Master of Science in Aerospace Engineering

Research Project S3 Report

Launch Vehicle Trajectory Module For Multidisciplinary Design Analysis and Optimization

Jorge L. valderrama

supervised by:

Dr. Mathieu balesdent
DTIS, ONERA

Dr. Loïc brevault
DTIS, ONERA

Dr. Annafederica urbano
DCAS, ISAE-SUPAERO

Starting date of project: January 27, 2020
Submission date: April 2, 2021

Declaration of authenticity

This assignment is entirely my own work. Quotations from literature are properly indicated with appropriate references in the text. All literature used in this piece of work is indicated in the bibliography placed at the end. I confirm that no sources have been used other than those stated.

I understand that plagiarism (copy without mentioning the reference) is a serious examination offense that may result in disciplinary action being taken.

Signature: Jorge Luis Valderrama _____ Date: 02/04/2021 _____

Jorge Luis Valderrama

Contents

List of Figures

List of Tables

Nomenclature

Multidisciplinary Design Optimization (MDO)

f	: Objective function
$\boldsymbol{x}$	: State variables
$\boldsymbol{y}$	: Input coupling variables
$\boldsymbol{z}$	: Design variables
$\boldsymbol{c}$	: Output coupling functions
$\boldsymbol{g}$	: Inequality constraints
$\boldsymbol{h}$	: Equality constraints
$\boldsymbol{R}$	: Residuals

Optimal control

J	: Performance index for the Bolza problem
$\boldsymbol{x}_t$	: State variables of the trajectory
$\boldsymbol{u}$	: Dynamic controls
$\boldsymbol{z}_t$	: Design variables of the trajectory
t	: Time
$\boldsymbol{g}_0$	: Initial boundary constraints
$\boldsymbol{g}_f$	: Final boundary constraints
$\boldsymbol{p}$	: Path constraints
f	: Mayer term of J
$\boldsymbol{\mathcal{L}}$	: Lagrangian

Propulsion

P_c	: Pressure at combustion chamber
P_e	: Pressure at nozzle exit
O/F	: Mass ratio of oxidizer to propellants
T_{vac}	: Thrust at vacuum for the stage
γ_t	: Isentropic coefficient at the throat
T_c	: Temperature at combustion chamber
M_c	: Molecular mass at combustion chamber
$c^\star$	: Characteristic velocity
ϵ	: Area ratio
C_f	: Thrust coefficient
A_t	: Throat area
A_e	: Nozzle exit area of one engine
$\eta_{c^\star}$	: $c^\star$ efficiency
η_{c_f}	: C_f efficiency
R	: Gas constant per unit weight
g_0	: Standard acceleration due to gravity
I_{sp_2}	: Specific impulse
$\dot{m}_{max}$	: Mass flow rate at full throttle
A_{e_t}	: Nozzle exit area of all engines

Mass/Sizing

m_p : Mass of propellants of the stage
m_s : Structural mass of the stage
D : Diameter of the vehicle

Trajectory

$\Delta\theta_{lpo}$: Change of pitch angle during Linear Pitch-over phase
$\Delta\theta_{btl}$: Change of pitch angle at the start of Exo-atmospheric phase
θ_{btl_f} : Pitch angle for bilinear tangent law
ξ : Shape parameter for bilinear tangent law
R_e : Radius of Earth at the equator
h : Altitude
ρ : Air density
θ_{btl_0} : Pitch angle at the beginning of Exo-atmospheric phase
θ_{gt_f} : Pitch angle at the end of Gravity turn phase
q : Dynamic pressure
q_{max} : Maximum dynamic pressure
q_{heat} : Heat flux
$n_{ax_{max}}$: Maximum axial load factor
m : Mass
v : Velocity
r : Radius
λ : Longitude
ϕ : Flight path angle
t_p : Time during phase p
t_{p_0} : Initial time of phase p
Δt_p : Duration of phase p
H : Specific angular momentum per unit mass
E : Specific energy per unit mass
μ : Standard gravitational parameter of earth
a : Semi-major axis of elliptical transfer orbit
e : Eccentricity of elliptical transfer orbit
r_a : Radius at apogee of elliptical transfer orbit
r_p : Radius at perigee of elliptical transfer orbit

LGL transcription

$\boldsymbol{x}_t$	: State variables of the Trajectory Discipline
$[A_i] = [B_i]$	: Hermite interpolation matrices
$[A_d] = [B_d]$	: Hermite differentiation matrices
t_{seg}	: Duration of segment
$\boldsymbol{x}'_t$	: Approximated derivative of states via Hermite polynomial
$\dot{\boldsymbol{x}}'_t$	: Derivative of states
$\boldsymbol{C}_x$	: Continuity constraints for states
$\boldsymbol{C}_t$	: Continuity constraints for time
$\boldsymbol{\Delta}$	: Defect constraints
$\boldsymbol{h}_{t_{trans}}$	: Transcription constraints containing boundary conditions, $\boldsymbol{C}_x$ and $\boldsymbol{C}_t$
t_t	: Time variables of the trajectory discipline
$()_a$	: Subscript for values at all nodes
$()_d$	: Subscript for values at state discretization nodes
$()_c$	: Subscript for values at collocation nodes

Abbreviations

Abstract

In Launch Vehicle Design (LDV), multiple disciplines as trajectory, sizing, mass calculations, propulsion, aerodynamics, and costs are tightly coupled with complex interactions. Multidisciplinary Design Optimization (MDO) methods are used to model these interactions and find designs that maximize performance and minimize cost. MultiDiscipline Feasible (MDF) is the most used MDO fomulation for launch vehicle desgin [4]. It relies on a shooting strategy to solve the optimal control problem related to the trajectory discipline and a Multidisciplinary Analysis (MDA) solver to satisfy the interdisciplinary couplings between the disciplines. Consequently, an auxiliary optimal control problem (based on a time marching integrator) and a fixed-point iterator are nested in an optimization loop driven by a global optimizer, resulting in a computational burden.

A new MDO strategy is presented in this work to reduce the computational cost. It relies on orthogonal collocation to solve the optimal control problem and uses an All-At-Once-like (AAO-like) MDO formulation for the feedback couplings. This strategy converts the whole MDO of the launch vehicle into a unique, high-dimensional non-linear programming problem that is solved using a gradient-based optimizer with analytic derivatives. A study case of a Two-Stage To Orbit (TSTO) vehicle is done to benchmark the proposed methodology against a traditional MDF formulation.

Keywords: Launch Vehicle Design, Multidisciplinary Design Analysis and Optimization(MDAO), Trajectory Optimization, Orthogonal Collocation

1 Introduction

In Launch Vehicle Design (LDV) is desired to obtain the maximum performance at the lowest cost possible for a given objective mission. In pursuing these aims, the designer must consider the interactions between multiple disciplines. Typically, these disciplines include trajectory, geometry and sizing, mass calculations, propulsion, aerodynamics, and costs. They are all intertwined with complex interactions that make the launch vehicle design process burdensome and complicated.

In a traditional design approach, individual optimization of each discipline is performed. Consequently, the results are transferred to the design offices in charge of the other disciplines in an iterative process. This approach presents difficulties to consider all of the complex interactions between the different disciplines, thus yielding non-optimal results. In the quest for designing more performant and lower-cost launch vehicles, Multidisciplinary Design Analysis and Optimization (MDAO) approaches have been used to better assess the complex interactions among disciplines during the optimization process. According to Balesdent et al., the most common MDAO approach for LDV is MultiDiscipline Feasible (MDF) [4]. In this method, the design variables for all disciplines are driven by a unique optimizer in a single level. At each iteration of the optimizer, the consistency of the coupling variables is guaranteed by solving a multidisciplinary analysis (MDA) with a fixed point iteration or a Newton solver. This nested loop represents a high computational cost as the MDA must involve the trajectory discipline. Meaning that the auxiliary optimal control problem that involves the Ordinary Differential Equations (ODE) solver must be called at each iteration of the MDA. In the MDAO of launch vehicles, the trajectory is considered of particular importance. It represents a high portion of the associated computational cost and is in charge of assessing the vehicle performance by feeding back the evaluation of mission constraints to the optimizer.

A new MDO approach for LDV is proposed in this research project to reduce the computational cost of traditional methods. It relies on a derived version of All-At-Once (AAO) only for the feedback coupling and orthogonal collocation to transcribe the trajectory optimization. In the proposed approach, the design variables, the coupling variables, and the variables associated with the transcribed trajectory are driven by the same optimizer, eliminating the nested loops and reducing the number of calls to the disciplines. Such methodology is implemented using NASA's OpenMDAO and Dymos, two Python

libraries with open source access. OpenMDAO enables the efficient propagation of partial derivatives to perform gradient-based MDAO processes. Dymos allows using the Legendre-Gauss-Lobatto (LGL) orthogonal collocation method for the optimal control of dynamic systems.

The performance of the new methodology is compared against Onera's Felin [5] using the case study of a TSTO vehicle. A routine for the MDAO of launch vehicles relying on an MDF architecture and an evolutionary global optimizer.

1.1 Aim

To integrate a trajectory optimization module using Legendre-Gauss-Lobatto (LGL) collocation into the MDAO of a launch vehicle.

1.2 Objectives

- To develop the trajectory module using gradient-based optimization with analytic derivatives.
- To evaluate the performance of the optimization process in terms of its convergence domain, sensitivity to the initialization, and computational cost.
- To compare the results and performance with a traditional methodology.

1.3 Justification of the potential degree of novelty

The proposed integration of the LGL orthogonal collocation technique in the MDO process of LDV has not been done yet in the literature. This approach combines four important elements (that will be discussed in details in the following sections):

- The use of an orthogonal optimal collocation technique for the trajectory discipline transcription.
- A re-organization of the MDAO process following a derived All-At-Once formulation (for the feedback couplings).
- A gradient-based optimization algorithm to efficiently solve the high-dimensional MDO problem.
- The propagation of gradient through the multidisciplinary design process using chain rule.

The author will work on the redaction of a journal paper during the current year to present the results described in this report to the academic community. A conference paper at the 14th World Congress of Structural and Multidisciplinary Optimization (WCSMO, June 2021) entitled "All-At-Once MDO formulation for coupled optimization of launch vehicle design and its trajectory using a pseudo spectral method" has been submitted and accepted.

1.4 Work breakdown

The author carried this research project during the second and third semesters of the MSc. in Aerospace Engineering program. The achieved results are the outcome of a progressive strategy to refine the complexity of the models. This strategy helped to test and validate results starting from simple cases and developed the understanding of the author in the fields of MDAO, optimal control, and LVD in a gradual way.

1.4.1 Second semester

The goals reached during the second semester are:

- Review of the literature.
- Programming of a single-shooting trajectory optimization for the direct injection into a circular orbit of a single-stage-to-orbit (SSTO) vehicle.
- Programming of a trajectory optimization using the LGL collocation technique for the direct injection into a circular orbit of a single-stage-to-orbit (SSTO) vehicle.

1.4.2 Third semester

The goals reached during the third semester are:

- Programming of a trajectory optimization using the LGL collocation technique for the Hohmann transfer ascent of a two-stage-to-orbit (TSTO) vehicle.
- Integration of the aforementioned trajectory optimization scheme in the MDO of a TSTO vehicle, including the mass-sizing and propulsion disciplines using an AAO-like approach for the feedback couplings.
- Comparison of the results and performance with Felin.

2 Literature Review

This literature review is related to three main topics. MDAO, optimal control, and launchers trajectories. In a first step, this literature review describes some of the main MDAO architectures used for LVD, followed by a presentation of optimal control methods used in the aerospace field. In a final step, aspects related to the trajectory are addressed.

2.1 MDAO

Typically, the MDAO of a launcher is divided into disciplines considering different physics or aspects of the design process. Nevertheless, they are strongly linked to each other as LVD considers the launcher as a whole, and changes in one discipline can greatly influence the others. As an example, let us consider two typical disciplines in the MDAO of launchers, like Mass and Trajectory. On one side, the Mass discipline is in charge of computing the mass of the different systems composing the launcher. On the other side, the Trajectory discipline is in charge of computing the path and the velocity profile that the vehicle must follow to fulfill the requirements of a given mission. A trajectory has a maximum dynamic pressure point that dictates the maximum aerodynamic forces experienced by the launcher. The stronger these aerodynamic loads, the stronger the launcher should be to withstand them. If the initial mass of the vehicle changes, the path, and velocity profile that it follows will change too. As the velocity profile changes, the maximum dynamic pressure changes; thus, the strength and mass required for the launcher to withstand the loads will change. The resultant dependency loop between both disciplines is an example of the intricate interactions that are solved during an MDAO process while trying to reach an optimal solution.

MDAO formulations can be classified according to the number of optimizers they require as Single-level or Multi-level. According to Falck and Gray [3] single-level methods usually outperform multilevel methods. In LVD, the Single-level formulations are the most common and are described in this subsection. For this purpose, let us first define the mathematical formulation of the MDAO problem, the treatment of the couplings, and the aims of the process:

2.1.1 Mathematical formulation of the MDO problem

$$
\begin{aligned}
\text{minimize} \quad & f(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \\
\text{with respect to} \quad & \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \\
\text{subject to:} \quad & \\
\text{inequality constraints} \quad & \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \leq \boldsymbol{0} && (2.1) \\
\text{equality constraints} \quad & \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \leq \boldsymbol{0} && (2.2) \\
\text{residuals} \quad & \forall i, \boldsymbol{R}_i(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{z}_i) = \boldsymbol{0} && (2.3) \\
\text{coupling variables} \quad & \forall i, \forall j \neq i, \boldsymbol{y}_{ji} = \boldsymbol{c}_{ji}(\boldsymbol{x}_j, \boldsymbol{y}_j, \boldsymbol{z}_j) && (2.4)
\end{aligned}
$$

Where $\boldsymbol{x}$ represents the state variables (e.g., variables defining a thermochemical equilibrium), $\boldsymbol{y}$ represents the input coupling variables (e.g., stage dry mass, specific impulse) and $\boldsymbol{z}$ represents the design variables (the variables defining the launcher architecture that are optimized e.g., propellant masses, stage diameter, chamber pressure). The subscripts mean that they belong to the discipline i or j. During the optimization, the cost function f is minimized (e.g., the Gross-Lift-Off-Weight). The output coupling functions are expressed by $\boldsymbol{c}$. The coupling variables vector $\boldsymbol{y}_{ij}$ is the input of discipline j and the output of discipline i.

2.1.2 Types of couplings and their handling

The distinction between the following two types of coupling variables is of special importance in an MDAO process:

- Feedforward: coupling variables transmitted to a component that has not been executed in the current function evaluation of the MDAO.
- Feedback: coupling variables transmitted to a component that has been already executed in the current function evaluation of the MDAO.

The couplings can be managed in two different approaches:

- Coupled: For one value of the design variables, an auxiliary process (referred to MultiDisciplinary Analysis) is in charge to determine the value of the coupling variables (both feedforward and feedbacks couplings) leading to an interdisciplinary consistent system of nonlinear equations (corresponding to Eq.2.4). For instance, fixed point iteration can be used for such purpose.
- Decoupled: Feedforward and feedback coupling variables are removed and input coupling variables are controlled at the system level along with the design variables and interdisciplinary constraints between the input coupling variables (controlled by the optimizer) and the output coupling variables (resulting from the discipline evaluations) are added (Eq.2.4) in order to ensure interdisciplinary consistency.

Feedback couplings are computationally expensive as they create loops that must be solved through an MDA in a coupled approach or through additional variables and constraints in a decoupled approach. The aforementioned loop for the dynamic pressure between the Mass and Trajectory disciplines is an example of a feedback coupling. Typically, the designer tries to minimize the number of feedback couplings by properly modeling the problem.

Multiple numerical methods can be involved in an MDAO process that at the same time can be approached with different formulations. Nevertheless, the following aims are always pursued:

2.1.3 Aims of the MDAO process

1. Disciplinary feasibility: Satisfaction of Eq. 2.3 by means of the state variables $\boldsymbol{x}$.
2. Multidisciplinary feasibility: Consistency of the couplings (Eq. 2.4) by means of the coupling variables $\boldsymbol{y}$.
3. Feasibility of the optimization problem: Satisfaction of Eq. 2.1 and Eq. 2.2.
4. Optimality of the problem: minimization of f by means of the design variables $\boldsymbol{z}$.

2.1.4 Single-level MDAO formulations

Let us now present a classification of the single-level MDAO formulations:

- **Multidiscipline feasible:**
  This architecture is the most commonly used in LVD, according to Balesdent et al. [4]. In this approach, each discipline is in charge of satisfying its disciplinary feasibility by means of Eq. 2.3. The Multidisclipline Feasible (MDF) is a coupled approach and uses an MDA to solve the couplings between the different disciplines to satisfy Eq. 2.4 at each optimizer iteration. Usually, in LVD the Trajectory discipline is inside the MDA loop, meaning that it is executed a large number of times, leading to a high computational cost. MDF is sensitive to the number of feedback couplings as they render the MDA more complex to solve.

