthy, V. (2009). A Robust and Efficient Hybrid Algorithm for Global Optimization. In *2009 IEEE International Advance Computing Conference (IACC 2009). Pariala, India.*

Gill, P., Murray, W., and Saunders, M. (1994). *Large-scale SQP Methods and their application in Trajectory Optimization.* Birkhäuser Verlag.

Gill, P., Murray, W., Saunders, M., and Wright, M. (1986). *Users Guide for NPSOL (Version 4.0) : A Fortran Package for Nonlinear Programming. Technical Report SOL 86-2.* Stanford University, USA.

Gill, P., Murray, W., and Wright, M. (1981). *Practical Optimization.* Academic Press, New-York, USA.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Professional.

Han, J. and Papalambros, P. (2010). A Note on the Convergence of Analytical Target Cascading With Infinite Norms. *Journal of Mechanical Design, vol132/034502.*

Hargraves, C. and Paris, S. (1987). Direct Trajectory Optimization Using Nonlinear Programming and Collocation. *Journal of Guidance, Control and Dynamics vol 10(4):338-342.*

Holland, J. (1975). *Adaptation in natural and artificial systems.* Ann Arbor : the University of Michigan Press. USA.

Huang, C. and Bloebaum, C. (2004). Multi-Objective Pareto Concurrent SubSpace Optimization for Multidisciplinary Design. In *42nd AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada, USA.*

Humble, R., Henry, G., and Larson, W. (1995). *Space Propulsion Analysis and Design.* Mc-Graw Hill.

Jodei, J., Ebrahimi, M., and Roshanian, J. (2006). An Automated Approach to Multidisciplinary System Design Optimization of Small Solid Propellant Launch Vehicles. In *1st ISSCAA International Symposium on Systems and Control in Aerospace and Astronautics. Harbin, China.*

Jodei, J., Ebrahimi, M., and Roshanian, J. (2009). Multidisciplinary design optimization of a small solid propellant launch vehicle using system sensitivity analysis. *Structural Multidisciplinary Optimization vol 38:93-100.*

Jones, D. (2001). A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization, vol 21:345-383.*

Jones, D., Schonlau, M., and Welch, W. (1998). Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization, vol 13:455-492.*

Kalden, O. (2007). Multidisciplinary Design and Trajectory Optimization of the Reusable Launch Vehicle Concept Hopper. In *3rd International Conference on Recent Advances in Space Technologies. Istanbul, Turkey.*

Kamm, Y., Horowitz, D., Brauner, N., and Gany, A. (2007). Design Optimization of a Solid Rocket Motor for a Suborital Space Flight. In *2nd European Conference for Aerospace Sciences. Brussels, Belgium.*

Keane, A. and Nair, P. (2005). *Computational Approaches for Aerospace Design : The Pursuit of Excellence.* John Wiley & Sons Ltd.

Keller, H. (1968). *Numerical Methods for Two-Point Boundary Value Problem.* Blaisdell Publishing Co. Waltham, Massachusetts.

Kim, H. (2001). *Target Cascading in Optimal System Design.* PhD thesis, University of Michigan, USA.

Kim, H., Chen, W., and Wiecek, M. (2006). *Lagrangian Coordination for Enhancing the Convergence of Analytical Target Cascading.* AIAA Journal vol 44(10):2197-2207.

Kleijnen, J. (2009). Kriging metamodeling in simulation : a review. *European Journal of Operational Research, vol 192(3):707-716.*

Kodiyalam, S. (1998). *Evaluation of Methods for Multidisciplinary Design Optimization (MDO), Phase 1.* NASA/CR-1998-20716.

Kodiyalam, S. (2000). *Evaluation of Methods for Multidisciplinary Design Optimization (MDO), Phase 2.* NASA/CR-2000-210313.

