Development of Multiphase Radau Pseudospectral Method for Optimal Control Problems

José Eduardo Valério Garrido
jose.garrido@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Portugal

January 2021

Abstract

In this work the flipped-Radau pseudospectral method is employed in order to solve multiphase optimal control problems. The application of interest is the generation of multi-stage rocket trajectories. The method is implemented in the MATLAB tool SPARTAN, developed at DLR, with the use of the NLP solver IPOPT. A relevant numerical example is implemented for validation of the algorithm. The example is appropriate because it contains mutations in the equations of motion from one phase to the next. The problem is composed of three distinct phases and it is concerned with the recovery of the main booster of the orbital launcher Falcon 9 via a boost-back manoeuvre and a vertical landing near the launch site. The algorithm is validated by analysis of the Hamiltonian associated with the trajectory. The results show that the method is able to generate optimal trajectories with accuracy comparable to state of the art solvers.
Keywords: optimal control, Radau pseudospectral method, multi-stage rocket trajectory generation, boost-back and vertical landing.

1 Introduction

Rocket trajectory optimization is profoundly tied to optimal control. Since launch vehicle technology developed, the optimization of rocket trajectories has been made through Pontryagin's Minimum principle [1]. The classical approach to solve optimal control problems (OCPs) is through the calculus of variations, leading to the Hamiltonian boundary-value problem, which is the indirect method approach [2]. But indirect methods are often impractical due to the associated necessity of deriving the problem-wise optimality conditions [3]. In addition, these methods most often comprise convergence issues [4], and thus, lack reliability when a good initial guess is not available.

In contrast, direct methods, and in particular pseudospectral methods, have gained popularity over the past decades since their generalization by Fahroo *et al.* [5, 6]. Pseudospectral methods consist on the transcription of the dynamics based on the roots of an orthogonal polynomial, typically a particular Jacobi polynomial [7]. The constraints are evaluated at each collocation point resulting in a large list of equality and inequality algebraic constraints, accompanied with the algebraic objective function; this list of constraints, referred to as a nonlinear programming (NLP) problem, is passed through an *off-the-shelf* nonlinear optimizer such as IPOPT [8] and the solution is obtained. In essence, in a pseudospectral method, the optimal control problem is transformed into a parametric optimization problem.

Regarding multiphase optimal control, successful efforts have been made in order to implement the feature into software [7]. Multiphase optimal control allows the study of trajectories that might contain necessary state discontinuities or even mutations in the equations of motion. In order to generate trajectories of orbital launchers it is relevant to develop a multiphase algorithm, this way state discontinuities can be accounted for.

In this work, the flipped Legendre-Gauss-Radau pseudospectral method (or, simply, flipped Radau method) is employed to generate optimal trajectories of multi-stage rockets. The flipped Radau method is well suited because it allows endpoint collocation, and it does not present dramatic convergence issues on the dual variables [9]. The foundation for the implementation of the multiphase feature is SPARTAN [10], a MATLAB tool developed at the German Aerospace Centre (DLR) with the objective of solving general purpose optimal control problems.

2 Multiphase Optimal Control Problem

The purpose of a generic multiphase problem formulation is to account for cases in which state discontinuities are expected. In the context of rocket launchers composed of multiple stages, state discontinuities

are expected at stage separation events.

Letting $p = 1, \ldots, P$ be a scalar integer indicating a specific phase, where P is the total number of phases in the problem, a generic multiphase optimal control problem can be formulated as minimizing the cost functional

$$\mathcal{J} = \Phi\left(\boldsymbol{x}(t_0^{(1)}), t_0^{(1)}, \boldsymbol{x}(t_f^{(P)}), t_f^{(P)}\right) + \sum_{p=1}^{P} \left\{ \int_{t_0^{(p)}}^{t_f^{(p)}} \Psi\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) \mathrm{d}t \right\}, \quad (1)$$

associated with the trajectory of the dynamic system

$$\dot{\boldsymbol{x}}^{(p)}(t) = \boldsymbol{f}^{(p)}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right), \quad (2)$$

subject to the path constraints

$$\boldsymbol{h}^{(p)}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) \leq \boldsymbol{0}, \quad (3)$$

the event constraints

$$\boldsymbol{\phi}^{(p)}\left(\boldsymbol{x}(t_0^{(p)}), t_0^{(p)}, \boldsymbol{x}(t_f^{(p)}), t_f^{(p)}\right) \leq \boldsymbol{0}, \quad (4)$$

and the phase linkage conditions

$$\Delta\boldsymbol{x}_{\min}^{(p)} \leq \boldsymbol{\ell}^{(p)} = \boldsymbol{x}(t_0^{(p+1)}) - \boldsymbol{x}(t_f^{(p)}) \leq \Delta\boldsymbol{x}_{\max}^{(p)}, \quad (5)$$

where $\boldsymbol{x}(t) \in \mathbb{R}^{n_x}$ is the state vector, $\boldsymbol{u}(t) \in \mathbb{R}^{n_u}$ is the control vector, and $\Delta\boldsymbol{x}_{\min}^{(p)}$ and $\Delta\boldsymbol{x}_{\max}^{(p)}$ are, respectively, the user-defined lower and upper boundary vectors for the linkage conditions. This formulation indicates that the constraints are not explicit functions of time, therefore, the systems in study are time invariant. Equation 5, which corresponds to the phase linkage conditions, consists of two distinct constraints, the upper bound and the lower bound of the eventual state discontinuities. This allows the specification of several discontinuity scenarios.