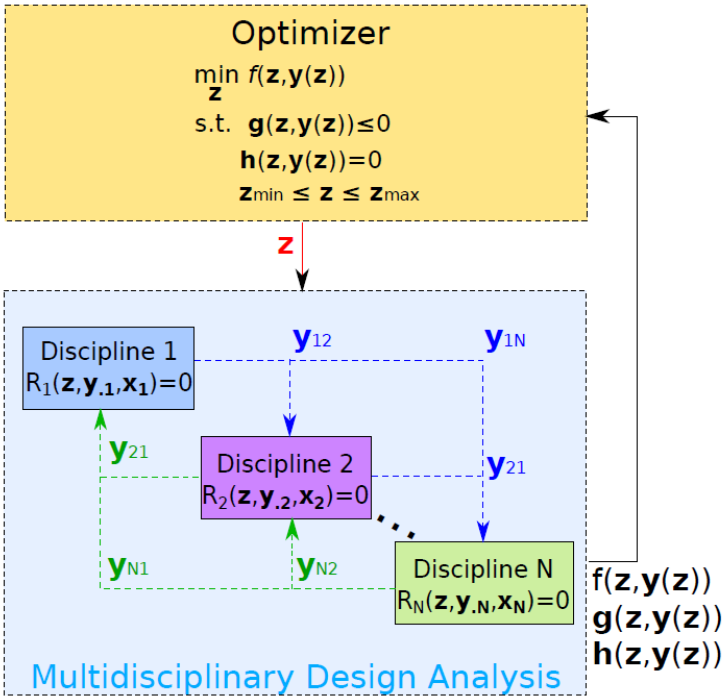

Figure 1: Multidiscipline Feasible (MDF) formulation. Schema taken from [1]

- **Individual discipline feasible**
  The individual discipline feasible is a decoupled approach. This formulation favors faster execution times than MDF. It does not use an MDA to satisfy the couplings (Eq. 2.4), but rather it does so at the optimizer level, requiring, in general, more optimizer iterations and ensuring multidisciplinary feasibility only at the convergence of the optimizer. In the IDF approach, the disciplinary feasibility is ensured by each discipline to satisfy Eq. 2.3. In [6], Castellini studied the MDAO of expendable launchers and compared the IDF and MDF formulations. He did not conclude which one represented a lower computational cost and found different final cost function values using an evolutionary global optimizer.

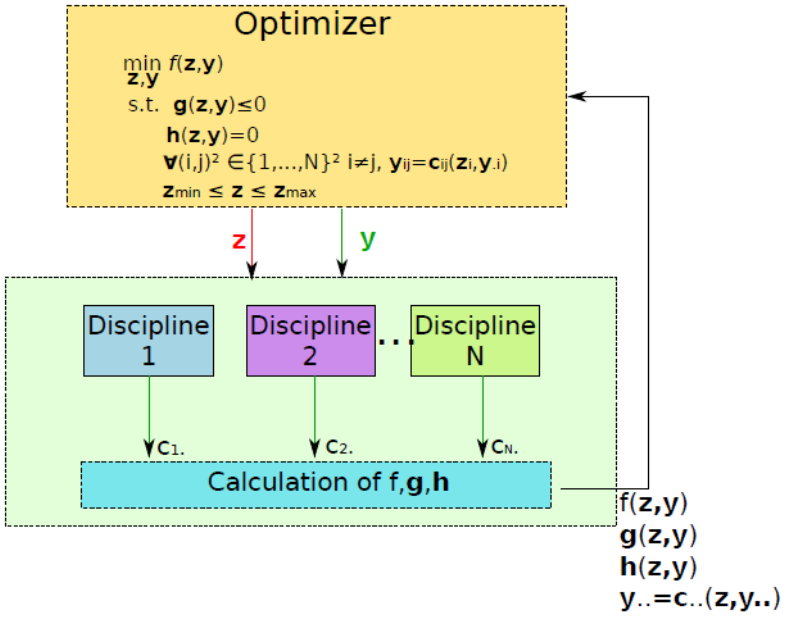

Figure 2: Individual Discipline Feasible (IDF) formulation. Schema taken from [1]

- **All-at-once**
  In the decoupled All-at-once (AAO) formulation, the couplings (Eq. 2.4) and the residuals (Eq. 2.3) are handled at the optimizer level, meaning that disciplinary and multidisciplinary feasibility are only guaranteed at the optimizer convergence. This approach does not have a nested MDA, but it handles the couplings by creating extra design variables and constraints. Furthermore, the state variables are also handled by the optimizer. The resulting problem is characterized by a limited number of discipline evaluations but high complexity due to an increased number of constraints and variables for the optimizer.

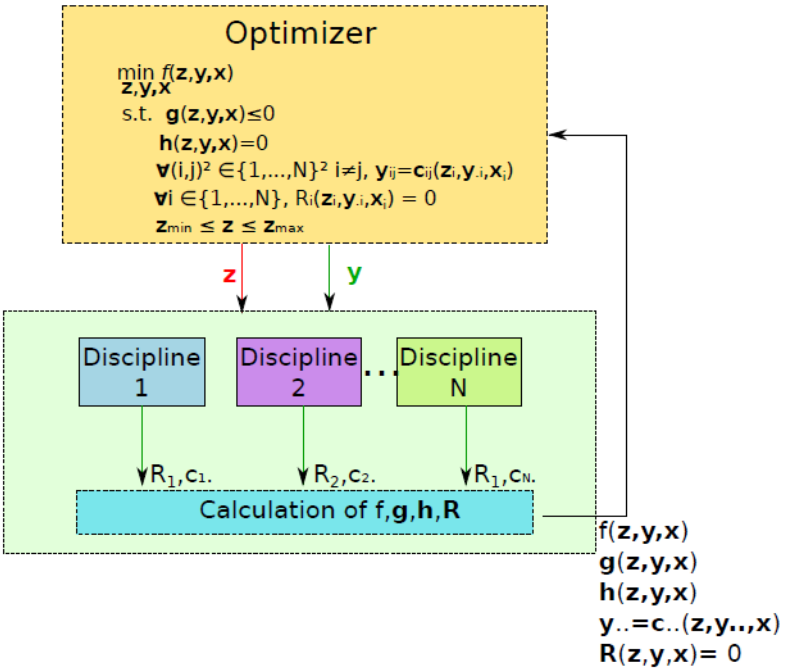

Figure 3: All-At-Once (AAO) formulation. Schema taken from [1]

2.1.5 Optimization algorithms

There are numerous types of optimization algorithms used in an MDAO process. A classification of these algorithms was done by Martins and Ning in [7]. However, their characterization is limited to 4 attributes in this work, namely, their order, search type, iteration procedure, and stochasticity.

- **Order**
  The order of the optimization algorithm is related to the derivative information they need. Zeroth-order algorithms do not need gradient information and are also called gradient-free. First-order methods require Jacobian information, and second-order methods require Jacobian and Hessian information. First and second-order methods can also be grouped in the category of gradient-based methods. Gradient-based methods benefit from having clear mathematical criteria to evaluate the optimality conditions and require at least a C^1 continuity condition.
- **Search type**
  The algorithms can perform a local search, meaning that they explore an area of the search space corresponding to a local minimum or look for optimal solutions in the whole domain doing a global search. Gradient-based methods perform a local search, excepted for specific cases (e.g., convex functions in a convex research space).
- **Iteration procedure**
  The method utilized by an optimizer to generate a new set of design variables can be classified

as mathematical or heuristic. Gradient-based algorithms follow a defined mathematical iteration, whereas population-based algorithms are based on heuristics or thumb rules. This raises the problem of defining optimality conditions.

- **Stochasticity**
  Depending on the utilization of randomness-based steps, an optimization algorithm can be classified as stochastic or deterministic. A stochastic algorithm will lead to different results when ran different times under the same conditions (if the random seed is not fixed). In contrast, deterministic algorithms lead to the same results under the same initialization conditions.

Castellini [6] used the Particle Swarm Optimization algorithm (PSO-1D) in his MDAO study for expendable launchers. PSO algorithms are global, gradient-free, heuristic, and stochastic. He found differences of around 7% in the minimum launch mass of a launcher in different runs of the optimization using an MDF strategy.

2.2 Optimal control

Optimal control problems are sometimes also called trajectory optimization problems. Betts introduced a compendium of methods for trajectory optimization in [8] and [9]. Rao did the same for optimal control in [2] and defined trajectory optimization problems as a subset of optimal control where the input parameters are static. Thus, dealing with the optimization of functions. Whereas in the more general case Optimal Control is related to functional optimization problems.

Three main classifications can be done for optimal control methods: heuristic optimization methods, indirect methods and direct methods. In the following sections, a brief description of heuristic and indirect methods is done followed by a broader discussion of direct methods.

2.2.1 Mathematical formulation of optimal control problem

Let us consider the mathematical formulation of the optimal control problem for a unique phase and with the performance index J for the Bolza form as defined in [2].

$$\text{minimize} \quad J = f\left[\boldsymbol{x}_t(t_0), t_0, \boldsymbol{x}_t\left(t_f\right), t_f; \boldsymbol{z}_t\right] + \int_{t_0}^{t_f} \mathcal{L}[\boldsymbol{x}_t(t), \boldsymbol{u}(t), t; \boldsymbol{z}_t]dt \tag{2.5}$$

$$\text{with respect to} \quad \boldsymbol{x}_t, t, \boldsymbol{u}, \boldsymbol{z}_t$$

$$\text{subject to:}$$

$$\text{dynamics} \quad \dot{\boldsymbol{x}}_t = \boldsymbol{f}_{ode}[\boldsymbol{x}_t(t), t, \boldsymbol{u}(t), \boldsymbol{z}_t] \tag{2.6}$$

$$\text{initial boundary constraints} \quad \boldsymbol{g}_{0,lb} \leq \boldsymbol{g}_0\left(\boldsymbol{x}_{t_0}, t_0, \boldsymbol{u}_0, \boldsymbol{z}_t\right) \leq \boldsymbol{g}_{0,ub} \tag{2.7}$$

$$\text{final boundary constraints} \quad \boldsymbol{g}_{f,lb} \leq \boldsymbol{g}_f\left(\boldsymbol{x}_{t_f}, t_f, \boldsymbol{u}_f, \boldsymbol{z}_t\right) \leq \boldsymbol{g}_{f,ub} \tag{2.8}$$

$$\text{path constraints} \quad \boldsymbol{p}_{lb} \leq \boldsymbol{p}_0\left(\boldsymbol{x_t}, t, \boldsymbol{u}, \boldsymbol{z}_t\right) \leq \boldsymbol{p}_{ub} \tag{2.9}$$

Where the aim is to determine the states $\boldsymbol{x}_t$ describing the trajectory, the dynamic controls $\boldsymbol{u}$, the design variables $\boldsymbol{z}_t$, the initial time t_0 and the final time t_f that minimize the cost functional J. The cost functional is composed of the Mayer term, f, and the Lagrangian, $\mathcal{L}$.

2.2.2 Heuristic methods

Usually, heuristic optimization methods are global, which implies that they are suited to find solutions in search spaces with many local minima. In a similar way to the optimizers mentioned is section 2.1.5, heuristic optimal control algorithms lack mathematical formality to define optimality conditions.

2.2.3 Indirect methods

Indirect methods are founded on the calculus of variations and the formulation of first-order optimality conditions for the original problem, forming a Hamiltonian Boundary Value Problem (HBVP) whose solutions are determined numerically. This strong formal background can ensure that solutions, when obtained, are indeed optimal. In [8], an application of indirect methods for low thrust orbit analysis was demonstrated. In general, these methods require experience to provide an initial guess due to their sensibility, making really difficult to explore changes on the launcher design variables during an MDAO process.

2.2.4 Direct methods

Direct methods are less sensitive to the quality of the initial guess. The mapping between direct and indirect formulation has been done for some specific methods. Herman and Conway [10] used these relations to determine the accuracy of a Direct high-order Gauss Lobatto Collocation scheme. Garg *et al.* [11] demonstrated this mapping for a Radau Pseudospectral Method. The importance of this notion is that some direct methods benefit from a low sensibility to the initial guess, and their outputs can be verified with the mapping with the indirect formulation, thus guaranteeing high accuracy. This also represents an advantage for the multidisciplinary optimization of launch vehicles as trajectories can vary drastically as the design variables ($\boldsymbol{z}$) vary along the MDO process. For these reasons, this work focuses on direct methods.

Direct methods use two classes of numerical methods as a basis. The first one deals with the numerical solution of differential equations and integration. The second one is related to non-linear optimization. The first class is used in a transcription process to formulate a non-linear optimization problem which then will look for the values of $\boldsymbol{x}_t$, t, $\boldsymbol{u}$ and $\boldsymbol{z}_t$ that minimize Eq. 2.5 satisfying the constraints (Eqs. 2.7, 2.8 and 2.9). These classes of numerical methods will be described in the following sections.

- **Numerical Solution of Differential Equations and Integration**
  Time marching methods are used to approximate the solution of Eq. 2.6 numerically at a given time step using the previous time steps solutions. Hence, in an Initial Boundary Value (IVB) problem they use the initial boundary conditions to solve for the next step and continue using the obtained value to solve for the next one in an recursive manner. Adams methods and Runge-Kutta methods, including Euler and Hermite-Simpson formulations, are covered in [2] and [9].

  In collocation methods, coefficients of an interpolating polynomial are calculated simultaneously and are constrained to match the ODE at the collocation points when approximating the solution of the equation. Different formulations as Gauss, Radau and Lobatto methods exist and their differences are based on the inclusion or not of endpoints on the current interval. A subset of these methods is orthogonal collocation which uses orthogonal polynomials to approximate the functions, typically Chebyshev or Legendre polynomials as explained in [2], but also Jacobi polynomials were used for a High-Order Gauss Lobatto method implemented in [10].

  Numeric integration is also required as the cost functional (Eq. 2.5) in the optimal control formulation often includes an integral term. The approximation of the cost function and the differential equations must be consistent.

- **Non-Linear Optimization**
  In the direct methods, the optimal control problem is transcribed to a Non-Linear Programming (NLP) problem. NLP solvers are mostly based on Newton's method and formulate the problem in the Karush-Khun-Tucker (KKT) form. These formulations are widely covered in [9]. One of the biggest challenges of the transcription process and the NLP solvers is to take advantage of the sparsity of the matrix formulation, specially when solutions for the equations are discretized in multiple segments, thus increasing the number of NLP variables. Some NLP solvers that are used frequently in the literature are SNOPT in [2] [12] [13] , IPOPT in [2] and NPSOL in [8].

Two broad families of direct methods can be distinguished, namely shooting and collocation. Rao [2] proposed the classification shown in figure 4 for the direct optimal control methods. This classification addresses an ambiguity in the literature, where a special type of collocation methods that use orthogonal polynomials are often called pseudospectral methods.

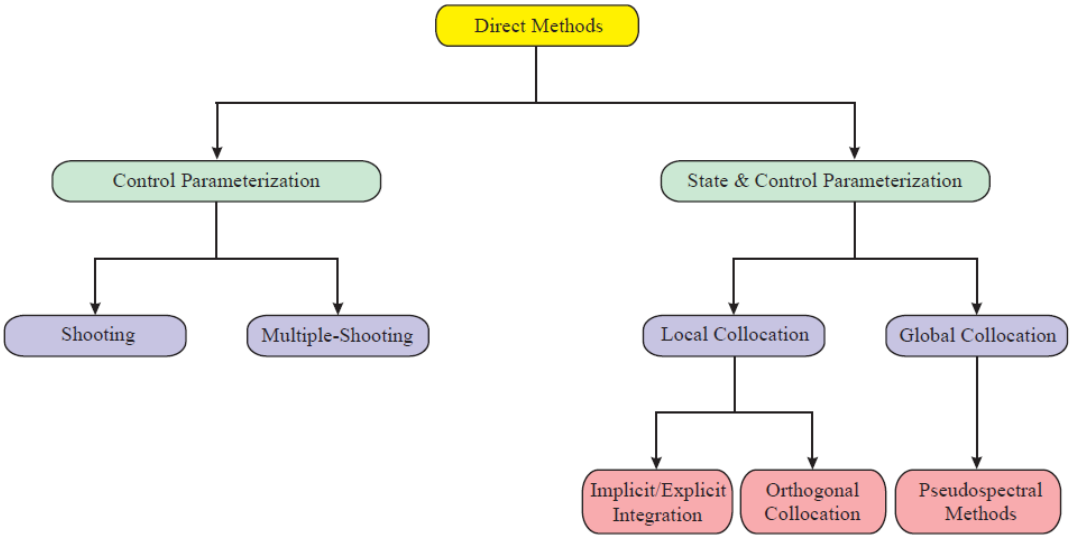

Figure 4: Direct methods for optimal control. Taken from [2]

- **Single-shooting**
  The simplest direct method is called single shooting. In the case of a Two-Point Boundary Value (TPBV) problem, it uses an initial guess for the design variables, the dynamic controls and a numerical method like Runge-Kutta to propagate the solution accordingly. The obtained value at the final time is then compared with the final boundary conditions to calculate an error. Then, the NLP solver is used to find the set of design variables and dynamic controls that take the error to zero.

- **Multiple-shooting**
  A more complex method is multiple shooting. It discretizes the time into a set of intervals and the associated starting and final interval state values are initialized. Then, within each time interval, a single shooting approach is carried out.

  The interest of multiple shooting techniques is the decrease of sensitivity of the final state (and the optimal control objective function Eq. 2.5) with respect to the value of the control variables ($\boldsymbol{u}$). Multiple shooting allows a parallel execution and its often used for launcher ascent trajectories. To guarantee continuity of the solution between the different time intervals, this method requires constraints that increase the size of the NLP problem (number of optimization variables - the initial and final state variables, the number of constraints - match between the initial and final interval state values). Single and multiple-shooting methods are covered in [2] and [8].

- **Local Orthogonal Collocation**
  Collocation methods require a polynomial parameterization of the states. This induces a discretization error that can be reduced by increasing the interpolating polynomial degree or by increasing the number of segments used in the parameterization. In local orthogonal collocation, the interpolation polynomial degree is fixed and the number of segments is varied.