Kroo, I. (2004). Distributed Multidisciplinary Design and Collaborative Optimization. In *VKI lecture series on Optimization Methods and Tools for Multicriteria/Multidisciplinary Design.*

Kudryavtsev, L. (2001). Implicit function. *Encyclopaedia of Mathematics, Ed. Michiel Hazewinkel, Springer.*

Kuratani, N., Suzuki, H., and R.A.Goeklich (2005). Multidisciplinary Optimization of Space Transportation Systems. In *56th International Astronautical Congress of the International Astronautical Federation. Fukuoka, Japan.*

Kuri-Morales, A. and Gutierrez-Garcia, J. (2002). Penalty Function Methods for Constrained Optimization with Genetic Algorithms: A Statistical Analysis. *MICAI 2002: Advances in Artificial Intelligence. Lecture Notes in Computer Science, vol 2313/2002:187-200.*

Lagarias, J., Reeds, J., Wright, M., and Wright, P. (1998). Convergence properties of the Nelder - Mead simplex method in low dimensions. *Journal of Optimization, SIAM vol9(1):112-147.*

Lambe, A. (2011). A Unified Description of MDO Architectures. In *9th World Congress on Structural and Multidisciplinary Optimization, Shizuoka, Japan .*

Laurent-Varin, J., Bolanowski, O., Cristiani, E., and Zidani, H. (2009). Trajectory optimization of launcher with Hamilton-Jacobi-Bellman approach. In *3rd European Conference for Aerospace Sciences. Versailles, France.*

Ledsinger, L. and Olds, J. (2002). Optimized Solutions for Kistler K-1 Branching Trajectories Using Multidisciplinary Design Optimization Techniques. *Journal of Spacecraft and Rockets, vol 39(3):420-429.*

Lee, J. and Jong, H. (2006). A decomposition based design method coordinated by disciplinary subspace optimization. *JSME International Journal Series C, 49(3):935-941.*

Lee, J.-W., Choi, Y.-C., and Byun, Y.-H. (2005a). Optimal supersonic air-launching rocket design using multi-disciplinary system optimization approach. *Journal of the Korean Society Aeronautical and Space Sciences, vol 33:12.*

Lee, J.-W., Jeon, K.-S., Byun, Y.-H., and Kim, S.-J. (2005b). Optimal Space Launcher Design Using a Refined Response Surface Method. *Lecture Notes in Computer Science vol 3613/2005, Springer Berlin/Heidelberg.*

Lemaire, F., Collange, G., and Reynaud, S. (2010). GOAL : an industrial MDO platform used to design expendable launchers. In *3rd ASMO International Conference on Multidisciplinary Design Optimization and Applications. Paris, France, June 2010.*

Le Riche, R. and Haftka, R. (1994). Improved Genetic Algorithm for Minimum Thickness Composite Laminate Design. *Composites Engineering vol 3(1):121-139.*

Liu, X., Xiang, A., and Gang, T. (1997). A New Efficient Method of Multi-stage Solid Rocket Optimization. In *33rd AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit. Seattle, WA, USA.*

Luersen, M. and Le Riche, R. (2004). Globalized Nelder-Mead method for engineering optimization. *Computer & Structures, vol 82:2251-2260.*

Luersen, M., Le Riche, R., and Guyon, F. (2004). A constrained, globalized, and bounded Nelder-Mead method for engineering optimization. *Structural and Multidisciplinary Optimization vol 27:43-54.*

Luo, Y., Liang, Y., and Zhou, L. (2004). Multi-Level, Multi-Objective and Multidisciplinary optimization Design of a Series of Launch Vehicles. In *55th International Astronautical Congress. Vancouver, Canada.*

Martins, J. and Chittick, I. (2007). Subspace Optimization of Multidisciplinary Systems Using Coupled Post-Optimality Sensitivity Analysis. In *7th World Congress on Structural and Multidisciplinary Optimization,.*

Martins, J. and Marriage, C. (2007). An Object-Oriented Framework for Multidisciplinary Design Optimization. In *3rd AIAA Multidisciplinary Design Optimization Specialist Conference. Waikiki, Hawaii, USA.*

Marty, D. (1986). *Conception des véhicules spatiaux.* Masson.