2.1 Gaussian Quadrature and Domain Mapping

It is possible to compute the definite integral from –1 to 1 of a given polynomial function, $f(\tau)$, according to the *Gaussian quadrature* where the function is evaluated at N discrete points and a weighted sum of the samples is performed [11, 12]. In order to integrate over a generic interval $t \in [t_0, t_f]$, the domain of the independent variable, t, is mapped into the normalized domain $\tau \in [-1, 1]$. This mapping is linear and it can be expressed as [13]

$$t(\tau) = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2}, \quad \tau \in [-1, 1]. \quad (6)$$

By performing a change of variables, the Gaussian quadrature can be expressed for a generic interval as

$$\int_{t_0}^{t_f} f(t)\,\mathrm{d}t = \frac{t_f - t_0}{2} \sum_{k=1}^{N} w_k f_k, \quad (7)$$

where $f_k = f\left(t(\tau_k)\right)$ is the value of the function evaluated at $t(\tau_k)$, and w_k is a scalar quadrature weight associated with the k^{th} function sample. It is important to note that it is not sufficient to evaluate the function at any given set of N discrete points. For the quadrature to be accurate it is crucial that the distribution of points follows a pattern similar to the distribution of roots of a particular Jacobi polynomial (also called Gauss points) [11, 12].

2.2 Lagrange Polynomial Interpolation and the Differentiation Matrix

In order to compute the first derivative of a polynomial function $\boldsymbol{x}(\tau)$ with respect to $\tau \in [-1, 1]$, a differentiation matrix, $\boldsymbol{A}$, can be built based on a Lagrange polynomial interpolation [7, 14, 15]

$$\boldsymbol{A}_{ki} = \sum_{\substack{l \\ l \neq i}} \frac{1}{\tau_i - \tau_l} \prod_{\substack{j \\ j \neq i,l}} \frac{\tau_k - \tau_j}{\tau_i - \tau_j}, \quad (8)$$

where $\boldsymbol{A}_{ki}$ is a scalar element on the k^{th} row and i^{th} column of matrix $\boldsymbol{A}$, and the derivative operation with respect to a generic domain, $t \in [t_0, t_f]$, is

$$\dot{\boldsymbol{x}}_k = \frac{2}{t_f - t_0} \sum_i \boldsymbol{A}_{ki} \boldsymbol{x}_i, \quad (9)$$

where $\boldsymbol{x}_k$ is the polynomial $\boldsymbol{x}(\tau)$ evaluated at k^{th} sample τ_k. As was the case for Gaussian quadrature, the accuracy of the Lagrange polynomial interpolation, and that of the differentiation matrix, is directly related to the choice of sample points along the domain of τ.

2.3 Root distribution of the flipped Radau polynomial

A good choice of sample points is a proportional mapping of the roots of Legendre-based polynomials, such as the *flipped Radau polynomial*. The flipped Radau polynomial is the result of the difference between two Legendre polynomials of consecutive order, as

$$R_N(\tau) = P_N(\tau) - P_{N-1}(\tau), \quad \tau \in [-1, 1], \quad (10)$$

where $P_N(\tau)$ is the Legendre polynomial of order N. And the Gaussian quadrature weights associated with the roots of the flipped Radau polynomial can be computed by doing a *flip* operation on the weights of the direct Radau, reversing their order [12, 13]

$$w_k = \texttt{flip}\left\{ \frac{1 - \tau_k^{\text{DR}}}{N^2 P_{N-1}^2(\tau_k^{\text{DR}})} \right\}, \quad k = 1, 2, .., N, \quad (11)$$

where τ_k^{DR} is the k^{th} abscissa of the direct Radau roots, N is the degree of the Radau polynomial, and

$P_{N-1}(\tau_k^{\mathrm{DR}})$ is the Legendre polynomial of degree $N-1$ evaluated at τ_k^{DR}.

Figure 1 illustrates the pattern of collocation nodes for a discretization of the domain composed of multiple phases using the flipped Radau pseudospectral method. The multiphase discretization scheme is a simple concatenation of several "single phase" schemes. It can be noticed that the "density" of nodes increases near the phase transition points. One peculiarity of the roots of the Radau polynomial is that they are asymmetric with respect to the origin.

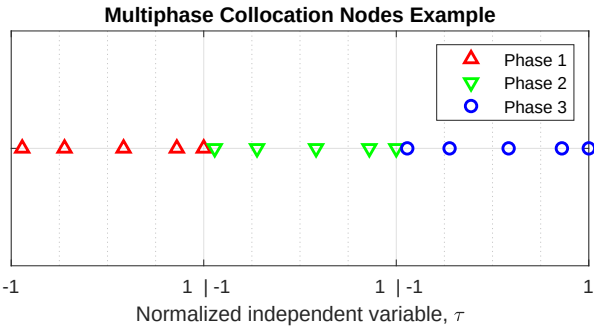

Figure 1: Illustration of the pattern of collocation nodes for multiple phase transcription. Example of three phases, each with five collocation nodes.

2.3.1 Multiphase OCP expressed as a NLP problem

The multiphase nonlinear programming problem can be expressed as minimizing the cost

$$\mathcal{J}^N = \Phi + \sum_{p=1}^{P} \frac{t_f^{(p)} - t_0^{(p)}}{2} \sum_{k}^{N^{(p)}} w_k^{(p)} \Psi_k \,, \tag{12}$$

subject to

$$\boldsymbol{\xi}_k^{(p)} = \frac{t_f^{(p)} - t_0^{(p)}}{2} \boldsymbol{f}_k^{(p)} - \sum_i \boldsymbol{A}_{ki}^{(p)} \boldsymbol{x}_i^{(p)} = \boldsymbol{0} \,, \tag{13}$$

$$\boldsymbol{h}_k^{(p)} \leq \boldsymbol{0} \,, \tag{14}$$

$$\boldsymbol{\phi}^{(p)} \leq \boldsymbol{0} \,, \tag{15}$$

$$\Delta \boldsymbol{x}_{\min}^{(p)} \leq \boldsymbol{\ell}^{(p)} \leq \Delta \boldsymbol{x}_{\max}^{(p)} \,, \tag{16}$$

where the vector $\boldsymbol{\xi}_k^{(p)}$ is a shortened representation of the constraints for dynamic defects from (13).