- **Global collocation - Pseudospectral**
  In Global Orthogonal Collocation or Pseudospectral methods, the state or solution is approximated using global polynomials collocated at the collocation points. In these methods, the number of segments or meshes is fixed and the degree of the polynomial is varied. One remarkable advantage of these methods is the spectral convergence (exponential rate) as a function of the number of collocation points. In the survey of methods done by Rao [2], the Gauss-Lobatto Pseudospectral method and the Pseudospectral method Using Legendre-Gauss-Radau points were described. As mentioned before, Herman and Conway [10] worked with High Order Gauss Lobatto methods. They showed better convergence characteristics for these methods when compared to low order methods of the same family thanks to the reduction of rounding errors associated with the lower number of segments required to approximate a function with high order polynomials. Garg *et al.* [11] demonstrated the spectral convergence of a Radau Pseudospectral method and developed the mapping with its indirect formulation. In a case study integrating a propulsion and a trajectory optimization module into an MDAO formulation, Hendricks *et al.* [12] showed that Legendre-Gauss-Radau has better performance than Legendre-Gauss-Lobatto when parallelization is used.

2.3 MDAO and its integration with optimal control

In the MDAO of launch vehicles the trajectory discipline solves an optimal control problem. In the traditional MDF formulation, this optimal control problem is solved using a shooting method. Castellini used a Runge-Kutta 4-5 for the IDF and MDF formulations in [6].

In [12], Hendricks et al. proposed a coupled modeling approach of trajectories and propulsion systems of an aircraft using the Legendre-Gauss-Radau pseudospectral method. The strategy was validated using known results for the trajectory optimization for minimum time-to-climb of an F-4 supersonic jet. They implemented their models in the OpenMDAO environment, an open-source Python framework developed at NASA Glenn Research Center, allowing the efficient handling of partial derivatives in MDAO process. Falck and Gray [13] proposed a Legendre-Gauss-Lobatto based collocation scheme with analytic derivatives for the trajectory optimization of an aircraft using OpenMDAO. The same authors presented in [3] the Dymos tool, an OpenMDAO module for the solution of optimal control problems. Dymos implements the Legendre-Gauss-Radau (LGR) pseudospectral method and the Legendre-Gauss-Lobatto collocation method and it is meant to be integrated in single-level MDAO formulations. OpenMDAO and Dymos have built-in interfaces with NLP solvers as SNOPT, IPOPT and SLSQP.

2.4 Launcher trajectories

This part of the literature review addresses the governing equations of motion (EoM) for launcher trajectories for 2D and 3D models in spherical coordinates and some strategies to cope with the singularities of the 3D case, followed by the guidance laws used for launcher ascent.

- **Equations Of Motion (EoM)**
  The model to simulate launcher trajectories is commonly simplified by neglecting rotation dynamics to keep low computation times. This simplification is based on the assumption that the attitude control system of the launcher ensures the right attitude at every time. This results in model with 3 Degrees Of Freedom (DOF). The assumption of equatorial plane trajectories results in 2D model

described by a set of 5 coupled Ordinary Differential Equations (ODE) representing changes in velocity, radial distance to the center of the planet, longitude, flight path angle and mass.

To describe the motion of the launcher in 3D space, a set of 7 differential equations is commonly used in the literature, adding 2 extra equations to describe the changes in latitude and heading angles. In the work of Briese et al. [14], the following coordinate systems are defined to obtain the set of equations in spherical coordinates:

- Earth Centered Inertial (ECI)
- Earth Centered Earth Fixed (ECEF)
- North-East-Down (N)
- Kinematic (K)
- Body fixed (B)

The different transformations between the coordinates systems are presented and a transformation matrix to express the launcher's position according to the *World Geodetic System 1984* (WGS'84) is given. Also in that work, a transformation between the ECEF and N coordinate systems is used to avoid the singularity presented when the velocity is near $0m/s$ and the flight path angle near 90°. The EoM in spherical coordinates according to the *Aerospace Standard ISO 1511* are presented in the work of Castellini [6] in the context of MDAO for expendable launch vehicles. Cartesian coordinates to express the EoM were used by Volo [15] to perform trajectory optimization using pseudospectral methods, as an important remark, the EoM in these coordinates do not present the singularity of their spherical counterpart.

All of the aforementioned models neglect the rotation dynamics to reduce the number of equations and variables. Equivalently, they have 3 DOF. Wang and Wu [16] compared the performance of 3DOF and 6DOF models for reusable launcher optimization using pseudospectral methods, finding the later to require 46% more time to be solved.

- **Forces**
  The EoM are function of gravity, thrust and aerodynamic force terms. In this subsection some approaches found in the literature referring to these topics are be described.

  A simplified model for gravity is Newton's law of universal gravitation. A more complete model accounting for Earth's elliptical shape was used in [6] to include up to the J_4 term for modelling the spherical harmonics of the Earth. The aforementioned models are broadly explained by Tewari [17]. The *Earth Gravitational Model 1996* (EGM96) is used in [14].

  Aerodynamic data is usually provided in the form of tables where drag an lift coefficients are given as functions of the angle of attack and Mach number. Any interpolation used to integrate this tabular data into the trajectory optimization model must be C_2 continuous if Non-Linear-Programming (NLP) solvers are to be used as expressed by Betts in [9]. Betts suggests to use B-splines for this purpose. Variations on atmospheric properties were modeled with the *U.S. Standard Atmosphere 1976* in [6], [12] and [18]. In [15], the *COSPAR International Reference Atmosphere 2012* was used.

- **Guidance laws**
  The guidance laws used to steer the launcher through a certain trajectory will be first described for the ascent phase and later for different recovery scenarios. To steer, a launcher can perform maneuvers with its gimbaling engines or its reaction control system (RCS) composed by thrusters with thrust components perpendicular to its longitudinal axis. The effect of these steering strategies is reflected on the pitch and yaw angles of the vehicle. In their works, Castellini [6] and Pagano [18] used similar approaches for the control strategy during the ascent phase of the launcher. This strategy considers the following sequence for the control of the pitch angle:

  1. Vertical lift-off: the vehicle ascends vertically at least up to the minimum height constrained by launchpad safety criteria.

2. Linear pitch-over: the vehicle modifies its attitude and generates angle of attack.
3. Exponential decay of pitch: the vehicle returns to its zero angle of attack position with a pitch angle different from the vertical position.
4. Gravity turn: the gravity force helps steer the vehicle towards its orbital attitude.
5. Bilinear tangent law: insertion of vehicle to its orbital attitude and motion.
6. Coast phases: engines are off and trajectory is ballistic. It happens after stage and fairings separation.
7. Stage burns: after stage separation the mass of the launcher changes.

The yaw angle guidance laws are also described by the aforementioned authors as a function of vehicle latitude and orbit inclination.

3 Methods

A new MDAO methodology for LVD is proposed to reduce the computational cost of the traditional methods. It relies on orthogonal collocation to solve the optimal control problem in the trajectory module using the methods available in Dymos. Orthogonal collocation methods are suitable for gradient-based optimization and are efficiently solved with NLP solvers. The proposed methodology uses the OpenMDAO environment to propagate analytic derivatives through the different disciplines, exploiting the potential of gradient-based optimizers. In orthogonal collocation, the physical sense and feasibility of the trajectory are only ensured at the optimizer convergence. To match this characteristic the chosen MDO strategy consists of a modified AAO approach for the feedback couplings. In the resulting strategy, the feasibility of the trajectory, the satisfaction of the couplings, and the optimality of the MDAO are driven by the same NLP optimizer and are only guaranteed once the optimizer converges. As a matter of fact, the number of constraints and variables handled by the optimizer is greatly increased compared to a traditional MDF formulation with a single-shooting strategy (from few dozen up to several hundreds). Nonetheless, the elimination of the MDA procedure reduces the number of calls to the disciplines.

A case study of a TSTO vehicle performing a Hohmann transfer ascent to a circular orbit in the equatorial plane is proposed to test the new methodology. From an MDAO perspective, it involves the Propulsion, Mass/Sizing and Trajectory disciplines. To assess the performance of the new methodology the same test case is run using a tool from ONERA called FELIN, that implements a single shooting technique with a "MDF-like" formulation with an evolutionary optimization algorithm. A description of each of the disciplines is done in the following sections, emphasizing the trajectory, followed by an explanation of the specific mathematical tools used. Finally, a brief description of Onera's FELIN is also covered.

3.1 Propulsion discipline

The propulsion module uses interpolated data from the open-source software Chemical Equilibrium with Applications (CEA) to calculate the specific impulse and nozzle parameters of the engines. CEA has been developed from the 1950s at the NASA Glenn research center and allows the calculation of thermodynamic and transport properties for different mixtures of reactants. The most common propellants for launchers are available in CEA. In this work, the oxidizer is assumed to be liquid oxygen (LOx) and the fuel is kerosene (RP-1).

The same models are used for the first and second stages. The module is based on a set of design variables $\boldsymbol{z}_p$, consisting of the pressure at the combustion chamber (P_c), the pressure at the nozzle exit (P_e), the mixture ratio (O/F) and the thrust at vacuum (T_{vac}) for the first and second stages. Hence,

$\boldsymbol{z}_p \in \mathbb{R}^8$. To avoid flow separation in the first stage nozzle the value of P_{e_1} is lower bounded to 0.4 times the atmospheric pressure at sea level. This threshold is known as the Summerfield criterion [19].

CEA is used to compute the values of the isentropic coefficient at the throat (γ_t), the temperature at the chamber (T_c) and the molecular mass (M_c) as a function of P_c and O/F using the problem type "Rocket" and the option "Frozen". Meaning that the reaction products at the combustion chamber and nozzle exit are assumed to be identical. According to Sutton and Biblarz [20], this approach usually leads to an underestimation of performance between 1 to 4%. Even so, this is preferred as a conservative approach. The calls to CEA are replaced by a surrogate model using a bivariate spline on a rectangular mesh. This approach gives access to the partial derivatives required for the optimization process. The plots of the interpolated outputs and their partial derivatives are shown in appendix B. Special care should be taken at modifying the bounds as some attempts to do so lead to noisy partial derivatives and outputs in the past.

The characteristic velocity ($c^\star$) can be computed given a set of design variables and the outputs from the surrogate models

$$c^\star = \eta_{c^\star} \cdot \frac{\sqrt{\gamma_t R T_c}}{\gamma_t \left(\frac{2}{\gamma_t+1}\right)^{\frac{\gamma_t+1}{(\gamma_t-1)2}}} \tag{3.1}$$

Where $R = \frac{8314}{M_c}$ (J/kg-K) is the gas constant per unit weight calculated as a function of the molecular mass at combustion chamber (M_c). γ_t is the sentropic coefficient at the throat, T_c the temperature at combustion chamber and $\eta_{c^\star}$ the $c^\star$ efficiency. Later, the nozzle area ratio (nozzle exit area divided by the throat area) (ϵ) is calculated

$$\epsilon = \frac{\left(\frac{2}{\gamma_t+1}\right)^{\frac{1}{\gamma_t-1}} \left(\frac{P_c}{P_e}\right)^{\frac{1}{\gamma_t}}}{\sqrt{\left(\frac{\gamma_t+1}{\gamma_t-1}\right)\left(1-\left(\frac{P_e}{P_c}\right)^{\frac{\gamma_t-1}{\gamma_t}}\right)}} \tag{3.2}$$

The performance of the propulsion system varies with the atmospheric pressure at which it operates. Nevertheless, this variation is modeled in the trajectory module so that the propulsion discipline can be evaluated at only one atmospheric pressure point per optimization cycle. The atmospheric pressure (P_a) assumed for the Propulsion discipline corresponds to the pressure at vacuum. Hence,the thrust coefficient (C_f) at vacuum reads

$$C_f = \eta_{c_f} \sqrt{\frac{2\gamma_t^2}{\gamma_t - 1} \cdot \frac{2}{\gamma_t + 1}^{\frac{\gamma_t+1}{\gamma_t-1}} \cdot \left(1 - \left(\frac{P_e}{P_c}\right)^{\frac{\gamma_t-1}{\gamma_t}}\right)} + \frac{\eta_{c_f}\epsilon}{P_c} \cdot (P_e - P_a) \tag{3.3}$$

Where $P_a = 0$ (Pa) and η_{c_f} is the thrust coefficient efficiency. The thrust coefficient is a measure of the gain in thrust due to the expansion in the nozzle and typically takes values in the range $[1, 2]$. Accordingly, the specific impulse at vacuum I_{sp} is calculated.

$$I_{sp} = \frac{c^\star \cdot C_f}{g_0} \tag{3.4}$$

For the purpose of this study, the mass flow rate is assumed constant. This corresponds to an engine that can only operate at full throttle without any capability of modifying the throttle level. Nevertheless, the models and scripts were developed thinking of introducing throttle capabilities in the future by assuming a proportional variation of thrust with mass flow rate. The possible variation of

throttle would be considered in the trajectory module. For this reason, the output of the propulsion is defined as $\dot{m}_{max}$ and corresponds to the mass flow rate at full throttle capacity. It reads

$$\dot{m}_{max} = \frac{T}{I_{sp} g_0} \tag{3.5}$$

Afterwards, the throat area (A_t) is computed

$$A_t = \frac{c^\star \dot{m}_{max}}{P_c} \tag{3.6}$$

The nozzle area of one engine (A_e) is calculated as

$$A_e = A_t \cdot \epsilon \tag{3.7}$$

With this value, the total nozzle of one stage A_{e_t} can be found by multiplying A_e by a predefined number of engines for each stage. The total nozzle exit area is constrained by the stage diameter that is calculated in the Mass/Sizing module. As this constraint involves two different disciplines it will be formally defined in a following section. The number of engines is not optimized in this work as it is a discrete variable and would need a different treatment to be included in the optimization process.

3.2 Mass-sizing discipline

The main purpose of the Mass/Sizing discipline is to calculate the dry mass of the first (m_{s_1}) and second (m_{s_2}) stages. It uses the models presented in [6] by Castellini for the first stage and a simplified model for the second stage. This discipline requires the design parameters $\boldsymbol{z}_m$ containing the mass of propellants of the first and second stages, m_{p_1} and m_{p_2}, and the diameter of the vehicle, D. Hence, $\boldsymbol{z}_m \in \mathbb{R}^3$. For simplicity, the two stages and fairing are considered to have the same diameter.

The models for this discipline require as input the thrust at vacuum, which is a design variable shared among all the disciplines and the mixture ratio that is shared with the Propulsion discipline. It also requires the feedback couplings with the trajectory discipline in order to dimension the structures according to the maximum dynamic pressure (q_{max}) and the maximum axial load factor ($n_{ax_{max}}$). In the proposed approach, these Mass-Sizing discipline coupling inputs ($n_{ax_{max}}$, q_{max}) are provided by the system level optimizer along with the design variables. Additional equality constraints are added in order to ensure that at the convergence of the MDO problem, the input couplings ($n_{ax_{max}}$, q_{max}) matched the output couplings given by the trajectory.

The mass models for the first stage consider a cryo-storable propulsion type with RP-1 as fuel and liquid oxygen as oxidizer. The cycle corresponds to a gas generator. The thrust frame and structures are assumed to be built in aluminum. The hanks are disposed in an "intertank" configuration. The thrust vector control is assumed to be hydraulic with a redundancy level of 3. The surface of the inter-stage is fixed at 30 m^2.

The structural mass of the first stage (m_{s_1}) is modeled as a function of the mass of propellants (m_{p_1}), the mixture ratio (O/F_1), the stage diameter (D), the thrust at vacuum of the stage (T_{vac_1}), the number of engines, the maximum dynamic pressure ($q_{dyn_{max}}$) and the maximum axial load factor ($n_{f_{max}}$). The structural mass of the second stage is modeled only as a function of the mass of propellants (m_{p_2}).

3.3 Trajectory discipline

The trajectory discipline is the core of the present work and the way it was modeled follows the structure of an optimal control problem and the capabilities available in Dymos. The guidance of the launcher is performed with a parameterized history of the pitch angle, this strategy is frequently used for the MDAO of LVD as it relies on few design variables. As a simplification, the motion is assumed to happen in the equatorial plane and is modeled in 2D polar coordinates, this reduces the number of ODE to be solved compared to the 3D model that is also singular if spherical coordinates are used. The jettisoning of the first stage and payload fairing for the TSTO vehicle are considered and constrained according to threshold values of dynamic pressure and heat flux. The injection into circular orbit of the payload is done using a Hohmann transfer ascent, neglecting the coast phase between first stage engine cut-off and second stage engine start. The models are easily modifiable to perform direct injection into elliptical orbits. During the trajectory, the aerodynamic loads and drag are computed as a function of velocity and height, based on U.S. Standard Atmosphere models.

In a first step, this section defines the Equations of Motion (EoM), followed by a description of the phases required to model the optimal control problem. In a final step, the models used for the forces experienced by the launcher are addressed.

3.3.1 Equations of motion

The 2D trajectory model is depicted in figure 5 and can be represented by a set of 5 differential equations in polar coordinates (equations 3.8 to 3.12).