Marzat, J., Walter, E., Piet-Lahanier, H., and Damongeot, F. (2011). Réglage automatique et comparaison de méthodes de diagnostique par Krigeage. In *4èmes journées Doctorales du GDR MACS. Marseille, France.*

Masmoudi, M. and Parte, Y. (2006). Discipline Interaction Variable Elimination (DIVE) approach for MDO. In *European Conference on Computational Fluid Dynamics (ECCOMAS CFD). Egmond aan Zee, The Netherlands.*

Masmoudi, M., Parte, Y., Hermetz, J., and Clement, J. (2008). DIVE : A New Approach for Multidisciplinary Optimization. In *5th European Congress on Computational Methods in Applied Sciences (ECCOMAS). Venice, Italy.*

Matheron, G. (1963). Principles of geostatistics. *Economy Geology, vol 58(8):1246-1266.*

Michalewicz, Z., Dasgupta, D., Le Riche, R., and Schoenauer, M. (1996). Evolutionary Algorithms for Constrained Engineering Problems. *Computers & Industrial Engineering Journal, vol 30(4):851-869.*

Michelena, N., Kim, H., and Papalambros, P. (1999). A system partitioning and optimization approach to target cascading. In *Proceedings of the 12th International Conference on Engineering Design. Munich, Germany.*

Michelena, N., Park, H., and Papalambros, P. (2003). Convergence properties of analytical target cascading. *AIAA Journal, vol41(5):897-905.*

Moore, A., Braun, R., and Powell, R. (1995). Optimal Technology Investment Strategies for a Reusable Launch Vehicle. In *AIAA Space Programs and Technologies Conference. Huntsville, AL, USA.*

NASA, NOAA, and USAF (1976). U.S. Standard Atmosphere, 1976. *U.S. Government Printing Office, Washington, D.C.*

Nelder, J. and Mead, R. (1965). A Simplex Method for Function Minimization. *Computer Journal, vol 7:308-313.*

Nocedal, J. and Wright, S. (2006). *Numerical Optimization, second edition.* Springer-Verlag.

Olds, J. (1992). The Suitability of selected Multidisciplinary Design and Optimization techniques to Conceptual Aerospace Vehicle Design. In *4th AIAA/USAF/NASA/OAI Symposium on Multidsiciplinary Analysis and Optimization. Cleveland, OH, USA.*

Olds, J. (1994). System Sensitivity Analysis Applied to the Conceptual Design of a Dual-Fuel Rocket SSTO. In *5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization. Panama City Beach, FL, USA.*

Olds, J. and Walberg, G. (1993). Multidisciplinary Design Of a Rocket-Based Combined Cycle SSTO Launch Vehicle using Taguchi Methods. In *AIAA/AHS/ASEE Aerospace Design Conference. Irvine, CA, USA.*

Orvosh, D. and Davis, L. (1993). Shall We Repair? Genetic Algorithms, Combinatorial Optimization and Feasibility Constraints. In *Proceedings of the Fifth International Conference on Genetic Algorithms, Urbana-Champaign, IL, USA.*

Pareto, V. (1971). *Manual of Political Economy.* A.M Kelley. New-york, NY, USA.

Perez, V., Renaud, J., and Watson, L. (2002). Adaptive Experimental Design for Construction of Response Surface Approximations. *AIAA Journal, vol 40(12):2495-2503.*

Perrot, L. (2003). Optimisation of reusable launchers by decomposition and coordination of trajectory optimisation problems. In *5th International Conference on Launcher Technology, Space Launchers : Missions, Control and Avionics. Madrid , Spain.*

Poles, S. (2003). MOGA-II, An Improved Multi-Objective Genetic Algorithm. *Technical Report 2003-006. Esteco, Trieste.*

Pontryagin, L., Boltyansky, V., Gamkrelidze, R., and Mischenko, E. (1962). *The Mathematical Theory of Optimal Processes.* Interscience Publishers. New-York, USA.