3 Optimality Conditions

The classical approach (indirect methods) of finding a solution to an optimal control problem begins by deriving the first order optimality conditions. In order to do so, the cost functional (1) is augmented by means of the complementary vectors, $\boldsymbol{\nu}$, $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, to include all constraints expressed in (2) to (4). This process is often referred to as *dualization* [4]. It is important to discuss these conditions in order to verify if a trajectory obtained with the flipped Radau pseudospectral method is optimal. Omitting the phase indicator p for simplicity, one can define the augmented Hamiltonian as [15]:

$$\mathcal{H}\big(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{\mu}\big) = \Psi\big(\boldsymbol{x}, \boldsymbol{u}\big) + \boldsymbol{\lambda}^{\mathsf{T}} \boldsymbol{f}\big(\boldsymbol{x}, \boldsymbol{u}\big) + \boldsymbol{\mu}^{\mathsf{T}} \boldsymbol{h}\big(\boldsymbol{x}, \boldsymbol{u}\big) \,, \tag{17}$$

and the augmented cost functional is written with respect to the Hamiltonian as [9]

$$\mathcal{J}^{\lambda} = \Phi + \boldsymbol{\nu}^{\mathsf{T}} \boldsymbol{\phi} + \int_{t_0}^{t_f} \big(\mathcal{H} - \boldsymbol{\lambda}^{\mathsf{T}} \dot{\boldsymbol{x}}\big) \, \mathrm{d}t \,, \tag{18}$$

where $\boldsymbol{\lambda}(t) \in \mathbb{R}^{n_x}$ is the state covector (or costate), $\boldsymbol{\mu}(t) \in \mathbb{R}^{n_h}$ is the constraint covector and $\boldsymbol{\nu} \in \mathbb{R}^{n_\phi}$ is the endpoint covector.

The first order necessary conditions for optimality can be derived from (18) by setting the first variation of the augmented cost functional equal to zero, $\delta \mathcal{J}^{\lambda} = 0$ [2, 16]. These necessary conditions are expressed in terms of the Hamiltonian as [2, 12, 15, 16]:

$$\boldsymbol{f}\big(\boldsymbol{x}, \boldsymbol{u}\big) - \dot{\boldsymbol{x}} = \boldsymbol{0} \,, \tag{19}$$

$$\boldsymbol{\mu}^{\mathsf{T}} \boldsymbol{h}\big(\boldsymbol{x}, \boldsymbol{u}\big) = 0 \,, \tag{20}$$

$$\boldsymbol{\nu}^{\mathsf{T}} \boldsymbol{\phi}\big(\boldsymbol{x}(t_0), t_0, \boldsymbol{x}(t_f), t_f\big) = 0 \,, \tag{21}$$

$$\frac{\partial \mathcal{H}}{\partial \boldsymbol{u}}\big(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{\mu}\big) = \boldsymbol{0} \,, \tag{22}$$

$$\dot{\boldsymbol{\lambda}}^{\mathsf{T}} + \frac{\partial \mathcal{H}}{\partial \boldsymbol{x}}\big(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{\mu}\big) = \boldsymbol{0} \,, \tag{23}$$

$$\frac{\partial \Phi}{\partial \boldsymbol{x}(t_f)} + \boldsymbol{\nu}^{\mathsf{T}} \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}(t_f)} - \boldsymbol{\lambda}^{\mathsf{T}}(t_f) = \boldsymbol{0} \,, \tag{24}$$

$$\mathcal{H}\big(t_f\big) + \frac{\partial \Phi}{\partial t_f} + \boldsymbol{\nu}^{\mathsf{T}} \frac{\partial \boldsymbol{\phi}}{\partial t_f} = 0 \,, \tag{25}$$

where $\mathcal{H}\big(t_f\big)$ is the Hamiltonian evaluated at t_f, $\mathcal{H}\big(t_f\big) = \mathcal{H}\big(\boldsymbol{x}(t_f), \boldsymbol{u}(t_f), \boldsymbol{\lambda}(t_f)\big)$.

Equations (19), (20) and (21) result from the variation of the augmented cost about the three covectors, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$, respectively, and they are a rewriting of the constraints of the original problem statement in (2) through (4). One interesting result that yields from the optimality conditions is the fact that the inequality constraints (3) and (4) are turned into equality constraints by means of a product with the respective covectors: the conditions are satisfied whenever the inequality constraints are active ($\boldsymbol{h} = \boldsymbol{0}$ and/or $\boldsymbol{\phi} = \boldsymbol{0}$), and they can be satisfied when the constraints are inactive ($\boldsymbol{h} < \boldsymbol{0}$ and/or $\boldsymbol{\phi} < \boldsymbol{0}$) by making sure that the covectors bind to zero at the corresponding times ($\boldsymbol{\mu} = \boldsymbol{0}$ and/or $\boldsymbol{\nu} = \boldsymbol{0}$), thus ensuring a null product in (20) and (21). Because of this, (20) and (21) are often referred to as *slackness conditions*.

Equations (22) and (23) are called the *Euler-Lagrange* equations and they apply at every point along the domain. In particular (23) is called the *costate equation* because it describes the rate of change of the costate, effectively serving as a new equation of motion corresponding to the costates. Equation (22) is also referred to as the strong form of Pontryagin's minimum principle and it only applies if the optimal control sequence lies exclusively within the allowable control set.

Finally (24) and (25) are the endpoint conditions that must be satisfied for the cases of unknown final state and unknown final time, respectively [2].

In addition, because it is not an explicit function of time, the Hamiltonian of the optimal solution will be a constant [2, 9],:

$$\mathcal{H}\big(\boldsymbol{x}(t),\boldsymbol{u}(t),\boldsymbol{\lambda}(t)\big)=\mathcal{H}(t_f)=\mathcal{H}^{\dagger}\,, \tag{26}$$

where $\mathcal{H}^{\dagger}$ represents a generic constant scalar.

The problem of finding a solution that satisfies all first order optimality conditions is known as the *Hamiltonian boundary value problem* (HBVP).