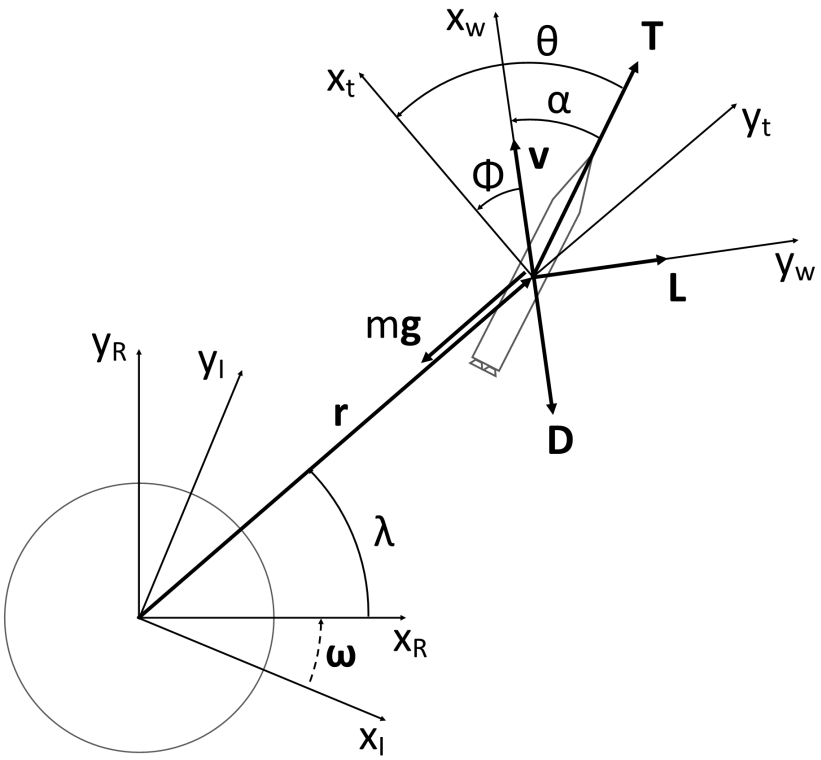

Figure 5: Reference frames and variables for 2D EoM

where ω is Earth's angular velocity, λ is the longitude, $\mathbf{r}$ is the position vector from Earth's center to a fixed point in the launcher (e.g., the geometric center), θ is the pitch angle, ϕ is the flight path angle, α is the angle of attack, m is the mass of the launcher at any given time, $\mathbf{g}$ is the acceleration of gravity, $\mathbf{D}$ is the drag force, $\mathbf{L}$ is the lift force, $\mathbf{T}$ is the thrust force and $\mathbf{v}$ the velocity vector. The reference frames

involved are the Inertial Earth Centered reference frame ($\mathcal{F}_I$), the Rotating Earth Centered reference frame ($\mathcal{F}_R$), the Local Vertical - Local Horizontal reference frame ($\mathcal{F}_t$) and the Wind reference frame ($\mathcal{F}_w$).

$$\dot{r} = v\sin(\phi) \tag{3.8}$$

$$\dot{\lambda} = \frac{v\cos(\phi)}{r} \tag{3.9}$$

$$\dot{v} = \frac{-D + T\cos(\theta - \phi)}{m} + \left(-g + \omega^2 r\right)\sin(\phi) \tag{3.10}$$

$$\dot{\phi} = \frac{L}{mv} + \frac{T\sin(\theta - \phi)}{mv} + \frac{\left(\omega^2 r - g\right)\cos(\phi)}{v} + 2\omega + \frac{v\cos(\phi)}{r} \tag{3.11}$$

$$\dot{m} = \dot{m}_{max} \tag{3.12}$$

The demonstration of the equations of motion is included in appendix A.

3.3.2 Phases

Continuity requirements must be respected within a phase of an optimal control problem when gradient-based optimization is used. However, discontinuities can be introduced between phases. In this work, the definition of new phases is dictated by 3 groups of events. The first group is related to discontinuities in the equations used to parameterize the pitch angle guidance (for instance by switching from a vertical lift-off to a pitch-over maneuver). The second one is related to discontinuities in mass due to jettisoning events (e.g., jettisoning of the first stage), and the last one is related to the finding of a maximum value for the dynamic pressure. In the implementation of the orthogonal collocation methods of Dymos, the initial time and duration of each phase are handled by the optimizer. In the following parts of this section, phases are defined according to the discontinuities in the pitch angle guidance. The subdivisions due to discontinuities in mass and the finding of maximum dynamic pressure are specified within them.

- **Time management**
  3 quantities related to the time of a phase p, are defined. Their use is of special importance for the parameterized pitch angle guidance.
  - Initial time (t_{p_0}): time at which the phase starts.
  - Phase duration (Δt_p): duration of the phase.
  - Phase time (t_p): time spanned during the phase ($t_{p_0} \leq t_p \leq t_{p_0} + \Delta t_p$).
- **Lift-off (lo)**
  Launcher trajectories are a two-point boundary value problem, this implies that the trajectory starts at a fixed state that specifies the initial location of the launcher (corresponding to that of the launch pad), its initial flight path angle of 90° corresponding to a vertical direction, and its initial speed. The initial mass of the launcher corresponds to the Gross Lift-off Weight (GLOW) and is unknown at the beginning of the optimization process. The final boundary conditions of the two-point boundary value problem are dictated by the desired orbit and will be defined in the last phase with the help of constraints.

  As for the guidance law for the pitch angle, it just ensures that vehicle remains vertical following the expression

$$\theta = 90° \tag{3.13}$$

The end of the lift-off phase happens when the launcher clears the launchpad and can start changing its attitude safely. In some launchers as Ariane 5 and Falcon 9 this happens when the altitude is near $200m$. To model this, a two-sided constraint is used to ensure that the radius at the end of the lift-off phase, r_{lo_f}, lies in the interval $[R_e + 150, R_e + 2000]$ m, where R_e is the radius of Earth at the equator.

The initial conditions, $\boldsymbol{g}_0$, during this phase are

$$r = R_e \tag{3.14}$$
$$v = 10^{-3}(\text{m/s}) \tag{3.15}$$
$$\lambda = 0(\text{deg}) \tag{3.16}$$
$$\phi = 90(\text{deg}) \tag{3.17}$$

- **Linear Pitch-over (lpo)**
  For defining this phase it is first necessary to define the concept of angle of attack. In figure 5, the angle of attack is represented by the Greek letter, α, and it is defined as the difference between the pitch angle, θ, and the flight path angle ϕ. During this phase, the angle of attack increases linearly by a quantity $\Delta\theta_{lpo}$. Consequently, the guidance law reads

$$\theta = \phi + \frac{\Delta\theta_{lpo}}{\Delta t_{lpo}}(t_{lpo} - t_{lpo_0}) \tag{3.18}$$

  It is important to notice that at the early stage of the flight the air density is high, thus high angles of attack result in high aerodynamic loads. As launchers are build up in a lightweight fashion those loads are kept low by bounding $\Delta\theta_{lpo}$ to low values.

- **Exponential Decay of Pitch (edp)**
  After the linear pitch over phase, the vehicle returns to a state of zero angle of attack to reduce aerodynamic loads. This process is done by following an exponential law that is defined as follows

$$\theta = \phi - \Delta\theta_{lpo} \cdot \exp\left(\frac{-3 \cdot (t_{edp} - t_{edp_0})}{\Delta t_{edp}}\right) \tag{3.19}$$

  The linear pitch over and exponential decay of pitch phases are shown in figure 6 for different values of $\Delta\theta_{lpo}$.

- **Gravity Turn**
  During the previous phases the flight path angle changed, allowing the vehicle to have a downrange velocity component. At this phase, the vehicle continues to increment its downrange velocity component thanks to the action of gravity that "bends" the trajectory. The law that governs this phase simply reads as

$$\theta = \phi \tag{3.20}$$

  Making zero the angle of attack. The gravity turn is divided into 3 phases to find the maximum dynamic pressure and to allow a discontinuity in mass due the jettisoning of the first stage. All the new phases use Eq. 3.20 for the pitch angle guidance.

  – **Gravity Turn A (gta)**
    This phase ends when the maximum dynamic pressure is reached. The point of maximum dynamic pressure happens when the partial derivative of dynamic pressure w.r.t time is zero. The dynamic pressure is defined as

$$q = \frac{1}{2}\rho \cdot v^2 = \frac{1}{2}\rho(h(r(t))) \cdot v(t)^2 \tag{3.21}$$

Where h is the current altitude and ρ is the air density at h. Its partial derivative w.r.t. time reads

$$\frac{\partial q}{\partial t} = \frac{1}{2}\frac{\partial \rho}{\partial h}\frac{\partial h}{\partial r}\frac{\partial r}{\partial t} \cdot v^2 + \rho \cdot v \cdot \frac{\partial v}{\partial t} = \dot{q} \tag{3.22}$$

A constraint is used to ensure that $\dot{q} = 0$ at the end of this phase. It is important to notice that for the typical shape of q, $\dot{q} = 0$ also happens at two minimum points. This can cause problems when random initialization are used for the optimization process.

- **Gravity Turn B (gtb)**
  This phase follows Gravity Turn A and ends when the first stage is jettisoned. A constraint will be defined in a following section to ensure that the ejected mass corresponds to that of the first stage as calculated in the Mass/Sizing discipline.
- **Gravity Turn C (gtc)**
  After the first stage is jettisoned the vehicle continues the gravity turn maneuver. No coast phase is modeled between first stage jettison and second stage engine start. A new phase can be defined in the future to properly model such event. The end of this phase is reached when the dynamic pressure reaches values lower than 1 kPa. This is ensured with a constraint at the end of the phase. Once this is ensured, the vehicle can perform an aggressive change in attitude for the coming phase.

- **Exo-atmospheric / Bilinear Tangent Law**
  The insertion of the payload into the elliptical transfer orbit (ETO) by the second stage is done using a bilinear tangent law. This guidance technique was shown to be optimal by Brusch in [21] when a uniform gravitational field is assumed. The bilinear tangent law starts by an aggressive change in pitch by an angle $\Delta\theta_{btl}$. Such maneuver is only possible because at the height where it starts the air density is really low, hence the aerodynamic forces experienced by the launcher are low despite the high angle of attack that can be created. In the bilinear tangent law equation (3.23), the initial angle θ_{btl_0} is calculated as the sum of the pitch angle at the end of gravity turn phase, θ_{gt_f}, and $\Delta\theta_{btl}$. If the bounds of $\Delta\theta_{btl}$ are wide, θ_{btl_0} could take values above 90 deg, creating an undesired jump in the tangent functions. The bilinear tangent law finishes with a pitch angle θ_{btl_f} that can take values different from zero. This means that at its final orbit, the local axis can be misaligned from w.r.t. the Local Vertical - Local Horizontal reference frame (F_t). The shape of the trajectory is controlled by the design variable ξ, that can take values in the interval $[-1, 1]$. The parameter a controls the amplitude that θ can reach during the phase and it is fixed at 100. Figure 7 shows a plot of θ for different values of ξ, and fixed values of θ_{btl_f} and θ_{btl_0} at -5 and 45 degrees correspondingly.

$$\theta = \arctan\left(\frac{a^{\xi}\tan\theta_{btl_0} + \left(\tan\theta_{btl_f} - a^{\xi}\tan\theta_{btl_0}\right)\frac{t_{btl}-t_{btl_0}}{\Delta t_{btl}}}{a^{\xi} + \left(1-a^{\xi}\right)\frac{t_{btl}-t_{btl_0}}{\Delta t_{btl}}}\right) \tag{3.23}$$

The payload fairing jettison is assumed to happen during the bilinear tangent law. Hence, the phase is divided in to two phases that use the same equation for the pitch angle guidance (Eq. 3.23), namely the Exo-atmospheric A and B phases. It is important to notice that such equation is function of the total duration of the bilinear tangent law, Δt_{btl}. This duration reads

$$\Delta t_{btl} = \Delta t_{btla} + \Delta t_{btlb} \tag{3.24}$$

Additionally, the time of the during the Exo-atmospheric B phase must be shifted to account for the duration of the Exo-atmospheric A phase. From an application perspective, this required to overrun the time-management in Dymos and replace it by design variables.

- **Exo-atmospheric A (btla)**
  This phase corresponds to a portion of the second stage flight during which the fairing is

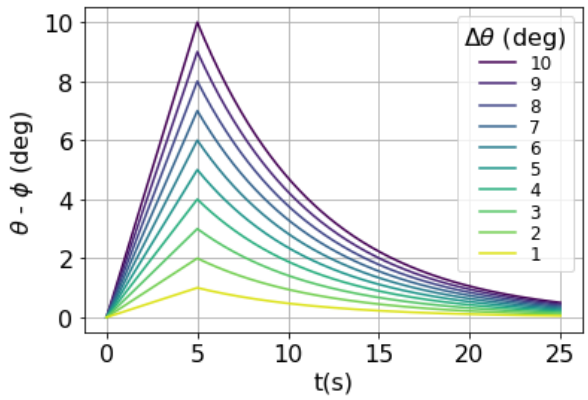

Figure 6: Pitch over (linear and exponential)

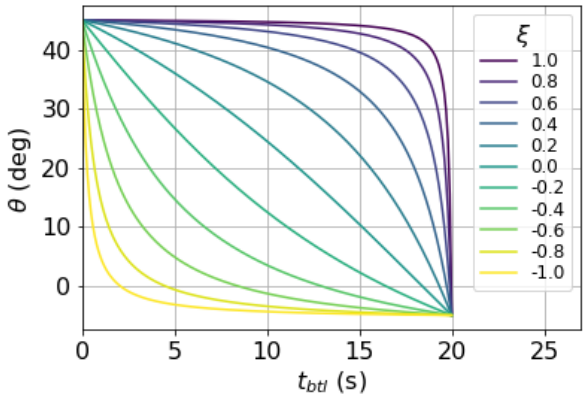

Figure 7: Bilinear tangent law phase

still protecting the payload from aerodynamic loads. An upper bound constraint was used to delimit the end of the phase once the heat flux is below 1135 (W/m^2). The payload fairing is jettisoned at the end of this phase. As in Dymos 0.15.0 it was not possible to define the magnitude of a desired discontinuity of a state, a constraint was used to ensure that the jettisoned mass corresponds to that of the fairing.

– **Exo-atmospheric B (btlb)**
The payload is inserted into orbit using a Hohmann Transfer Ascent (HTA). This implies that the second stage performs two separate burns. The end of this phase corresponds to the end of the first burn of the second stage and it is defined using constraints on the orbital elements to ensure that the vehicle is inserted first into an elliptical transfer orbit. The coast phase of the second stage and the second/final burn to circularize the orbit are not included in the time span where the EoM (Eqs. 3.8 to 3.12) are solved using the orthogonal collocation methods. But are rather considered in the optimization process using analytical expressions.
In a first step, the inertial velocity (v_I) is found based on the velocity (v) and assuming that the vehicle is launched from a radius equivalent to R_e. In the code developed for this research project, a change of initial condition for the radius r at the lift-off phase would lead to an erroneous calculation of (v_I). The code should be updated in the future to account for different initial conditions.
The specific energy (E) and momentum (H) for the elliptical transfer orbit are found with the following equations

$$E = \frac{v_I^2}{2} - \frac{\mu}{r} \tag{3.25}$$

$$H = r \cdot v_I \cdot \cos(\phi) \tag{3.26}$$

Followed by the calculation of the semi-major (a) axis and the eccentricity (e)

$$a = -\frac{\mu}{2 \cdot E} \tag{3.27}$$

$$e = \sqrt{1 + 2 \cdot \frac{H^2 \cdot E}{\mu^2}} \tag{3.28}$$

And the radius at apogee and perigee

$$r_a = a \cdot (1 + e) \tag{3.29}$$

$$r_p = a \cdot (1 - e) \tag{3.30}$$

A lower bound constraint is used to ensure that the radius at perigee of the elliptical transfer orbit (ETO) does not fall below $R_e + 145$ km. The radius at apogee is also lower bounded with a constraint to ensure that it matches the radius of the desired circular orbit. This is only valid if the objective function is to minimize the GLOW of the vehicle as the optimizer will search for the lowest energy orbit possible. If a different objective function is used these constraints should be revisited.

The Δv necessary during the second burn of the second stage to circularize the orbit is calculated with the following expression for a Hohmann transfer

$$\Delta v_2 = -\sqrt{\frac{2 \cdot \mu \cdot r_p}{r_a \cdot (r_a + r_p)}} + \sqrt{\frac{\mu}{r_a}} \tag{3.31}$$

The final mass of the vehicle, after orbit circularization, is found with Tsiolkovsky's equation

$$m_f = m_{btlb_f} \cdot \exp\left(-\frac{\Delta v_2}{I_{sp_2} \cdot g_0}\right) \tag{3.32}$$

Where m_{btlb_f} is the mass at the end of the Exo-atmospheric B phase and m_f is the analytic calculation for the mass of the vehicle after circularization. A constraint is later used to ensure that m_f accounts for the dry mass of the second stage and the payload mass.

The different phases are depicted in figure 8.