Qazi, M. and Linshu, H. (2006). Nearly-orthogonal sampling and neural network metamodel driven conceptual design of multistage space launch vehicle. *Computer-Aided Design, vol38:595-607, Elsevier.*

Qu, M., Gage, P., Sohn, J., Bradford, J., and Starr, B. (2004). Generalized vehicle performance closure model for two-stage-to-orbit launch vehicles. In *42nd AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada, USA.*

Rafique, A., Linshu, H., Kamran, A., and Zeeshan, Q. (2010). Multidisciplinary design of air launched satellite launch vehicle : Performance comparison of heuristic optimization methods. *Acta Astronautica, vol 67:826-844.*

Rafique, A., Linshu, H., Zeeshan, Q., Nisar, K., and Xiaowei, W. (2009a). Hybrid Optimization Method for Multidisciplinary Design of Air Launched Satellite Launch Vehicle. In *45th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit. Denver, Colorado, USA.*

Rafique, A., Linshu, H., Zeeshan, Q., Nisar, K., and Xiaowei, W. (2009b). Integrated System Design of Air Launched Small Space Launch Vehicle using Genetic Algorithm. In *45th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit. Denver, Colorado, USA.*

Rahn, M. and Schöttle, U. (1994). Decomposition Algorithm For Flight Performance And System Optimization of an Airbreathing Launch Vehicle. In *45th Congress of International Astronautical Federation. Jerusalem, Israel.*

Rao, A. (2009). A Survey of Numerical Methods for Optimal Control. In *2009 AAS/AIAA Astrodynamics Specialist Conference. Pittsburg, PA, USA.*

Renaud, J. and Gabriele, G. (1993). Improved Coordination in Nonhierarchic System Optimization. *AIAA Journal, vol 31(12):2367-2373.*

Renaud, J. and Gabriele, G. (1994). Approximation in Nonhierarchic System Optimization. *AIAA Journal, vol 32(1):198-205.*

Richardson, J., Palmer, M., Liepins, G., and Hilliard, M. (1989). Some Guidelines for Genetic Algorithms with Penalty Functions. In *Proceedings of the 3rd International Conference on Genetic Algorithms, Fairfax, VA, USA.*

Rodriguez, J., Perez, V., Padmanabhan, D., and Renaud, J. (2001). Sequential Approximate Optimization Using Variable Fidelity Response Surface Approximations. *Structural and Multidisciplinary Optimization, vol 22(1):24-34.*

Rodriguez, J., Renaud, J., and Watson, L. (1998). Trust Region Augmented Lagrangian Methods for Sequential Response Surface Approximation and Optimization. *Journal of Mechanical Design, vol 120:58-66.*

Rudolph, G. (2004). Convergence Analysis of Canonical Genetic Algorithms. *IEEE Transactions on Neural Networks, vol 5(1):96-101.*

Russell, R. and Shampine, L. (1972). A Collocation Methods for Bound-Value Problems. *Numerische Mathematik, vol 19:13-36.*

Santner, T., Williams, B., and Notz, W. (2003). *The design and analysis of computer experiments.* Springer-Verlag, Berlin-Heidelberg.

Sasena, M. (2002). *Flexibility and Efficiency Enhancements for Constraints Global Design Optimization with Kriging Approximations.* PhD thesis, University of Michigan, USA.

Schonlau, M. (1997). *Computer Experiments and Global Optimization.* PhD thesis, University of Waterloo, Canada.

Schoonover, P., Crossley, W., and Heister, S. (2000). Application of Genetic Algorithm to the Optimization of Hybrid Rockets. *Journal of Spacecraft and Rockets, vol37(5):622-629.*

Sellar, R. and Batill, S. (1996). Concurrent SubSpace Optimization Using Gradient-Enhanced Neural Network Approximations. In *6th AIAA/NASA/USAF Multidisciplinary Analysis and Optimization Symposium. Bellevue, WA, USA.*

Sellar, R., Batill, S., and Renaud, J. (1996). Response Surface Based, Concurrent SubSpace Optimization for Multidisciplinary System Design. In *34th AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada, USA.*

Sharapov, R. and Lapshin, A. (2006). Convergence of genetic algorithms. *Patter Recognition and Image Analysis, vol 16(3):392-397.*

Shin, M.-K. (2001). *Multidisciplinary design optimization based on independent subspaces.* PhD thesis, Hanyang University, Korea.