4 NLP Solver Formatting

4.1 Generic input format of the nonlinear solver

In order to solve an optimal control problem via the Radau pseudospectral method, one needs to transcribe the original problem statement into a format which a nonlinear solver can interpret. A nonlinear program is expressed as [8, 17, 18] minimizing

$$J(\boldsymbol{X}_{\text{NLP}})\,, \tag{27}$$

subject to

$$\boldsymbol{C}_{\min}\leq\boldsymbol{C}(\boldsymbol{X}_{\text{NLP}})\leq\boldsymbol{C}_{\max}\,, \tag{28}$$

$$\boldsymbol{X}_{\min}\leq\quad\boldsymbol{X}_{\text{NLP}}\quad\leq\boldsymbol{X}_{\max}\,. \tag{29}$$

where $\boldsymbol{C}_{\max}$ and $\boldsymbol{C}_{\min}$ are, respectively, the upper and lower boundary vectors of the constraints, and $\boldsymbol{X}_{\max}$ and $\boldsymbol{X}_{\min}$ are the upper and lower boundary vectors of the decision variables, respectively. Essentially, there is a cost function, $J(\boldsymbol{X}_{\text{NLP}})$, to be minimized, a list of algebraic constraints, $\boldsymbol{C}(\boldsymbol{X}_{\text{NLP}})$, to be satisfied and a list of parameters, $\boldsymbol{X}_{\text{NLP}}$, that act as decision variables. In order to transform the optimal control problem into a NLP it is necessary to concatenate all the decision variables into a main vector and to aggregate all constraints into a main constraint vector, this is depicted by vectors $\boldsymbol{X}_{\text{NLP}}$ and $\boldsymbol{C}(\boldsymbol{X}_{\text{NLP}})$ respectively. The algebraic constraints contained in $\boldsymbol{C}(\boldsymbol{X}_{\text{NLP}})$ can assert either an equality or an inequality by setting $\boldsymbol{C}_{\max}=\boldsymbol{C}_{\min}$ or $\boldsymbol{C}_{\max}>\boldsymbol{C}_{\min}$ respectively.

4.2 Formatting the vector of decision variables

In order to construct a nonlinear programming problem one needs both, a list of decision variables and a list of algebraic constraints. Let

$$\boldsymbol{XU}^{(p)}=\big[\boldsymbol{x}_0^{(p)\mathsf{T}}\;\boldsymbol{x}_1^{(p)\mathsf{T}}\;\boldsymbol{u}_1^{(p)\mathsf{T}}\;\dots\;\boldsymbol{x}_k^{(p)\mathsf{T}}\;\boldsymbol{u}_k^{(p)\mathsf{T}}\;\dots\;\boldsymbol{x}_{N^{(p)}}^{(p)\;\mathsf{T}}\;\boldsymbol{u}_{N^{(p)}}^{(p)\;\mathsf{T}}\big]^{\mathsf{T}}\,, \tag{30}$$

be the concatenated decision vector of state and control associated with the nodes of a given phase p. Notice that there is no control variable associated with the very first node $i=0$, this is because this node is not collocated [13]. The complete vector of decision variables can be expressed as

$$\boldsymbol{X}_{\text{NLP}}=\big[\boldsymbol{XU}^{(1)\mathsf{T}}\;\dots\;\boldsymbol{XU}^{(p)\mathsf{T}}\;\dots\;\boldsymbol{XU}^{(P)\mathsf{T}}\;t_f^{(1)}\;\dots\;t_f^{(p)}\;\dots\;t_f^{(P)}\big]^{\mathsf{T}}\,. \tag{31}$$

The lower and upper boundaries of the vector of decision variables, $\boldsymbol{X}_{\min}$ and $\boldsymbol{X}_{\max}$ can be build according to an analogous procedure, given the parameters specified by the user for each problem.

4.3 Constraints formatting

The cost functional fed to the nonlinear solver is identical to the cost described in (12), thus,

$$J(\boldsymbol{X}_{\text{NLP}})=\mathcal{J}^{N}\,. \tag{32}$$

In order to construct the concatenated vector, $\boldsymbol{C}(\boldsymbol{X}_{\text{NLP}})$, two additional arrays are introduced, namely $\boldsymbol{F}^{(p)}$ and $\boldsymbol{H}^{(p)}$, containing all the constraints corresponding to phase p from (13) and (14):

$$\boldsymbol{F}^{(p)}=\Big[\boldsymbol{\xi}_1^{(p)\mathsf{T}}\;\boldsymbol{\xi}_2^{(p)\mathsf{T}}\;\dots\;\boldsymbol{\xi}_k^{(p)\mathsf{T}}\;\dots\;\boldsymbol{\xi}_{N^{(p)}}^{(p)\;\mathsf{T}}\Big]^{\mathsf{T}} \tag{33}$$

$$\boldsymbol{H}^{(p)}=\Big[\boldsymbol{h}_1^{(p)\mathsf{T}}\;\boldsymbol{h}_2^{(p)\mathsf{T}}\;\dots\;\boldsymbol{h}_k^{(p)\mathsf{T}}\;\dots\;\boldsymbol{h}_{N^{(p)}}^{(p)\;\mathsf{T}}\Big]^{\mathsf{T}} \tag{34}$$

where $\boldsymbol{\xi}_k^{(p)}$ and $\boldsymbol{h}_k^{(p)}$ are as in (13) and (14), respectively. This way, the concatenated vector of constraints can be written as

$$\boldsymbol{C}(\boldsymbol{X}_{\text{NLP}})=\big[\boldsymbol{F}^{(1)\mathsf{T}}\;\dots\;\boldsymbol{F}^{(P)\mathsf{T}}\;\boldsymbol{H}^{(1)\mathsf{T}}\;\dots\;\boldsymbol{H}^{(P)\mathsf{T}}\;\boldsymbol{\phi}^{(1)\mathsf{T}}\;\dots\;\boldsymbol{\phi}^{(P)\mathsf{T}}\;\boldsymbol{\ell}^{(1)\mathsf{T}}\;\dots\;\boldsymbol{\ell}^{(P-1)\mathsf{T}}\big]^{\mathsf{T}}\,. \tag{35}$$

The lower and upper boundaries for the vector of constraints, $\boldsymbol{C}_{\min}$ and $\boldsymbol{C}_{\max}$, are constructed in an analogous manner, with the peculiarity that the boundaries corresponding to the dynamic defects, must be set invariably equal to zero, $\boldsymbol{F}^{(1:P)}=\boldsymbol{0}$, such that the dynamics are enforced and that the trajectory satisfies the equations of motion at all times.