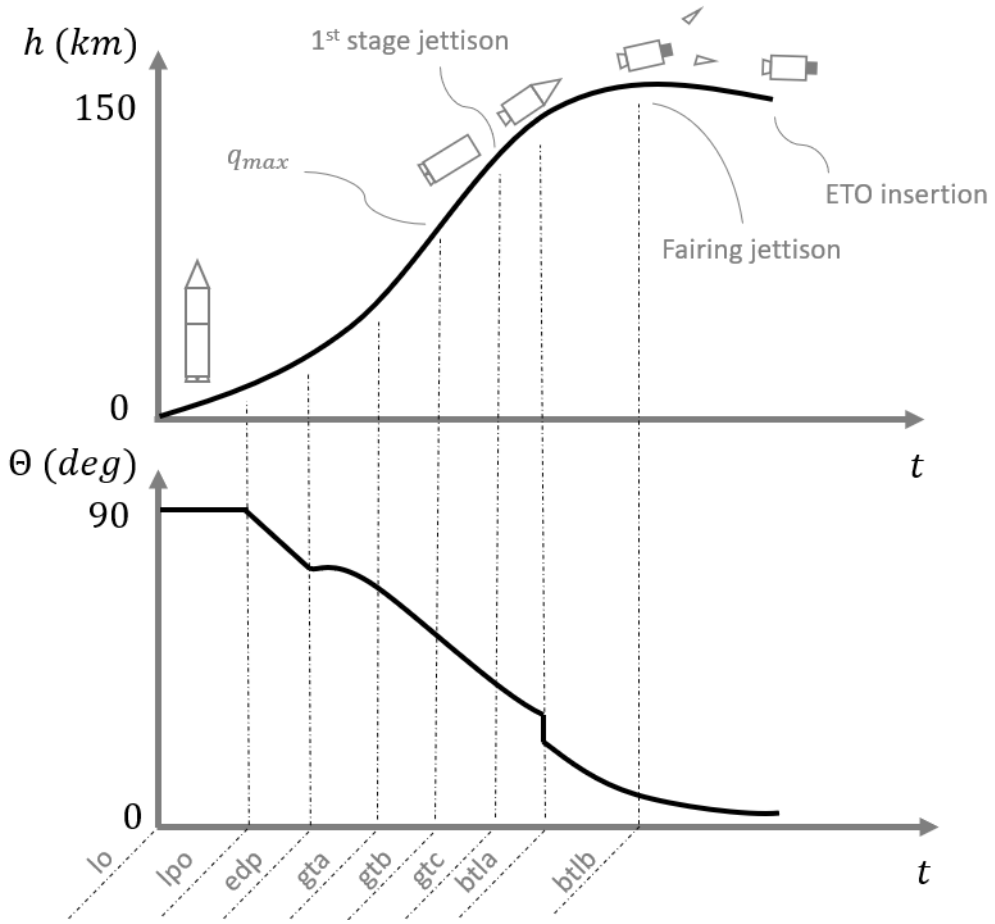

Figure 8: Phases of the trajectory depicted with the altitude (h) and the pitch angle (θ).

3.3.3 Force models

- **Gravity**
  Newton's law of universal gravitation is used to model the gravitational field of a spherical planet. The magnitude of the gravitational acceleration according to Newton's model reads

$$g = \frac{\mu}{r^2} \tag{3.33}$$

  Where μ is the product of the universal gravitational constant an the celestial body's mass and r is the radius of the point where the magnitude of the gravitational acceleration is to be calculated. For the case of planet Earth, the constant μ equals $3.986004 \times 10^{14} \quad m^3/s^2$.

- **Atmosphere**
  One of the purposes of the atmospheric model is to provide the value of air density as a function of altitude that is used to calculate the aerodynamic forces experienced by the vehicle. Other outputs of the atmospheric model are the local Mach number that is used to determine the vehicle's drag coefficient and the local atmospheric pressure that is used to model the variation of thrust due to the expansion of the combustion products at the exit of the nozzle. The model used in this work is based on the 1976 and 1962 U.S. Standard Atmospheres as described in [17] and shown in appendix C. It ignores the variation of atmospheric parameters with time, latitude, longitude and winds.

  The model is sampled in the range $[0, 2000]$ km of altitude every km and an Akima interpolator is fitted in order to have access to the derivatives required for the optimization procedure.

- **Aerodynamics**
  Typically, aerodynamic forces experienced by a launcher are decomposed into the drag force, D, the lift force, L, and lateral force. As launchers usually have one plane of symmetry and the sideslip angle is close to zero during the trajectory the lateral forces can be ignored. In a launcher like the space shuttle that has big lifting surfaces, the lift force plays an important role. On the other hand, for cylindrical-shape launchers as the one considered in this work, lift forces are really low due to the low lift characteristics of cylindrical shapes and the fact that during an important part of the early phases of the trajectory (that is when the dynamic pressure is significant) the angle of attack is zero. For these reasons, lift forces will also be ignored, leaving only the drag force.

  Drag force is modeled according to the expression

$$D = q \cdot C_d \cdot S \tag{3.34}$$

  where C_d is the drag coefficient based on transversal area , S the transverse area and q the dynamic pressure. The drag coefficient changes as a function of the angle of attack and the Mach number for a given launcher. To model this changes it is necessary to perform a wind tunnel testing campaign or multiple computational fluid dynamics (CFD) simulations. Usually the angle of attack is kept low when the dynamic pressure is high, therefore the variation of drag coefficient with angle of attack can be ignored.

  The Cd curve of an Ariane 5 launcher is reported in [22]. In order to use it, sample points were taken to later fit a Pchip interpolator as shown in figure 9. Pchip interpolators are C^1 continuous.

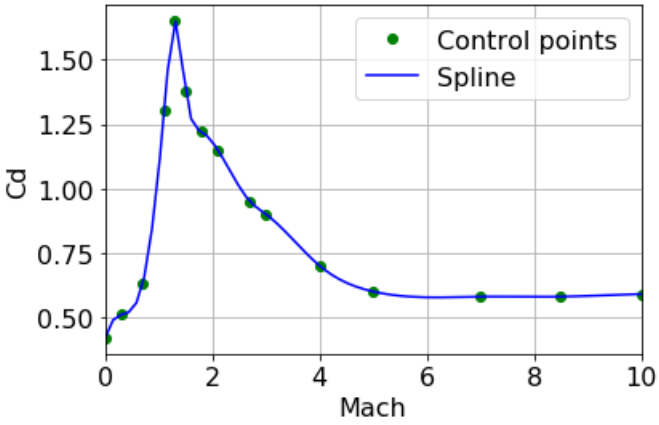

Figure 9: M Vs. C_d curve for an Ariane 5 launcher

As can be seen, the maximum drag coefficient is reached when the vehicle is trans-sonic. The data reported only extends up to $M = 10$, but as in some simulations the launcher exceeds this Mach number it was decided to extend the M Vs. C_d curve up to $M = 15$ with a constant C_d value equal to that of $M = 10$. It is important to notice that such big M values are reached at high altitudes where the atmospheric pressure is low, thus the drag force is low too.

- **Thrust losses**

  The thrust, T, is modeled as function of the thrust at vacuum,T_{vac} the total nozzle exit area, A_{e_t}, and the local atmospheric pressure, p_a, as expressed by the following equation

$$T = T_{vac} \cdot \delta - A_{e_t} \cdot p_a \tag{3.35}$$

  The throttle parameter δ is assumed to be 1 for constant throttle. In the future, this parameter can be used as a dynamic control to model a throttle capacity based on the modification of mass flow rate. In that case, the mass flow rate reads

$$\dot{m} = \dot{m}_{max} \cdot \delta \tag{3.36}$$

  At altitudes near sea level, the atmospheric pressure is maximum and therefore the losses in thrust are maximum. Once the launcher is at vacuum the losses are null and the thrust T is maximum.

- **Loads**
  The calculation of the dynamic pressure was already described. This quantity is calculated for all the phases along the with the axial load factor.

$$n_{ax} = \frac{T - D \cdot \cos(\theta - \phi)}{m \cdot g_0} \tag{3.37}$$

  The maximum load factor is assumed to happen at the end of the first stage flight, *i.e.* at the end of the Gravity Turn B phase. The maximum dynamic pressure (q_{max}) and the maximum load factor ($n_{ax_{max}}$) are feedback to the Mass/Sizing discipline in order to dimension the structures accordingly.

3.3.4 Constraints

The constraints belonging to the trajectory that were described in the different phases are summarized in this section

Equality Constraints ($\boldsymbol{h}_t$):

$\dot{q}_{gta_f} = 0$: $\dot{q}$ at the end of Gravity Turn A

Inequality Constraints ($\boldsymbol{g}_t$):

$R_e + 150\text{m} \leq r_{lo_f} \leq R_e + 2000\text{m}$	: Radius at the end of Lift-off phase
$q_{gtc_f} \leq 1\text{kPa}$	: Dynamic pressure at the end of Gravity Turn C
$q_{heat_{btl_f}} \leq 1135\text{W m}^2$	: Heat flux at the end of Exo-atmospheric A phase
$r_a \geq 400\text{km}$	: Radius at apogee of elliptical transfer orbit
$r_p \geq 145\text{km}$	: Radius at perigee of elliptical transfer orbit

3.4 Legendre-Gauss-Lobatto orthogonal collocation

In orthogonal collocation methods a discretization of the states in time is done in order to transcribe the optimal control problem into a NLP problem. The Dymos package implements the Legendre-Gauss-Radau (LGR) pseudospectral method and the Legendre-Gauss-Lobatto (LGL) collocation methods, allowing to use them both for different phases of the same trajectory. The LGL transcription requires a 2 step evaluation of the ODE per optimizer iteration and the LGR requires only one. Nevertheless, LGR relies on a higher number of variables and constraints and its use with final boundary constraints requires an special treatment as the endpoint of the trajectory is free. In this work, only the LGL of order 3 is used and a performance comparison with higher order LGL and LGR transcriptions is left as a proposal for future work.

In Dymos, the optimal control is divided in the 8 phases defined in section 3.3.2. At the same time, each phase is divided into a different amount of segments that can be specified by the user. For the LGL transcription of order 3, a segment has a state discretization node at the each of its two extremes and a collocation node at the center as shown in figure 10. In an uncompressed transcription, two adjacent segments are bounded by using a continuity constraint,in such a way that the state discretization nodes at the junction point for each segment take the same values. Continuity constraints are also used to bound two adjacent phases.

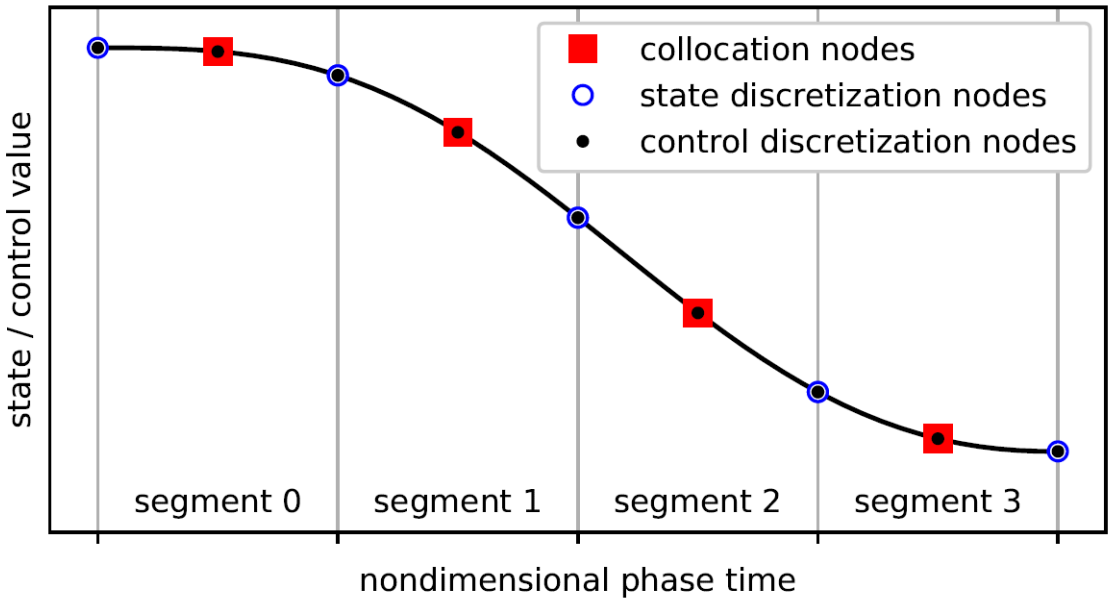

Figure 10: Legendre-Gauss Lobatto transcription of order 3. Taken from [3]

The LGL transcription allows the optimization of dynamic controls via the control discretization nodes. This nodes are defined at the same locations in time as the state discretization nodes and the collocation nodes, resulting in 3 control discretization nodes per segment. Dynamic controls are not implemented in this work, nonetheless, the models are built up to account for throttle as a dynamic control so that it can be easily modifiable in the future.

Hence, the vector of state variables $\boldsymbol{x}_t$ of the optimal control problem of section 2.2.1 is transformed into a discrete set, $\boldsymbol{x}_{t_a}$, containing the state discretization nodes $\boldsymbol{x}_{t_d}$ and the collocation nodes $\boldsymbol{x}_{t_c}$. The time is also discretized to form t_a, this set is completely defined by the initial time t_{p_0} and the phase duration Δt_p of all phases.

As defined in [3], the first step for the optimization consists in an input from the user providing an initial guess for $\boldsymbol{x}_{t_d}$, the time parameters controlling t_a, and the design variables $\boldsymbol{z}_t$. Then, the dynamics (Eq. 2.6) are evaluated to obtain the state rates at the state discretization nodes following

$$\dot{\boldsymbol{x}}_{t_d} = \boldsymbol{f}_{ode}[\boldsymbol{x}_{t_d}, t_d, \boldsymbol{z}_t] \tag{3.38}$$

Using $\dot{\boldsymbol{x}}_{t_d}$ and $\boldsymbol{x}_{t_d}$, an Hermite polynomial is fitted so that the value of the states and the state rates can be approximated at the collocation nodes by using

$$\boldsymbol{x}_{t_c} = \frac{2}{t_{seg}}[A_d]\boldsymbol{x}_{t_d} + [B_d]\dot{\boldsymbol{x}}_{t_d} \tag{3.39}$$

$$\boldsymbol{x}'_{t_c} = [A_i]\boldsymbol{x}_{t_d} + \frac{t_{seg}}{2}[B_i]\dot{\boldsymbol{x}}_{t_d} \tag{3.40}$$

Where $[A_i]$ and $[B_i]$ are the Hermite interpolation matrices, $[A_d]$ and $[B_d]$ are the Hermite differentiation matrices and t_{seg} is the duration of the segment of the current node. A value for the state rate at the discretization node can also be obtained by evaluating the dynamics (Eq. 2.6) with the approximated state values $\dot{\boldsymbol{x}}_{t_c}$

$$\dot{\boldsymbol{x}}_{t_c} = \boldsymbol{f}_{ode}[\boldsymbol{x}_{t_c}, t_c, \boldsymbol{z}_t] \tag{3.41}$$

The difference between the derivative at the collocation node obtained by evaluating the dynamics, $\dot{\boldsymbol{x}}_{t_c}$, and the approximated derivative obtained with the Hermite polynomial $\dot{\boldsymbol{x}}'_{t_c}$ represents the collocation defect. This defect is formulated as an equality constraint following

$$\boldsymbol{\Delta} = \boldsymbol{x}'_{t_c} - \frac{t_{seg}}{2}\dot{\boldsymbol{x}}_{t_c} = \boldsymbol{0} \tag{3.42}$$

Considering the segment i of a state with N segments, the continuity constraints for the state values read $\boldsymbol{C}_x(\boldsymbol{x}_d) = \boldsymbol{0}$. And considering the phase p of a state with P phases the time continuity constraints to link the time read $\boldsymbol{C}_t(t_d) = \boldsymbol{0}$.

56 segments were used distributed in a non-uniform manner among the 8 phases of the trajectory. The state discretization nodes corresponding to the initial boundary conditions specified in equations 3.14 to 3.17 are fixed, hence are not handled by the optimizer. The initial time of the Lift-Off phase is fixed at 0 seconds.

Transcription bill	
Number of states :	5
Number of segments (N) :	56
Number of boundary conditions :	4
Number of state discretization node values :	$2 \times 56 \times 5 = 560$
Number of time values :	$2 \times 8 - 1 = 15$
Number of defect constrains:	$56 \times 5 = 280$
Number of continuity constraints:	$(56 - 1) \times 5 + (8 - 1) = 282$

Regarding the pseudospectral transcription, the optimizer handles a set of variable state discretization node values $\boldsymbol{x}_{t_d} \in \mathbb{R}^{560}$, and the variable time values $t_t \in \mathbb{R}^{15}$ controlling the time at all nodes t_a. The optimizer must also satisfy the totality of equality constraints corresponding to continuity ($\boldsymbol{C}_x$ and $\boldsymbol{C}_t$), defects ($\boldsymbol{\Delta}$) and boundary constraints ($\boldsymbol{g}_0$). Hence, the total number of equality constraints related to the trajectory transcription is $\boldsymbol{R}_t \in \mathbb{R}^{566}$.

The transcription constraints $\boldsymbol{R}_t$ are only guaranteed to be satisfied at the convergence of the optimizer. This means that during the iterations of the optimization the trajectory is not feasible. Furthermore, during the iterations the trajectory does not even have a physical sense. Once the optimizer converges an IVP can be solved with a Runge-Kutta method using the obtained results (t_t, $\boldsymbol{z}_t$, $\boldsymbol{u}_t$) and the initial boundary conditions of an orthogonal collocation method to verify that they really follow the dynamics. As the states and time were discretized, it is also important to verify that the chosen number of segments lead to the desired accuracy of the solution. With this aim. Dymos implements grid refinement algorithms that can vary the number of segments used to discretized the trajectory and their order to reach a desired accuracy. Such convergence analysis was not performed but it is proposed as a future work.

3.5 Constraints for feedforward couplings

Some of the feedforward couplings were defined as constraints as this methods fits the architecture of Dymos and sometimes it was the only solution that was found. This concerns the jettisoning events, the mass of propellants of each stage and the nozzle exit area.