Shin, M.-K. and Park, G.-J. (2005). Multidisciplinary design optimization based on independent subspaces. *International Journal for Numerical Methods in Engineering, vol62:599-617.*

Sivaraj, R. and Ravichandran, T. (2011). A review of selection methods in Genetic Algorithms. *International Journal of Engineering Science and Technology, vol 3(5):3792-3797.*

Sobieski, I. and Kroo, I. (2000). Collaborative Optimization Using Response Surface Estimation. *AIAAJournal,vol38(10):1931-1938.*

Sobieszczanski-Sobieski, J. (1988a). *Optimization by Decomposition : A Step from Hierarchic to Non-Hierarchic Systems.* NASA Technical Report CP-3031.

Sobieszczanski-Sobieski, J. (1988b). *Sensitivity analysis and multidisciplinary optimization for aircraft design : recent advances and results.* NASA Technical Memorandum 100630.

Sobieszczanski-Sobieski, J. (1990). Sensitivity of Complex Internally Coupled Systems. *AIAA Journal, vol 28(1):153-160.*

Sobieszczanski-Sobieski, J., Agte, J., and Sandusky, R. (1998). *Bi-Level Integrated System Synthesis (BLISS).* NASA/TM-1998-208715.

Sobieszczanski-Sobieski, J., Altus, T., Phillips, M., and Sandusky, R. (2003). Bi-level Integrated System Synthesis for Concurrent and Distributed Processing. *AIAA Journal, vol 41(10):1996-2003.*

Sobieszczanski-Sobieski, J., Barthelemy, J., and Riley, K. (1982). Sensitivity of Optimum Solutions of Problem Parameters. *AIAA Journal, vol 20(9).*

Sobieszczanski-Sobieski, J., Emiley, M., Agte, J., and Sandusky, R. (2000). *Advancement of Bi-Level Integrated System Synthesis (BLISS).* NASA/TM-2000-210305.

Sobieszczanski-Sobieski, J. and Haftka, R. (1997). Multidisciplinary aerospace design optimization : survey of recent developments. *Structural Optimization, n°14:1-23.*

Srinivas, M. and Patnaik, L. (1994). Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms. *IEEE Transactions on Systems, Man and Cybernetics, vol 24(4):656-667.*

Summerfield, M. (1951). A Theory of Unstable Combustion in Liquid Propellant Rocket Systems. *Journal of the American Rocket Society, vol 21(5):108-114.*

Sutton, G. and Biblarz, O. (2001). *Rocket Propulsion Elements.* John Wiley & Sons, Inc.

Tava, M. and Suzuki, S. (2002). Multidisciplinary Design Optimization of the Shape and Trajectory of a Reentry Vehicle. *Transactions of the Japan Society for Aeronautical and Space Sciences, vol 45(147):10-19.*

Tava, M. and Suzuki, S. (2003). Integrated multidisciplinary and multicriteria optimization of a space transportation system and its trajectory. In *54th International Astronautical Congress of the International Astronautical Federation. Bremen, Germany.*

Tosserams, S., Etman, L., and Rooda, J. (2009). A classification of methods for distributed system optimization based on formulation structure. *Structural Multidisciplinary Optimization.*

Tosserams, S., Kokkolaras, M., Etman, L., and Rooda, J. (2010). A Nonhierarchical Formulation of Analytical Target Cascading. *Journal of Mechanical Design vol 23/051002.*

Tribes, C., Dubé, J., and Trépanier, J. (2005). Decomposition of multidisciplinary optimization problems : formulations and application to a simplified wing design. *Engineering Optimization, vol 37(8):775-796.*

Tsuchiya, T. and Mori, T. (2002). Multidisciplinary Design Optimization to Future Space Transportation Vehicles. In *AIAA-2002-5171.*

Tsuchiya, T. and Mori, T. (2004). Optimal Conceptual Design of Two-Stage Reusable Rocket Vehicles Including Trajectory Optimization. *Journal of Spacecraft and Rockets,vol 41(5):770-778.*

Villeneuve, F. and Mavris, D. (2005). A New Method of Architecture Selection for Launch Vehicles. In *13th AIAA/CIRA International Space Planes and Hypersonics Systems and Technologies. Capua, Italy.*

Walters, F., Parker, L., Morgan, S., and Deming, S. (1991). *Sequential Simplex Optimization.* CRC Press, 1st Edition.