4.4 Jacobian Matrix

The Jacobian matrix is a quantitative description of the derivatives of the constraints with respect to the decision variables, providing a first order gradient that is necessary to aid the nonlinear solver. This section serves to overview the components of the Jacobian matrix that results from a discretization based on the flipped Radau method. Each row of the Jacobian matrix corresponds to an algebraic constraint, and each column corresponds to a decision variable. Ultimately, each element represents a linear dependency between a constraint (row) with respect to a decision variable (column). The Jacobian can be expressed as [13]

$$\mathbf{Jac} = \nabla_{\boldsymbol{X}_{\mathrm{NLP}}} \begin{bmatrix} J(\boldsymbol{X}_{\mathrm{NLP}}) \\ \boldsymbol{C}(\boldsymbol{X}_{\mathrm{NLP}}) \end{bmatrix} = \begin{bmatrix} \nabla \mathcal{J}^N \\ \nabla \boldsymbol{F}^{(1:P)} \\ \nabla \boldsymbol{H}^{(1:P)} \\ \nabla \boldsymbol{\phi}^{(1:P)} \\ \nabla \boldsymbol{\ell}^{(1:P-1)} \end{bmatrix}. \quad (36)$$

The format of the vectors $\boldsymbol{X}_{\mathrm{NLP}}$ and $\boldsymbol{C}(\boldsymbol{X}_{\mathrm{NLP}})$ dictate the sparsity pattern of the Jacobian matrix. Figure 2 is a visual representation of the pattern of the Jacobian for an example problem with four phases. In this Figure, zero elements are represented by white space, while non-zero elements are represented with coloured dots. The first row of the matrix corresponds to the discrete cost functional where the magenta circles represent a Lagrangian cost and the blue dot at the end of phase four represents a Mayer cost; there are four open terminal times (green columns); a scalar path constraint applied to all phases (concatenated diagonals in cyan); vector event constraints at the terminal times of every phase (magenta blocks); and three linkage conditions connecting the four phases sequentially (red diagonals at the bottom rows). The four phases can be distinguished by the four large red blocks with blue diagonal smaller blocks (corresponding to the dynamic defects of each phase). The prominent red diagonals in the Jacobian correspond to the sparsity pattern of the differentiation matrices and to the identity matrices corresponding to the linkage conditions. The values of the Jacobian matrix corresponding to the red dots are static, i.e., they will remain unchanged in every iteration loop.

4.5 Overview of the Solving Procedure

In order to solve a given optimal control problem a procedural approach is taken. This procedure needs to be generic in order to be able to handle as many problems as possible. The main task at hand is to perform a transformation of the user input into variables and constraints that are useful to feed an NLP solver. The following description applies to the MATLAB tool SPARTAN developed at DLR [10, 13]. Figure 3 presents a high-lever overview of the solving procedure implemented in SPARTAN for visual aid.

5 Rocket Boost-Back and Vertical Landing Problem

The problem to consider is an adaptation from [19]. This example is concerned with the recovery of the first stage of an orbital launcher via vertical landing, where the landing target is located close to the launching site (return-to-launch-site scenario). The vehicle in question is based on the characteristics of the main booster of SpaceX's Falcon 9 rocket. This problem is appropriate to validate the algorithm developed in this work because it is composed of three distinct phases, resulting from the mutation of the equations of motion from one phase to the next. The algorithm is validated by analysis of the optimality conditions.

The problem occurs in a vertical, two-dimensional plane (2-D). At the beginning of the trajectory it is assumed that stage separation has already occurred and that the booster has performed the "turn back" attitude manoeuvre such that the boost-back-burn may begin. The trajectory is composed of three phases:

Phase 1 Boost-back. At $t = t_0$ the rocket is in mid air with ascending linear momentum. Three engines (out of nine) ignite and the thrust direction is constrained to point strictly horizontally. In this phase the rocket inverts its horizontal motion. The duration of the boost-back burn, $t = t_1$, is a known, fixed parameter.

Phase 2 Coast arc. At $t = t_1$ the engines shut down and remain off during all of phase 2. The vehicle takes a parabolic flight profile during this phase. It is taken for granted that an attitude justification manoeuvre takes place during this phase in order to roughly align the thrust vector with the velocity vector at the beginning of phase 3.

Phase 3 Landing. At $t = t_2$ one engine (out of nine) ignites. The vehicle is now controllable and the booster makes its way to the target site. Touchdown happens at $t = t_f$. Both t_2 and t_f are unknown variables to be determined.

The equations of motion are formulated in the target-centred reference frame [19]. Throughout the trajectory, both position and velocity are modelled in Cartesian coordinates, with the reference frame of displacement centred at the landing target location. Figure 4 presents a free-body diagram of the vehicle for visual reference. Downrange is represented by the horizontal axis, x, increasing from left to right, and altitude is represented by the vertical axis, y, increasing from bottom to top. The angle θ indicates the direc-

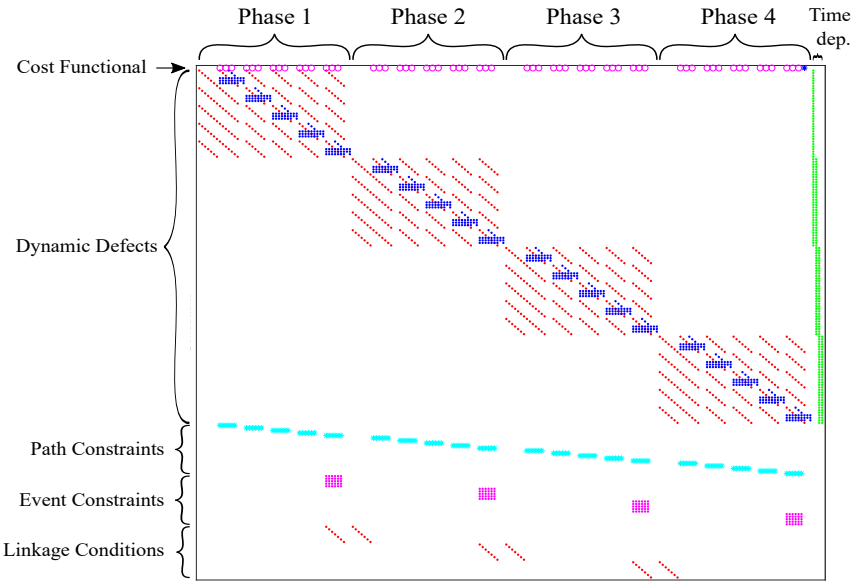

Figure 2: Sparsity pattern of the Jacobian matrix for an example problem with 4 phases, 5 collocation points on each phase. 4 open terminal times. One scalar path constraint applying to every phase and a terminal constraint vector applying to every phase. White spacing represents zero-valued elements.