Jettisoning events are a discontinuity in the mass state. In the version of Dymos that was used for this project (0.15.0) the magnitude of a discontinuity of a state cannot be defined, this made it necessary to use constraints in order to define the amount of mass being jettisoned. The mass discontinuity for the first stage jettison corresponds to the dry mass of the first stage (m_{s_1}) that was calculated in the Mass/Sizing discipline, hence it can be written

$$0 \geq m_{s_1} - (m_{gtb_f} - m_{gtc_0}) \tag{3.43}$$

Where m_{gtb_f} is the mass at the end of the Gravity Turn B phase and m_{gtc_0} is the mass at the beginning of the Gravity Turn C phase. Even though an inequality constraint is used, the optimizer drives it to an equality value if the objective function is to minimize GLOW. If a different objective function is used, this constraint must be revisited.

In a similar way the mass jettisoned for the payload fairing must correspond to the input from the user for this value, m_{plf}. Hence, the constraint reads

$$0 \geq m_{plf} - (m_{btla_f} - m_{btlb_0}) \tag{3.44}$$

Where m_{btla_f} is the mass at the end of the Exo-atmospheric A phase and m_{btlb_0} is the mass at the beginning of the Exo-atmospheric B phase. This values come from collocation nodes and belong to $\boldsymbol{x}_{t_d}$. In a similar way to the first stage jettison, this inequality constraint only works if the objective function is to minimize GLOW.

The final mass after orbit circularization is also constrained in such a way that it accounts for the dry mass of the second stage m_{s_2} and the payload mass m_d, following the equation

$$0 \geq m_f - (m_{s_2} + m_d) \tag{3.45}$$

The mass of propellants consumed for each stage during the trajectory must also coincide with the mass of propellants used in the Mass/Sizing module. Another two inequality constraints are used to guarantee this

$$0 \geq m_{p_1} - (m_{lo_0} - m_{gtb_f}) \tag{3.46}$$

$$0 \geq m_{p_2} - (m_{gtc_0} - m_f - m_{plf}) \tag{3.47}$$

Where m_{lo_0} is the mass at the beginning of the Lift-Off phase, m_{gtb_f} is the mass at the end of the Gravity Turn B phase, m_{gtc_0} is the mass at the beginning of the Gravity Turn C phase, m_{btlb_f} is the mass at the end of the Exo-atmospheric B phase and m_{plf} is the mass of the payload fairing. Again, the optimizer drives this constraints to zero when the objective is to minimize GLOW. A diagram depicting the evolution of the mass of the vehicle during the jettisoning events is shown in figure 11.

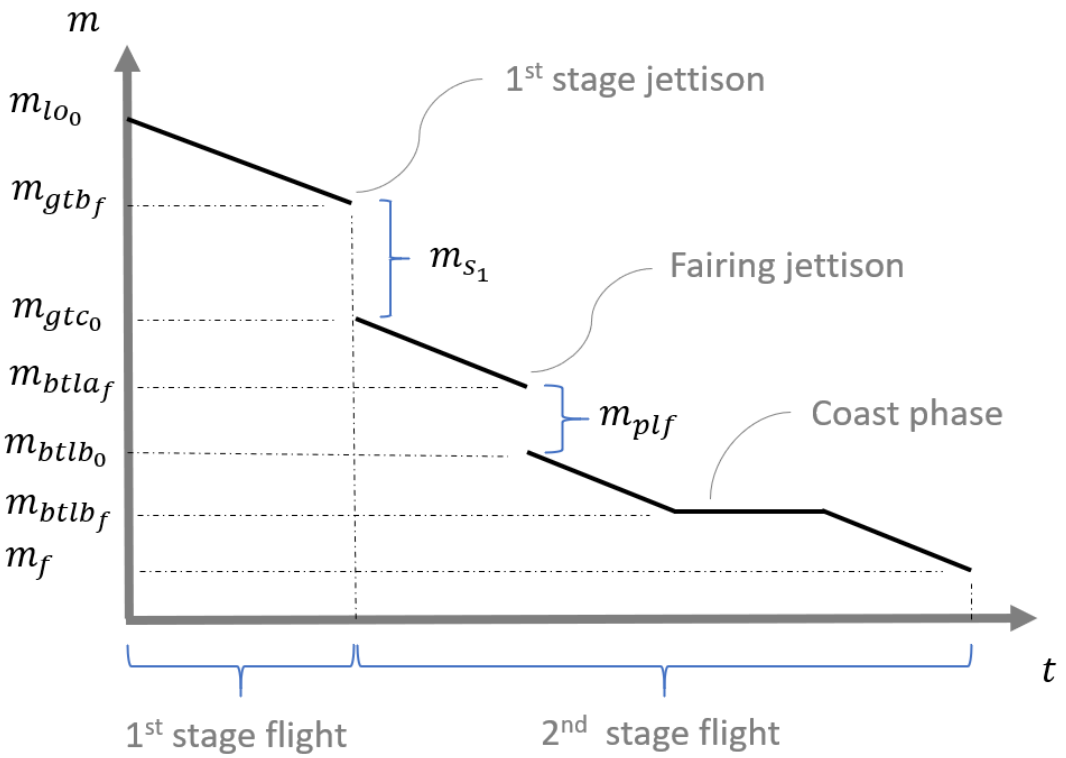

Figure 11: Mass jettison events during the different phases of the trajectory.

The nozzle exit area must be constrained in such a way that the nozzles fit in the cross-section of the stage. The total nozzle exit area of each stage was constrained to be less than the 64% of the cross-section area of the stage. Following the constraints

$$0 \geq \frac{\pi}{4} D^2 - A_{e_{t_1}} \tag{3.48}$$

$$0 \geq \frac{\pi}{4} D^2 - A_{e_{t_2}} \tag{3.49}$$

The totality of the constraints associated to feed forward couplings are grouped in the vector $\boldsymbol{g}_{ff} \in \mathbb{R}^7$.

3.6 Variation All-At-Once MDO formulation for feedback couplings

The feedback couplings between the Mass/Sizing and the Trajectory discipline represent a high computational cost in the traditional MDF formulations due to the use of a nested MDA solver. IDF formulations do not require to solve the MDA at each iteration, but the relative increase in the number of variables and constraints that are managed by the optimizer banishes any possible advantage in terms of computational cost. In orthogonal collocation methods the optimal control problem is transcribed into highly dimensional NLP problem, if the same NLP solver is used as optimizer for the MDO problem the management of the feedback coupling variables through extra variables and constraints instead of using the nested MDA solver does not increase significantly the size of the NLP problem.

The All-At-Once approach for the feedback couplings proposed in this work in conjunction with the orthogonal collocation transcription implies that the multidisciplinary optimization, the multidisciplinary

feasibility and feasibility of the trajectory are handled by the same NLP solver. Removing the necessity of an MDA solver and replacing the time marching ODE solvers necessary for the solving of the dynamics by simple ODE evaluations thanks to the orthogonal collocation transcription. Both, the AAO-like approach and the orthogonal collocation transcription share the characteristic of guaranteeing feasibility only at the convergence of the optimizer. This means that in the proposed method, the disciplinary feasibility of the trajectory, the multidisciplinary feasibility, the feasibility of the optimization and the optimality of the problem are only guaranteed at the convergence of the optimizer.

In this work the maximum axial load factor and the maximum dynamic pressure represent the feedback coupling between the Mass/Sizing and trajectory disciplines. Both q_{max} and $n_{ax_{max}}$ are outputs of the trajectory. One extra variable is created for each of them and is handled by the optimizer, this new variables called $q_{max_{fb}}$ and $n_{ax_{max_{fb}}}$ are used by the Mass/Sizing module and are grouped in the vector $\boldsymbol{y}_{fb}$. A constraint is used to ensure that at the optimizer convergence they take the same values as q_{max} and $n_{ax_{max}}$, respectively. The constraints read

$$0 \geq q_{max_{fb}} - q_{max} \tag{3.50}$$

$$0 \geq n_{ax_{max_{fb}}} - n_{ax_{max}} \tag{3.51}$$

When the objective is to minimize GLOW, this constraints are driven to zero by the optimizer. Both constraints are grouped in the vector $\boldsymbol{g}_{fb}$.

3.7 Mathematical formulation of proposed strategy

The MDAO of the launch vehicle can be written as a unique NLP problem using the orthogonal collocation transcription and the AAO-like approach for the feedback couplings. For the purpose of this work, the objective is to minimize the GLOW. In the orthogonal collocation transcription the GLOW corresponds to a unique node in the discretization set $\boldsymbol{x}_{t_d}$, namely the initial mass during during the Lift-off phase. Hence

$$f = m_{lo_0} \tag{3.52}$$

The state discretization nodes of the trajectory and the variables controlling the time are considered as the state variables of the MDO problem, consequently

$$\boldsymbol{x} = \boldsymbol{x}_{t_d} \cup t_t \quad , \quad \text{where} \quad \boldsymbol{x}_{t_d} \in \mathbb{R}^{560} \quad \text{and} \quad t_t = \begin{bmatrix} t_{lpo_0} \\ t_{edp_0} \\ t_{gta_0} \\ t_{gtb_0} \\ t_{gtc_0} \\ t_{btla_0} \\ t_{btlb_0} \\ \Delta t_{lo} \\ \Delta t_{lpo} \\ \Delta t_{edp} \\ \Delta t_{gta} \\ \Delta t_{gtb} \\ \Delta t_{gtc} \\ \Delta t_{btla} \\ \Delta t_{btlb} \end{bmatrix} \tag{3.53}$$

The coupling variables are those corresponding to the AAO-like approach for the feedback couplings

$$\boldsymbol{y} = \boldsymbol{y}_{fb} \quad , \quad \text{where} \quad \boldsymbol{y}_{fb} = \begin{bmatrix} q_{max_{fb}} \\ n_{ax_{max_{fb}}} \end{bmatrix} \tag{3.54}$$

And the design variables correspond to those of the 3 disciplines, hence

$$\boldsymbol{z} = \boldsymbol{z}_p \cup \boldsymbol{z}_m \cup \boldsymbol{z}_t \quad , \quad \text{where} \quad \boldsymbol{z}_p = \begin{bmatrix} P_{c_1} \\ P_{c_2} \\ P_{e_1} \\ P_{e_2} \\ O/F_1 \\ O/F_2 \\ T_{vac_1} \\ T_{vac_2} \end{bmatrix} \quad , \quad \boldsymbol{z}_m = \begin{bmatrix} m_{p_1} \\ m_{p_2} \\ D \end{bmatrix} \quad \text{and} \quad \boldsymbol{z}_t = \begin{bmatrix} \Delta\theta_{lpo} \\ \Delta\theta_{btl} \\ \theta_{btl_f} \\ \xi \end{bmatrix} \tag{3.55}$$

The optimizer has to satisfy a set of inequality constraints composed by trajectory constraints, and the constraints formulated for the couplings

$$\boldsymbol{g} = \boldsymbol{g}_t \cup \boldsymbol{g}_{ff} \cup \boldsymbol{g}_{fb} \tag{3.56}$$

One equality constraint is used in the trajectory discipline, hence

$$\boldsymbol{h} = \boldsymbol{h}_t \quad , \quad \text{where} \quad \boldsymbol{h}_t = \dot{q}_{gta_f} = 0 \tag{3.57}$$

The equality constraints formulated for the orthogonal collocation transcription in the trajectory discipline correspond to the residuals, therefore

$$\boldsymbol{R} = \boldsymbol{R}_t \tag{3.58}$$

Finally, the MDAO problem can defined as

$$\begin{aligned} \text{minimize} \quad & f = m_{lo_0} \\ \text{with respect to} \quad & \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \\ \text{subject to:} \quad & \\ \text{inequality constraints} \quad & \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \leq \boldsymbol{0} \qquad (3.59) \\ \text{equality constraints} \quad & \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \leq \boldsymbol{0} \qquad (3.60) \\ \text{residuals} \quad & \boldsymbol{R} = \boldsymbol{0} \qquad (3.61) \end{aligned}$$

Where $\boldsymbol{x} \in \mathbb{R}^{575}$, $\boldsymbol{y} \in \mathbb{R}^2$, $\boldsymbol{z} \in \mathbb{R}^{15}$, $\boldsymbol{g} \in \mathbb{R}^{13}$, $\boldsymbol{h} \in \mathbb{R}^1$ and $\boldsymbol{R} \in \mathbb{R}^{566}$.

3.8 Propagation of analytic derivatives and optimizer

The resulting optimization problem is solved using a second order gradient-based optimizer. This optimizer does a numerical approximation of the Hessian and requires only first order derivative information.

This work utilizes the OpenMDAO environment [23], an open-source python library that uses chain rule to propagate derivative information through an MDAO process. Finite-difference and complex step are two numerical methods allowing to approximate the first order derivative information. The use of this methods in OpenMDAO is simple as the user does not have to provide the partial derivatives for each discipline, but they are rather calculated internally by the software. Nevertheless, this numerical approaches can be affected by noise, leading to a failure of the optimization or excessive computing time. In this work, the derivative information is provided using analytic expressions. This requires the user to define the partial derivatives of each discipline but with the benefit of not being affected by noise and faster computing times.

The NLP solver used for the resulting optimization problem is SLSQP. Its name stands for sequential least squares programming. This solver uses the derivatives for the objective function and constraints provided by OpenMDAO and solves the constraint optimization problem by evaluating the Karush–Kuhn–Tucker (KKT) conditions to determine its optimality. OpenMDAO also has interfaces with more performant NLP solvers as SNOPT, but SLSQP was used because of its open-source access.

3.9 FELIN

The FELIN framework was used to compare the results of the proposed optimization strategy and asses its performance. FELIN was developed by Onera and uses the OpenMDAO environment. This framework uses an MDF-like formulation associated to a global evolutionary optimizer called Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES). A single-shooting strategy is implemented to solve the associated optimal control problem in the trajectory discipline and a fixed point iterator is used to solve the MDA linked to the feedback couplings from the trajectory to the mass/sizing discipline. In FELIN, the control variables of the trajectory discipline are handled at the system level.

4 Results and discussion

A case study of a TSTO vehicle was performed, with the mission of injecting 11 tons of payload into a circular orbit of 400 km height. The radius at perigee of the elliptical transfer orbit was constrained to be above 145 km. The proposed optimization using Dymos relies on the SLSQP solver that performs a local search. Therefore, multiple executions with different initialization were performed in order to distinguish the global minimum from other local minimum. The design variables ($\boldsymbol{z}$) and the coupling variables ($\boldsymbol{y}$) were set to take random values within their bounds. The variables controlling the time of the phases ($\boldsymbol{t}_t$) were also randomized within their bounds but the state variables ($\boldsymbol{x}_{t_d}$) were kept invariable and unbounded. The values for the 560 state discretization nodes were defined using the linear interpolation methods of Dymos.

Figure 12 shows the optimization time and status of 25 different runs. The precision of SLSQP was set to stop the optimization when the cost function (GLOW) could not be improved by more than 100 kg. A limit in the number of iterations performed by SLSQP was set to 600 to finish earlier the runs that were taking excessive time. 4 runs converged to the same local minimum of 359.1 tons within the precision of the solver. As this was the smaller GLOW found among the iterations that successfully converged, this is referred to as the global minimum. 3 runs converged to 3 different local minimums. This highlights the importance of randomizing the initial guess as multiple local minimum exist. 11 runs failed after the optimizer found positive directional derivatives, this could be linked to the fact the state variables were kept unbounded and the optimizer was performing an exploration among a wide range of values that lead to the erratic behavior. It could also be linked to the discontinuity in the tangent function of the bilinear tangent law when the initial angle (θ_{btl_0}) takes values above 90 deg. Another possible reason is the 3 the condition on the derivative of the dynamic pressure to finish the Gravity Turn B phase as this derivative can take a value of zero in 3 different instants where only 1 correspond to

the maximum and the other 2 correspond to minimums. Finally, 7 runs reached the maximum iterations limit and it cannot be concluded about their convergence status, although it can be noticed that they took more than 20 % of extra time to execute when compared to the runs that converged or failed.

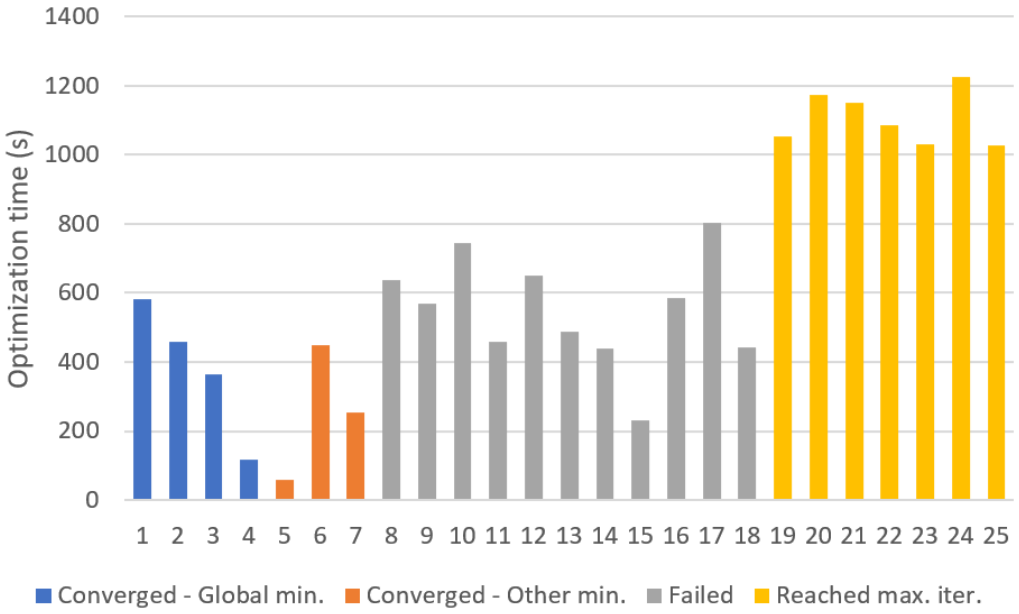

Figure 12: Optimization time for random initialization of proposed methodology in Dymos

The same bounds for the design variables were used to run FELIN 1 time, as it relies on the CMA-ES optimizar that performs a global search. Hence, its unique run is likely to converge close to the global optimum. 12 individuals were used at each iteration of the optimizer. The initialization of the coupling varaibles for the MDA was set close to expected values.