Wujek, B., Renaud, J., and Batill, S. (1997). A Concurrent Engineering Approach for Multidisciplinary Design in a Distributed Computing Environment. *Multidisciplinary Design Optimization: State-of-the-Art, N. Alexandrov and M.Y. Hussaini (Ed.), SIAM Series: Proceedings in Applied Mathematics 80, pp. 189 - 208.*

Wujek, B., Renaud, J., Batill, S., and Brockman, J. (1996). Concurrent Subspace Optimization Using Design Variable Sharing in a Distributed Computing Environment. *Concurrent Engineering, vol 4(4):361-377.*

Yokoyama, N., Suzuki, S., Tsuchiya, T., Taguchi, H., and Kanda, T. (2005). Multidisciplinary Design Optimization of SSTO Space Plane Considering Rigid Body Characteristics. In *42nd AIAA Aerospace Sciences Meeting and Exhibit. Reno, Nevada, USA.*

Yokoyama, N., Suzuki, S., Tsuchiya, T., Taguchi, H., and Kanda, T. (2007). Multidisciplinary Design Optimization of SSTO Space Plane Considering Rigid Body Characteristics. *Journal of Spacecraft and Rockets, vol 44(1):121-131.*

Index

ABSTRACT

Optimal design of launch vehicles is a complex multidisciplinary design optimization (MDO) problem that has the distinctive feature of integrating a trajectory optimization which is very constrained, difficult to solve and highly coupled with all the other disciplines involved in the design process (*e.g.* propulsion, aerodynamics, structure, *etc.*). This PhD thesis is focused on the methods which allow to adequately integrate the optimal control law calculation into the design variable optimization process. A new method, called "Stage-Wise decomposition for Optimal Rocket Design" (SWORD), has been proposed. This method splits up the design process into the different flight phases and transforms the multistage launch vehicle optimization problem into the coordination of the optimizations of each of the stages, which are easier to solve. The SWORD method has been compared to the classical MDO method (Multi Discipline Feasible) in the case of the global optimization of a three-stage-to-orbit launch vehicle. The results show that the SWORD method allows to improve the efficiency of the optimization process concerning the feasible solution space search velocity and the quality of the optimum obtained in a limited calculation time. Moreover, an optimization strategy dedicated to the SWORD method has been developed. It allows to improve the convergence velocity of the method without requiring any *a priori* knowledge from the user with regard to the initialization and the search space.

Keywords : Multidisciplinary Design Optimization, Launch Vehicle Design, MDO, optimal control, multi-level optimization, decomposition methods

RÉSUMÉ

La conception de lanceurs est un problème d'optimisation multidisciplinaire (MDO) complexe qui a la particularité d'intégrer une optimisation de trajectoire très contrainte, difficile à résoudre et fortement couplée à toutes les autres disciplines entrant en jeu dans le processus de conception (*e.g.* propulsion, aérodynamique, structure, *etc.*). Cette thèse s'intéresse aux méthodes permettant d'intégrer judicieusement l'optimisation de la trajectoire au sein du processus d'optimisation des variables de conception. Une nouvelle méthode, appelée "Stage-Wise decomposition for Optimal Rocket Design" (SWORD), a été proposée. Celle-ci décompose le processus de conception suivant les différentes phases de vol et transforme le problème d'optimisation de lanceur multiétage en un problème de coordination des optimisations de chacun des étages, plus facile à résoudre. La méthode SWORD a été comparée à la méthode MDO classique (Multi Discipline Feasible) dans le cas d'une optimisation globale d'un lanceur tri-étage. Les résultats montrent que la méthode SWORD permet d'améliorer l'efficacité du processus d'optimisation, tant au niveau de la vitesse de recherche de l'espace de solutions faisables que de la qualité de l'optimum trouvé en temps de calcul limité. Afin d'améliorer la vitesse de convergence de la méthode tout en ne requérant pas de connaissance *a priori* de l'utilisateur au niveau de l'initialisation et l'espace de recherche, une stratégie d'optimisation dédiée à la méthode SWORD a été développée.

Mots-clés : Optimisation multidisciplinaire, conception de lanceurs, MDO, commande optimale, optimisation multiniveau, méthodes de décomposition