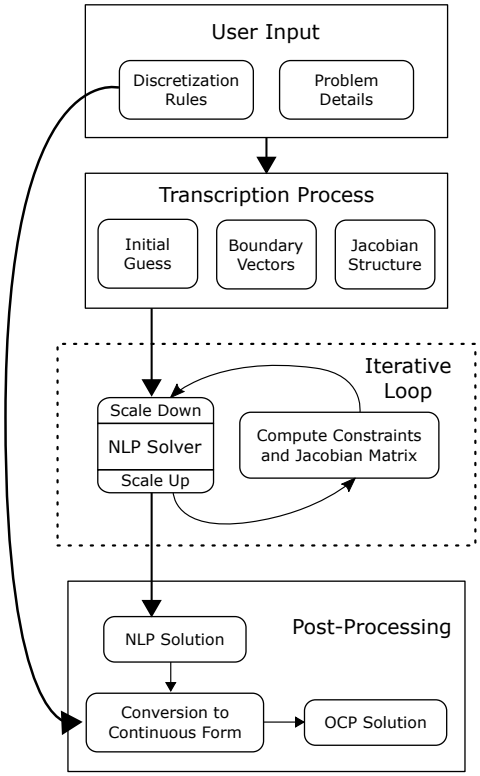

Figure 3: Flowchart of the solving process by the MATLAB tool SPARTAN.

tion of the thrust vector, $\boldsymbol{T}$, and the flight-path angle, represented by the angle γ, indicates the direction of the velocity vector, $\boldsymbol{v}$. Both angles are measured from the x axis and positive in the direction of increasing y. The Cartesian components of velocity, v_x and v_y, are positive in the direction of increasing respective position coordinates. The aerodynamic drag force is represented by vector $\boldsymbol{D}$ which is always collinear with the velocity vector, and m is the mass of the vehicle. The Earth is assumed to be flat and non rotating, and the acceleration of gravity is assumed to be invariant with altitude throughout the trajectory, taking the value g_0. The control variable is the thrust tilt angle, θ. A point-mass approximation of the vehicle is employed, thus the attitude of the spacecraft is not modelled and the angle of attack is assumed to be zero at all times. An exponential model is used for the drag force. The optimal control problem can be formulated

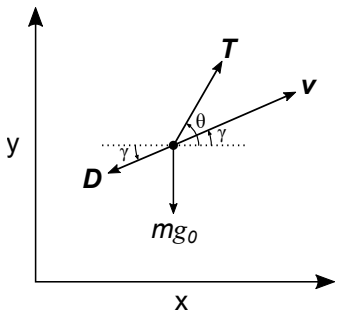

Figure 4: Free-body diagram of the vehicle.

as minimizing the cost

$$\mathcal{J} = \Phi\big(m(t_f)\big) = -m(t_f)\,, \tag{37}$$

associated with the dynamic system

$$\dot{x} = v_x\,, \tag{38}$$

$$\dot{y} = v_y\,, \tag{39}$$

$$\dot{v}_x = \frac{T}{m} k \cos\theta - \frac{D}{m} \cos\gamma \,, \tag{40}$$

$$\dot{v}_y = \frac{T}{m} k \sin\theta - \frac{D}{m} \sin\gamma - g_0 \,, \tag{41}$$

$$\dot{m} = -\frac{T}{I_{sp} g_0} k \tag{42}$$

and subject to the event constraints

$$x(t_0) = 36.022\,\text{km} \,, \tag{43}$$

$$y(t_0) = 60.708\,\text{km} \,, \tag{44}$$

$$v_x(t_0) = 1.052\,\text{km}\,\text{s}^{-1} \,, \tag{45}$$

$$v_y(t_0) = 1.060\,\text{km}\,\text{s}^{-1} \,, \tag{46}$$

$$x(t_f) = 0\,\text{km} \,, \tag{47}$$

$$y(t_f) = 0\,\text{km} \,, \tag{48}$$

$$v_x(t_f) = 0\,\text{km}\,\text{s}^{-1} \,, \tag{49}$$

$$-0.5\,\text{m}\,\text{s}^{-1} \leq v_y(t_f) \leq 0.5\,\text{m}\,\text{s}^{-1} \,, \tag{50}$$

with

$$D = \frac{1}{2}\rho_0 \exp\left\{-\frac{y}{h_0}\right\} C_D \pi \frac{d^2}{4} v^2 \,, \tag{51}$$

$$v^2 = v_x^2 + v_y^2 \,, \tag{52}$$

$$\cos\gamma = \frac{v_x}{v} \,, \tag{53}$$

$$\sin\gamma = \frac{v_y}{v} \,, \tag{54}$$

and

$$T = \begin{cases} \frac{1}{3} T_{LO} & t \in [t_0, t_1] \\ 0 & t \in [t_1, t_2] \\ \frac{1}{9} T_{LO} & t \in [t_2, t_3] \end{cases} \quad k = \begin{cases} 1 & t \in [t_0, t_1] \\ 0 & t \in [t_1, t_2] \\ 1 & t \in [t_2, t_3] \end{cases} \,, \tag{55}$$

where k is the engine throttle, ρ_0 is the air density at sea level, h_0 is the density scale height, C_D is the drag coefficient, d is the diameter of the vehicle and v is the norm of the velocity vector. The linkage conditions are omitted because there are no expected jump discontinuities on the sates of the systems in any phase transition. Table 1 shows the constants and parameters used in this problem. This problem was solved using 10, 8 and 12 collocation nodes in phases 1, 2 and 3, respectively. The NLP solver used was IPOPT [18], and the method used to compute partial derivative was the complex step differentiation method [20]. The results are depicted in Figs. 5 through 10. In addition, the resulting parameters of unknown times and final mass are presented in Table 2.