The evolution of GLOW as a function of the number of iterations is shown in figure 13. During the first runs, the non-feasible trajectories are penalized with a high cost (not shown in the plot). Around the iteration number 150 the optimizer reaches feasible solutions and after 400 iterations finds a solution that is only improved slightly until the limit of 1000 iterations. Assuming that the last iteration corresponds to the global minimum found with FELIN, the iteration corresponding to a GLOW that is 100 kg away from the global minimum is iteration number 926. This is taken as the reference iteration for the performance comparison with Dymos.

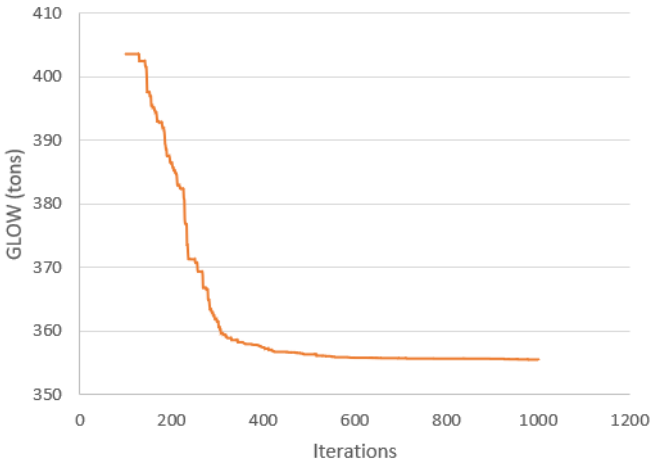

Figure 13: Convergence for the unique run of FELIN

The state history of the optimal solutions obtained with Dymos and FELIN is shown in figure 14. In the same figure, the results obtained with the Simulate methods from Dymos are shown. This method has the purpose of assessing the accuracy of the solution obtained with the LGL transcription. It uses a numerical integrator to propagate the dynamics based on the initial conditions and the design, time and coupling variables of the optimal solution. As the state history obtained with the Simulate method resembles the one obtained with the LGL transcription of Dymos, it means that the chosen discretization was good enough to led to a solution that models the desired dynamics. The height state (h) takes close shapes for the solutions obtained with Dymos and FELIN, but the velocity (V), the mass (m) and the flight path angle state histories present differences. 3 main causes could be leading to this differences. The first one is that the atmospheric models used in Dymos and FELIN are different, the second one is that FELIN considers a short coast phase of 2 seconds after first stage jettison that is neglected in Dymos. This short instant with zero trust can be noticed in figure 15. Finally, the multiple numerical methods that are used in both approaches still need to be tuned to a desired accuracy to allow a proper comparison between both methods.

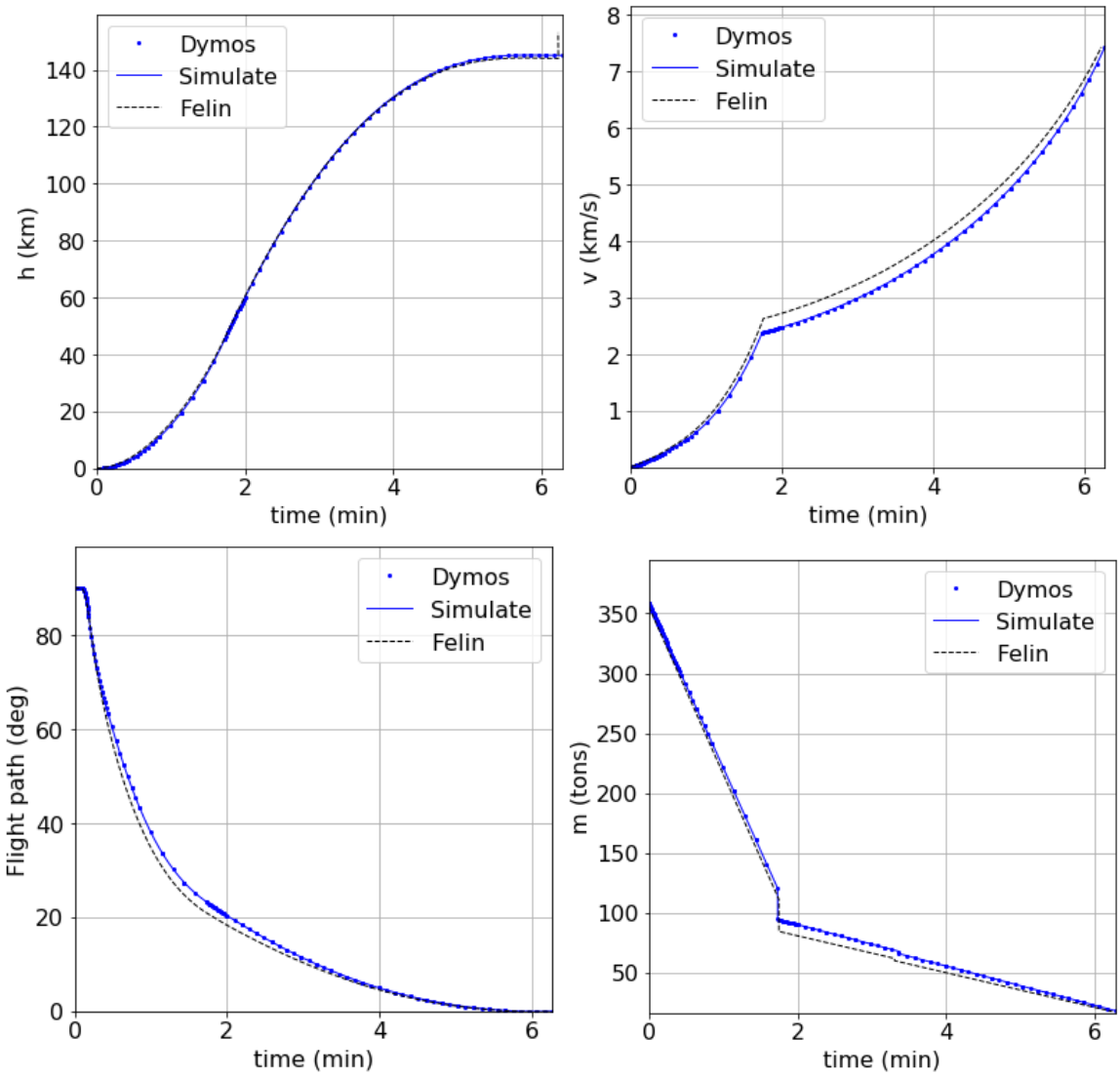

Figure 14: Comparison of the state history of the optimal solutions

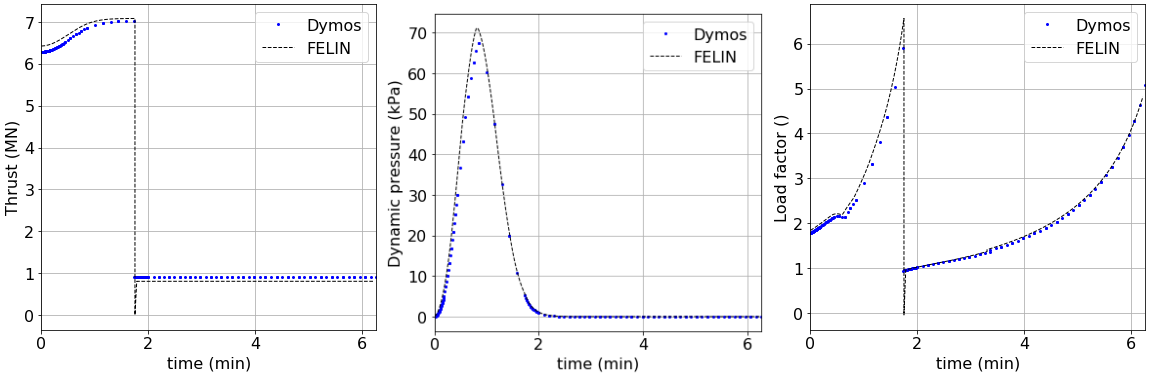

Figure 15: Comparison of the loads history of the the optimal solutions

A comparison of the design variables of the optimal solution obtained with Dymos and FELIN was done and is shown in the table below

Design Variables							
Discipline	Variable	Units	Lower	Upper	Dymos	Felin	Error
Propulsion	T_{vac_1}	N	5 000 000	15 000 000	7 026 653.2	7 084 316.2	8.14×10^{-3}
	T_{vac_2}	N	100 000	1 400 000	911 354.5	804 667.2	1.33×10^{-1}
	O/F_1		2	4	2.3026	2.3298	1.17×10^{-2}
	O/F_2		2	4	2.3445	2.3556	4.71×10^{-3}
	P_{c_1}	Pa	6 000 000	10 000 000	10 000 000.0	9 953 515.2	4.67×10^{-3}
	P_{c_2}	Pa	6 000 000	10 000 000	10 000 000.0	9 999 345.6	6.54×10^{-5}
	P_{e_1}	Pa	40 530	200 000	53 760.9	62 725.4	1.43×10^{-1}
	P_{e_2}	Pa	1	10 000	3 540.8	3 439.2	2.95×10^{-2}
Mass/Sizing	m_{p_1}	kg	230 000	280 000	238 461.2	244 349.4	2.41×10^{-2}
	m_{p_2}	kg	50 000	75 000	74 853.7	65 174.2	1.49×10^{-1}
	D	m	1	5	3.8066	3.6189	5.19×10^{-2}
Trajectory	$\Delta\theta_{lpo}$	deg	1	10	7.9985	9.2866	1.39×10^{-1}
	$\Delta\theta_{btl}$	deg	-60	60	2.1142	-	-
	θ_{btlf}	deg	-20	20	0.8422	0.77	9.38×10^{-2}
	ξ		1	-1	-0.152	-0.286	4.69×10^{-1}

The average error in the design variables is 4%. The maximum error is presented in the shape parameter ξ of the bilinear tangent law with a value of 46.9%.

From a performance point of view, the GLOW corresponding to the global optimum of Dymos is 359.08 tons, which is almost 3.5 tons higher than the GLOW obtained with FELIN of 355.6. This difference could also be due to the same reasons causing the errors in the state history. The best run in Dymos reached convergence in 0.03 h, but as Dymos uses a local optimizer, multiple runs were made with different initialization. The table below shows the total time required for the 25 runs in Dymos. It increases to 4.4 h and most of the computational cost is associated to failed runs and runs that reached the maximum number of iterations. On the other hand, FELIN uses a global optimizer and only one run was made, taking 1.88 hours to reach the defined convergence point.

	Dymos Global optimum Best run	**Dymos 25 runs**	**Felin Unique run**
GLOW (tons)	359.08	359.08	355.6
Function evaluations	169	31 035	51 019
Optimization time (h)	0.03	4.4	1.88

In the future, a method to compare Dymos and FELIN has to be defined as the use of local and global optimizers poses the question of how many runs should be performed with each method to guarantee that they have reached a global optimum.

5 Conclusions and perspective

A new methodology for the MDAO of launch vehicles was proposed using a pseudospectral method for optimal control. It is based on the SLSQP optimizer that drives the whole MDO problem thanks to the transcription of the optimal control problem using a Legendre-Gauss-Lobatto transcription of order 3 and the AAO-like approach to replace the feedback couplings. The propagation of analytic partial derivatives through the disciplines is also a key enabler of the presented methodology. The current model presents a high sensitivity to random initializations as only 7 out of 25 reached convergence under the defined optimization settings. Runs that did not reach convergence and failed runs represent the highest portion of the computational cost. A comparison with an existing methodology was done to verify the accuracy of the new optimization procedure. A difference of 1.0% in GLOW was found between the runs in Dymos and FELIN as well as differences in the state histories. These errors may be related to different atmospheric models, the absence of a coast phase in Dymos and the tuning of the accuracy of the different numerical methods that are involved.

Future work could be focused on the implementation of the same atmospheric models in Dymos an FELIN as well as on the addition of the short coast phase to Dymos. The sensitivity to the initial guess of Dymos could be solved by using better bounds on the variables handled by the optimizer and the addition of bounds for the state discretization nodes. Higher performance algorithms as SNOPT could be used for the same purpose.

References

[1] M. Balesdent and L. Brevault, "Multidisciplinary Design Optimization, application to aerospace vehicle design - MAP 554." 2020.

[2] A. V. Rao, "A SURVEY OF NUMERICAL METHODS FOR OPTIMAL CONTROL," *Advances in the Astronautical Sciences*, vol. 135, no. 1, pp. 497–528, 2009.

[3] R. D. Falck and J. S. Gray, "Optimal Control within the Context of Multidisciplinary Design, Analysis, and Optimization," in *AIAA Scitech 2019 Forum*, American Institute of Aeronautics and Astronautics, 2019.

[4] M. Balesdent, N. Bérend, P. Dépincé, and A. Chriette, "A survey of multidisciplinary design optimization methods in launch vehicle design," *Struct Multidisc Optim*, vol. 45, pp. 619–642, May 2012.

[5] L. Brevault and M. Balesdent, "Framework for Evolutive Launcher optImizatioN." https://github.com/l-brevault/FELIN.

[6] F. Castellini, "MULTIDISCIPLINARY DESIGN OPTIMIZATION FOR EXPENDABLE LAUNCH VEHICLES," 2012.

[7] J. R. Martins and A. Ning, "Engineering Design Optimization." Jan. 2021.

[8] J. T. Betts, "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, 1998.

[9] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. Advances in Design and Control, Society for Industrial and Applied Mathematics, Jan. 2010.

[10] A. L. Herman and B. A. Conway, "Direct optimization using collocation based on high-order Gauss-Lobatto quadrature rules," *Journal of Guidance, Control, and Dynamics*, 1996.

[11] D. Garg, M. Patterson, C. Darby, C. Francolin, G. Huntington, W. Hager, and A. Rao, "Direct Trajectory Optimization and Costate Estimation of General Optimal Control Problems Using a Radau Pseudospectral Method," June 2012.

[12] E. S. Hendricks, R. D. Falck, and J. S. Gray, "Simultaneous Propulsion System and Trajectory Optimization," AIAA AVIATION Forum, American Institute of Aeronautics and Astronautics, June 2017.

[13] R. D. Falck, J. S. Gray, and B. Naylor, "Parallel Aircraft Trajectory Optimization with Analytic Derivatives," in *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, American Institute of Aeronautics and Astronautics, June 2016.

[14] L. E. Briese, K. Schnepper, and P. Acquatella B., "Advanced modeling and trajectory optimization framework for reusable launch vehicles," Mar. 2018.

[15] G. D. C. B. de Volo, "Vega Launchers' Trajectory Optimization Using a Pseudospectral Transcription," 2017.

[16] Z. Wang and Z. Wu, "Six-DOF trajectory optimization for reusable launch vehicles via Gauss pseudospectral method," *Journal of Systems Engineering and Electronics*, Apr. 2016.

[17] A. Tewari, *Atmospheric and Space Flight Dynamics: Modeling and Simulation with MATLAB and Simulink*. Boston, Mass.: Birkhäuser, 2007.

[18] A. Pagano and E. Mooij, "Global Launcher Trajectory Optimization for Lunar Base Settlement," in *AIAA/AAS Astrodynamics Specialist Conference*, (Toronto, Ontario, Canada), Aug. 2010.

[19] M. Summerfield, C. R. Foster, and W. C. Swan, *Flow Separation in Overexpanded Supersonic Exhaust Nozzles*, vol. 24. AMER INST AERONAUT ASTRONAUT 1801 ALEXANDER BELL DRIVE, STE 500, RESTON, VA . . . , 1954.

[20] G. P. Sutton and O. Biblarz, *Rocket Propulsion Elements*. John Wiley & Sons, Dec. 2016.

[21] R. Brusch, "Bilinear tangent yaw guidance," in *Guidance and Control Conference*, Guidance, Navigation, and Control and Co-Located Conferences, American Institute of Aeronautics and Astronautics, Aug. 1979.

[22] A. Pagano, *Global Launcher Trajectory Optimization for Lunar Base Settlement*. PhD thesis, TU DELFT, May 2010.

[23] J. S. Gray, J. T. Hwang, J. R. Martins, K. T. Moore, and B. A. Naylor, "OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization," *Struct Multidisc Optim*, Apr. 2019.

A Derivation of 2D EoM for a rotating planet

The launcher is modeled as a particle moving in the equatorial plane of a rotating planet according to Newton's Second Law of dynamics. An Inertial Earth Centered reference frame, $\mathcal{F}_I \coloneqq \{x_I, y_I\}$, is defined as inertial reference frame. In this reference frame it can be written for a variable mass system

$$\mathbf{f} = m\mathbf{a_I} \tag{A.1}$$

Where m is the mass of the launcher at any given time, $\mathbf{a_I}$ is the inertial acceleration and $\mathbf{f}$ is the vector comprised of aerodynamic forces (drag ($\mathbf{D}$) and lift ($\mathbf{L}$)), thrust force ($\mathbf{T}$) and gravity force ($m\mathbf{g}$).

The position of the launcher is better expressed with respect to a fixed point on Earth, this is achieved with the Rotating Earth Centered reference frame, $\mathcal{F}_R \coloneqq \{x_R, y_R, z_R\}$, this frame is the one used by the reader to know its own position in Earth and has unitary vectors $\hat{\mathbf{I}}$, $\hat{\mathbf{J}}$ and $\hat{\mathbf{K}}$. To give a notion of the attitude of the vehicle the Local Vertical - Local Horizontal reference frame, $\mathcal{F}_t \coloneqq \{x_t, y_t, z_t\}$, is defined with its center lying at a fix point on the launcher (Ex.: the geometric center) and with unitary vectors $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$ and $\hat{\mathbf{k}}$. Finally, a fourth reference frame orientated with respect to the launcher velocity component is defined to better express the aerodynamic forces and is called the Wind reference frame $\mathcal{F}_w \coloneqq \{x_w, y_w\}$. All of the aforementioned reference frames are shown in figure 5.

For a vector $\mathbf{p}$ expressed in terms of the unitary vectors of the reference frame $\mathcal{F}_t$, the transformation matrix expressed in A.2 can be used to transform it and express it in terms of the unitary vectors of reference frame $\mathcal{F}_w$.

$$\mathbf{p_w} = \mathbf{R}'_\phi \mathbf{p_t} \quad , \quad \mathbf{R}_\phi = \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \tag{A.2}$$

To deduce the equations of motion some expressions for the change in time of the position $\dot{r}$ and the longitude $\dot{\lambda}$ will be find by using the velocity $\mathbf{v}$ as a firs step. Then the inertial velocity $\mathbf{v_I}$ will be used to find the inertial acceleration $\mathbf{a_I}$ and the necessary transformations will be done to use equation A.1 and finally find expressions for the change of velocity $\dot{v}$ and flight path angle $\dot{\phi}$.

From figure 5 a first expression for the velocity of the launcher relative to $\mathcal{F}_R$ but expressed in terms of the unitary vectors of $\mathcal{F}_t$ can be deduced.

$$\mathbf{v} = v(\cos(\phi)\mathbf{i} + \sin(\phi)\mathbf{j}) \tag{A.3}$$

A second expression for $\mathbf{v}$ is obtained by the relation with the linear change of the position vector $\dot{\mathbf{r}}$ and its angular rate $\boldsymbol{\Omega}$. Given that $\mathbf{r}$ can be expressed in terms of $\mathcal{F}_t$ unitary vectors as $r\mathbf{j}$, it can be written

$$\mathbf{v} = \dot{r}\mathbf{j} + \boldsymbol{\Omega} \times (r\mathbf{j}) \tag{A.4}$$

$\boldsymbol{\Omega}$ represents the angular velocity of the reference frame $\mathcal{F}_t$ with respect to $\mathcal{F}_r$ and can be written as $\Omega_x\mathbf{i} + \Omega_y\mathbf{j} + \Omega_z\mathbf{k}$. Solving the cross product it reads

$$\mathbf{v} = \dot{r}\mathbf{j} + \Omega_x r\mathbf{k} - \Omega_z r\mathbf{i} \tag{A.5}$$

By comparison of equations A.3 and A.5, it is obtained

$$\dot{r} = v\sin(\phi) \tag{A.6}$$

$$\Omega_z r = -v\cos(\phi) \tag{A.7}$$

As can be noticed in figure 5 the angular velocity of the reference frame $\mathcal{F}_t$ with respect to $\mathcal{F}_r$ is equal to the rate of change of λ, hence it can be expressed $\mathbf{\Omega} = \dot{\lambda}\mathbf{K}$ or whats is equivalent $\mathbf{\Omega} = -\dot{\lambda}\mathbf{k}$. Using this result and rewriting equation A.4 as

$$\mathbf{v} = \dot{r}\mathbf{j} + \dot{\lambda}r\mathbf{i} \tag{A.8}$$

and equating with A.7

$$\dot{\lambda} = \frac{v}{r}\cos(\phi) \tag{A.9}$$

the equations A.6 and A.9 are obtained and represent the first two EoM.

The inertial velocity $\mathbf{v_i}$ is comprised by the velocity $\mathbf{v}$ and the tangential component due to Earth's angular velocity ω. As the later can be expressed as $\omega\mathbf{K}$ or the equivalent $-\omega\mathbf{k}$, it reads

$$\mathbf{v_I} = \mathbf{v} + \omega r\mathbf{i} \tag{A.10}$$

The inertial acceleration can be find by derivating $\mathbf{v_I}$. This time the angular rate of change is $\omega + \mathbf{\Omega}$ or the equivalent $-(\omega + \dot{\lambda})\mathbf{k}$. Hence it can be written

$$\mathbf{a_I} = \frac{d\mathbf{v_I}}{dt} = \dot{\mathbf{v}} + \omega\dot{r}\mathbf{i} - (\omega + \dot{\lambda})\omega r\mathbf{j} \tag{A.11}$$

taking the derivative of $\mathbf{v}$ as expressed in A.8 and again using $-(\omega + \dot{\lambda})\mathbf{k}$ as the angular rate of change, it reads

$$\dot{\mathbf{v}} = \ddot{r}\mathbf{j} + \ddot{\lambda}r\mathbf{i} + \dot{\lambda}\dot{r}\mathbf{i} - (\omega + \dot{\lambda})\mathbf{k} \times (\dot{r}\mathbf{j} + \dot{\lambda}r\mathbf{i}) \tag{A.12}$$

By using A.12 and simplifying, equation A.11 can thus be rewritten as

$$\mathbf{a_I} = [\ddot{\lambda}r + 2\dot{r}(\dot{\lambda} + \omega)]\mathbf{i} + [\ddot{r} - r(\omega + \dot{\lambda})^2]\mathbf{j} \tag{A.13}$$

By finding the second derivatives of $\mathbf{r}$ and λ

$$\ddot{r} = v\cos(\phi)\dot{\phi} + \sin(\phi)\dot{v} \tag{A.14}$$

$$\ddot{\lambda} = -\frac{v\sin(\phi)\dot{\phi}}{r} + \frac{\cos(\phi)\dot{v}}{r} - \frac{v^2\sin(\phi)\cos(\phi)}{r^2} \tag{A.15}$$

and by replacing in equation A.13, we can rewrite $\mathbf{a_I}$ once again but in matrix form

$$\mathbf{a_I} = \begin{bmatrix} 2\omega v \sin(\phi) - v \sin(\phi)\dot{\phi} + \cos(\phi)\dot{v} + \frac{v^2 \sin(2\phi)}{2r} \\ -\omega^2 r - 2\omega v \cos(\phi) + v \cos(\phi)\dot{\phi} + \sin(\phi)\dot{v} - \frac{v^2 \cos^2(\phi)}{r} \end{bmatrix} \tag{A.16}$$

So far $\mathbf{a_I}$ is expressed in terms of the unitary vectors $\mathbf{i}$ and $\mathbf{i}$ that are the basis of $\mathcal{F}_t$. A final transformation to the Wind reference frame $\mathcal{F}_w$ must be done using the matrix transformation expressed in equation A.2. The new unitary vectors were not explicitly defined but $\mathbf{a_I}$ as a function of them reads

$$\mathbf{a_I} = \begin{bmatrix} -\omega^2 r \sin(\phi) + \dot{v} \\ -\omega^2 r \cos(\phi) - 2\omega v + v\dot{\phi} - \frac{v^2 \cos(\phi)}{r} \end{bmatrix} \tag{A.17}$$

Also in the $\mathcal{F}_w$ reference frame, the forces exerted by the launcher as can be seen in figure 5 can be expressed in matrix form as

$$\mathbf{f} = \begin{bmatrix} -D + T\cos(\theta - \phi) - gm\sin(\phi) \\ L + T\sin(\theta - \phi) - gm\cos(\phi) \end{bmatrix} \tag{A.18}$$

Finally, by applying Newton's second law as expressed in equation A.1, using A.17 and A.18 and solving for $\dot{v}$ and $\dot{\phi}$ we can find the two remaining equations of motion

$$\dot{v} = \frac{-D + T\cos(\theta - \phi)}{m} + \left(-g + \omega^2 r\right)\sin(\phi) \tag{A.19}$$

$$\dot{\phi} = \frac{L}{mv} + \frac{T\sin(\theta - \phi)}{mv} + \frac{(\omega^2 r - g)\cos(\phi)}{v} + 2\omega + \frac{v\cos(\phi)}{r} \tag{A.20}$$

B Bivariate interpolation of CEA outputs

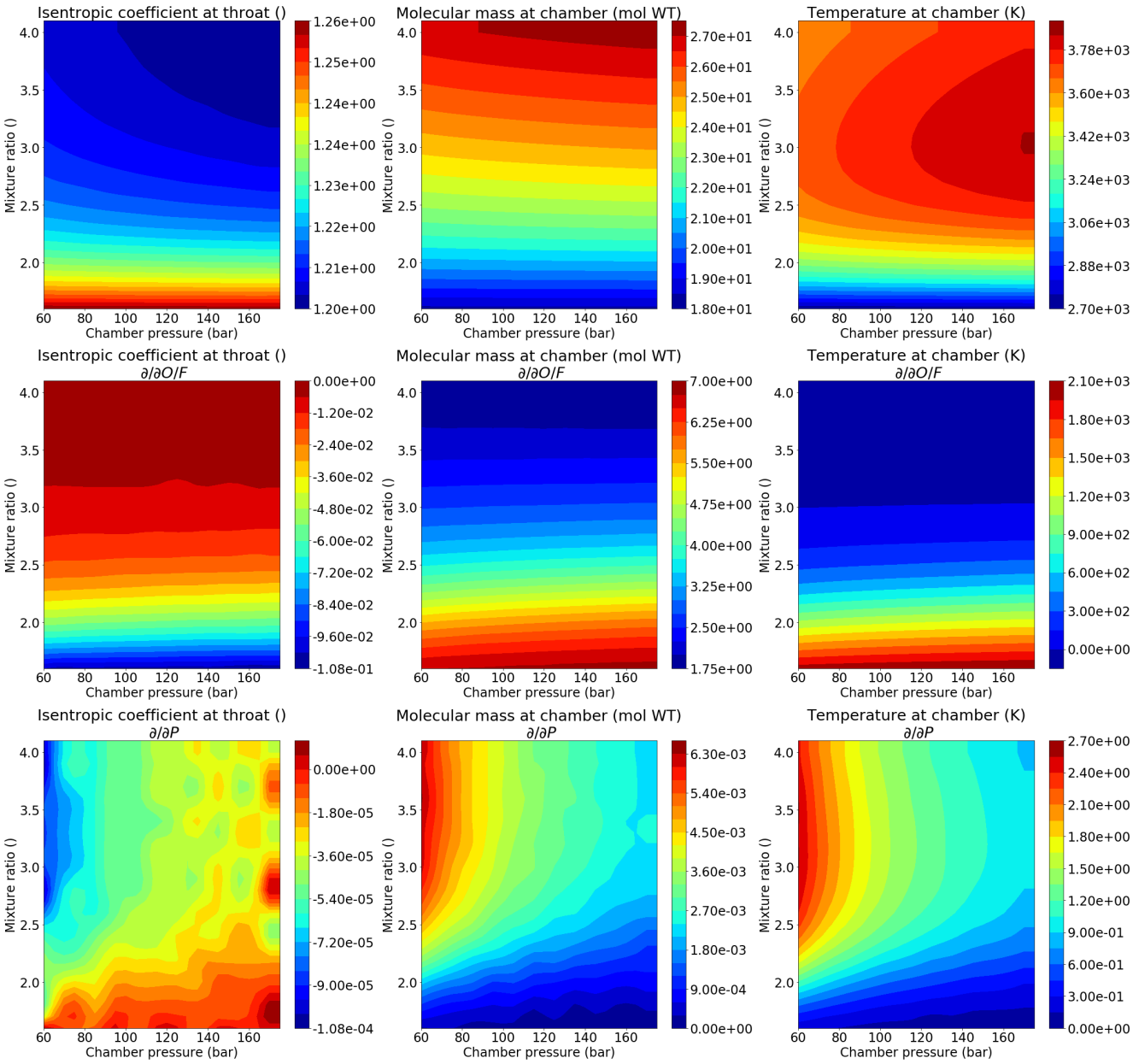

Figure 16: Bivariate interpolation of CEA outputs and their partial derivatives

C Atmospheric model based on the 1976 and 1962 U.S. Standard Atmospheres

This atmospheric model outputs atmospheric parameters by modeling the variation in temperature as a function of height. Height is divided into 21 segments called layers and the rate of change of temperature within a layer is constant but varies from layer to layer. The corresponding tabulated data is shown in table 1.

i	h_i (km)	T_i (K)	R (J/kg · K)	a (K/km)
1	0	288.15	287.0	-6.5
2	11.0191	216.65	287.0	0.0
3	20.0631	216.65	287.0	1.0
4	32.1619	228.65	287.0	2.8
5	47.3501	270.65	287.0	0.0
6	51.4125	270.65	287.0	-2.8
7	71.8020	214.65	287.02	-2.0
8	86	186.946	287.02	1.693
9	100	210.02	287.84	5.0
10	110	257.0	291.06	10.0
11	120	349.49	308.79	20.0
12	150	892.79	311.80	15.0
13	160	1022.2	313.69	10.0
14	170	1103.4	321.57	7.0
15	190	1205.4	336.68	5.0
16	230	1322.3	366.84	4.0
17	300	1432.1	416.88	3.3
18	400	1487.4	463.36	2.6
19	500	1506.1	493.63	1.7
20	600	1506.1	514.08	1.1
21	700	1507.6	514.08	0.0

Table 1: Atmospheric model based on the 1976 and 1962 U.S. Standard Atmospheres

Where h_i is the height above the sea level, T_i is the standard temperature, R is the gas constant and a is the slope that describes the variation of temperature with height across layers, better called thermal lapse rate.

The values of temperature between layers can be found according to the relationship

$$T = T_i + a\,(h - h_i) \tag{C.1}$$

It is worth noticing that in the first layer of this atmospheric model,between 0 and 11 km height, the temperature decreases with a thermal lapse rate of -6.5 K/km but in the second layer, between 11 km and 20 km the temperature does not vary with height, the former is called an isothermal layer.

For those layers whose thermal lapse rate, a, is different from zero, the pressure reads

$$p = p_i \left[1 + \frac{a\,(h - h_i)}{T_i}\right]^{\frac{g_0}{aR}\left[1+\beta\left(\frac{T_i}{a}-h_i\right)\right]} e^{\frac{\beta g_0}{aR}(h-h_i)} \tag{C.2}$$

And for the isothermal layers, the following expression is used

$$p = p_i e^{-\left[\frac{g_0(h-h_i)}{RT_i}\right]\left[1-\frac{\beta(h-h_i)}{2}\right]} \tag{C.3}$$

$$\beta = \frac{2}{r_e} \tag{C.4}$$

Knowing the temperature and pressure, the air density can be calculated as

$$\rho = \frac{p}{RT} \tag{C.5}$$

The speed of sound at the given height is calculated according to the equation

$$a_\infty = \sqrt{\gamma RT} \tag{C.6}$$

Where γ is the adiabatic index assumed as 1.405. Finally the Mach number, M, is defined as

$$M = \frac{v}{a_\infty} \tag{C.7}$$

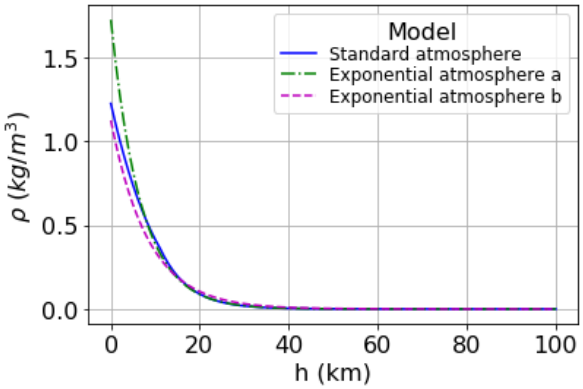

Figure 17: Air density in linear scale

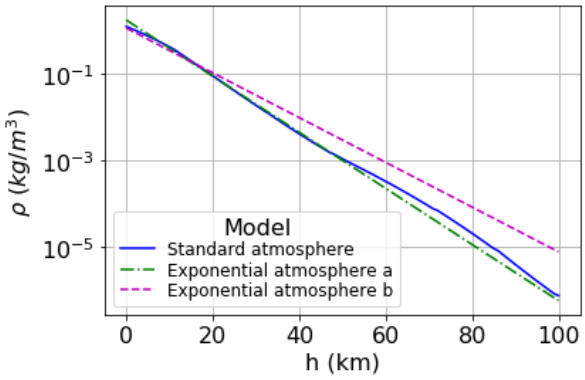

Figure 18: Air density in logarithmic scale

In figures 17 and 18 the result of air density obtained with the model based on the standard atmosphere is compared to simpler exponential atmosphere models where the equation modeling the air density, ρ, can be written as

$$\rho = \rho_{ref} \cdot exp\left(\frac{-h}{h_{scale}}\right) \tag{C.8}$$

Where ρ_{ref} is the reference air density, not necessarily at sea level, h is the height and h_{scale} is the reference scale height. Both ρ_{ref} and h_{scale} can be modified to better adapt the model in a specific zone. In figures 17 and 18, two exponential atmospheric models can be observed. Namely, the exponential atmosphere model a, with $h_{scale} = 6.7 \quad km$ and $\rho_{ref} = 1.725 \quad kg/m^3$ and the exponential atmosphere model b, with $h_{scale} = 8.4 \quad km$ and $\rho_{ref} = 1.225 \quad kg/m^3$. It can be noticed that the exponential atmosphere 'b' fits better the model based on the standard atmospheres at low altitudes where the air density is still high.