The trajectory of the vehicle is shown in Fig. 5, where the downrange and the altitude are plotted against each other. The boost-back phase is shown in blue, the coasting arc is shown in orange and the landing phase is represented in yellow. Curiously, the vehicle traces a path that vaguely resembles a shepherd's staff. The plot clearly shows that at the beginning of the trajectory the vehicle is travelling from left to right and with ascending velocity. The boost-back burn inverts the horizontal motion of the vehicle and during the coasting arc, the vehicle takes a parabolic flight due to being in complete free-fall. Finally, in the landing phase the vehicle approaches the zenith of the target and the flight path angle gets closer and closer to being vertical. Intuitively speaking, the trajectory follows a predictable path.

Table 1: Relevant constants and parameters for the Falcon 9 first stage recovery problem.

Constant	Value	Unit
Specific impulse, I_{sp}	282	s
Lift-off Thrust, T_{LO}	5886	kN
Rocket diameter, d	3.66	m
Drag Coefficient, C_D	0.75	1
Density scale height, h_0	7500	m
Gravity acceleration, g_0	9.80665	$\text{m}\,\text{s}^{-2}$
Sea level air density, ρ_0	1.225	$\text{kg}\,\text{m}^{-3}$
Phase 1 initial time, t_0	0	s
Boost-back duration, t_1	40.8	s
Dry mass, m_{dry}	25600	kg

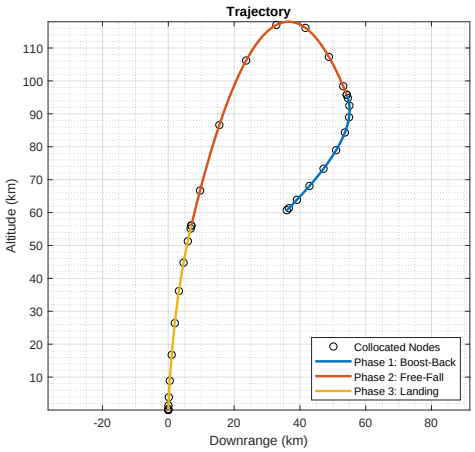

Figure 5: Altitude vs downrange trajectory.

Figure 6 illustrate the Cartesian components of velocity with time. The phase transition points are visible through the increase in density of collocation nodes and also through the removable discontinuities present in each component. The horizontal component of velocity decreases linearly during the first phase and goes from positive to negative, inverting the direction of flight, which is indicative of the boost-back burn phase. This component stays constant during the coasting phase, as expected. It is noticeable that during phases 1 and 2, the vertical component of velocity decreases linearly, without noticeable dis-

continuities, indicating that this component has been subject to a constant acceleration during these two phases, undoubtedly the acceleration of gravity. It is also visible that the vertical component of velocity goes from positive to negative close to the 110 s mark, inverting the direction flight. Both velocity components go to zero during the landing phase, as expected.

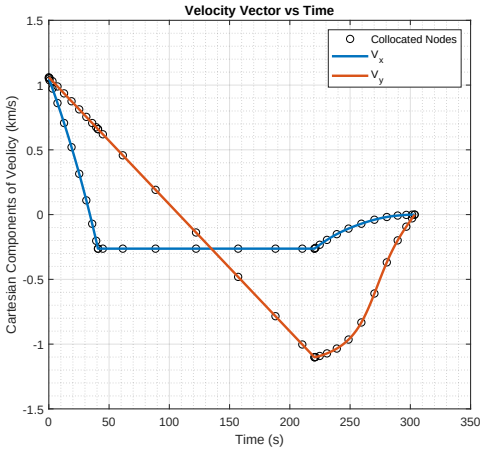

Figure 6: Cartesian components of velocity with time.

The mass profile of the vehicle can be seen in Fig. 7. Due to the constant throttle level, and the constant specific impulses, the mass of the vehicle decays linearly during phases 1 and 3. The mass flow-rate is higher in phase 1 due to the larger thrust associated — there are three engines on during phase 1 and only one engine ignited during phase 3. Not remarkably, the mass stays constant during phase 2 (coast phase).

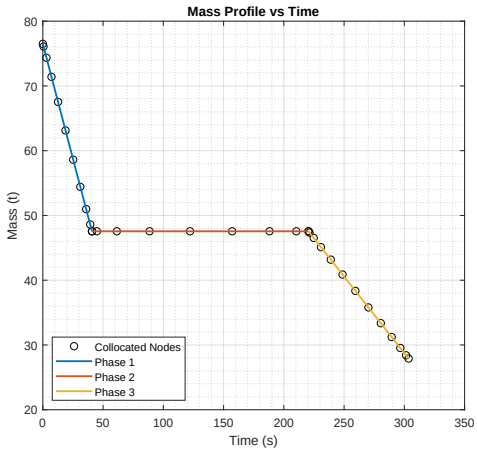

Figure 7: Mass of the vehicle with time.

The angle indicating the thrust direction (control) is presented in Fig. 8. It can be seen that the direction of thrust is constrained to 180° during phase 1, and constrained to 0° during phase 2 (during phase 2 the thrust direction is inconsequential due to the throttle being constrained to zero). Finally, in the landing phase the direction of thrust is allowed to vary, and the angle draws a curve that approaches 90°, indicating that the thrust points vertically at the end of the trajectory.

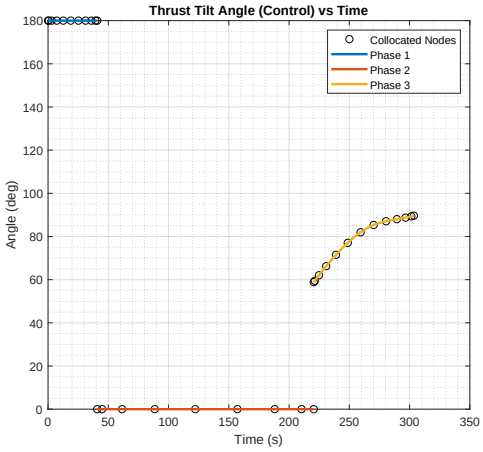

Figure 8: Thrust direction, θ, with time.

Moving on to the solution of dual variables, Fig. 9 shows the evolution of the mass costate along time. Recalling the first order necessary conditions from Section 3, it is possible to note that the endpoint condition expressed in (24) is verified at the final time of the trajectory, namely

$$\lambda_m(t_f) = \frac{\partial \Phi}{\partial m(t_f)} = -1\,. \tag{56}$$

This condition is not verified at any other phase endpoint due to the fact that the mass is either fixed at those points or completely determined by the initial conditions, the constant mass flow rates and the fixed endpoint time of phase 1. Ultimately, the validation of the endpoint condition at the final time of the trajectory brings confidence that the solution is optimal.

Finally, Fig. 10 shows the Hamiltonian associated with the trajectory. By inspection of the plot, one can assert that the Hamiltonian is phase-wise constant, and with regards to phases 2 and 3, one can verify that the Hamiltonian is zero, which implies that the endpoint condition (25) is satisfied in these phases:

$$\mathcal{H}\left(t_f^{(2)}\right) = \mathcal{H}\left(t_f^{(3)}\right) = 0\,, \tag{57}$$

Because the final time of phase 1 is fixed, the endpoint condition (25) does not apply to that phase. Briefly stated, the Hamiltonian satisfies the optimality conditions for time invariant systems [2, 7, 9], and therefore this result brings confirmation that the solution is optimal.

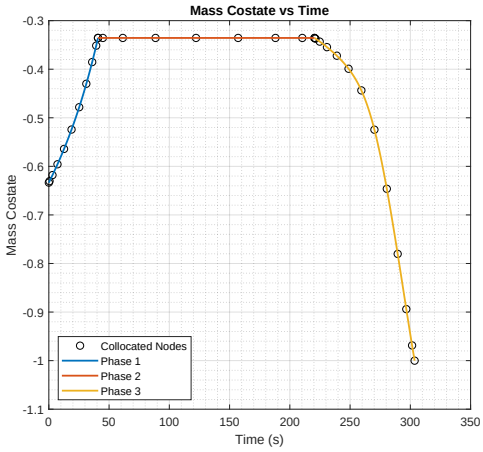

Figure 9: Mass costate with time. The mass costate takes the value -1 at the final time.

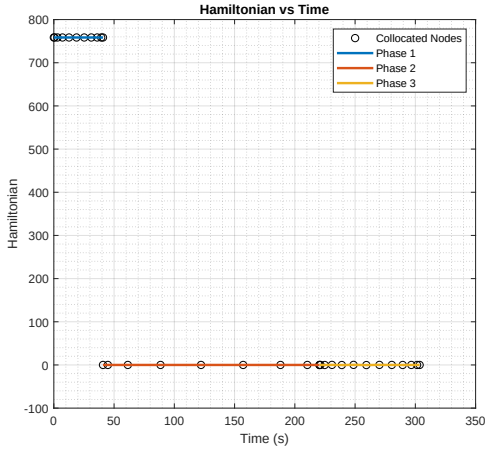

Figure 10: Hamiltonian vs time. Hamiltonian is zero during phases 2 and 3.

Table 2 presents the values of pertinent variables obtained with SPARTAN. The results are overall satisfactory.

6 Conclusions

In this work the flipped Radau pseudospectral method was applied to solve multiphase optimal control problems, specifically multi-stage rocket trajectory generation problems. The algorithm was validated by solving a reference problem containing multiple phases. A simplified version of the problem was implemented, which concerned a booster recovery of the Falcon 9 orbital launcher. Despite the difference in the formulations, the results were satisfactory as the solution was shown to be optimal by analysis of the Hamiltonian.

Table 2: Relevant optimization parameters obtained with SPARTAN.

Parameter	SPARTAN	Unit
Boost-back burn duration, t_1	40.8	s
Landing burn start time, t_2	220.4583	s
Total time of flight, t_f	303.5469	s
Final mass, $m(t_f)$	27905.5554	kg

References

[1] N. X. Vinh. General Theory of Optimal Trajectory for Rocket Flight in a Resisting Medium. *Journal of Optimization Theory and Applications*, 11(2): 189–202, feb 1973. doi: 10.1007/bf00935883.

[2] Arthur E. Bryson Jr. and Yu-Chi Ho. *Applied Optimal Control. Optimization, Estimation, and Control*. Hemisphere Publishing Corporation, Washington New York, 1975. ISBN 9780891162285.

[3] J. T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. SIAM, Philadelphia, second edition, 2010. ISBN 978-0-898716-88-7.

[4] D. A. Benson, G. T. Huntington, T. P. Thorvaldsen, and A. V. Rao. Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method. *Engineering Notes, Journal of Guidance, Control, and Dynamics*, 29(6): 1435–1440, Nov. 2006. doi: 10.2514/1.20478.

[5] F. Fahroo and I. M. Ross. Direct Trajectory Optimization by a Chebyshev Pseudospectral Method. *Journal of Guidance, Control, and Dynamics*, 25(1):160–166, Jan. 2002. ISSN 0731-5090. doi: 10.2514/2.4862.

[6] Q. Gong, W. Kang, N. S. Bedrossian, F. Fahroo, P. Sekhavat, and K. Bollino. Pseudospectral Optimal Control for Military and Industrial Applications. In *2007 46th IEEE Conference on Decision and Control*, pages 4128–4142. IEEE, 2007. doi: 10.1109/cdc.2007.4435052.

[7] A. V. Rao, D. A. Benson, C. L. Darby, M. A. Patterson, C. Francolin, I. Sanders, and G. T. Huntington. Algorithm 902: GPOPS, A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using the Gauss Pseudospectral Method. *ACM Transactions on Mathematical Software*, 37(2):1–39, Apr. 2010. doi: 10.1145/1731022.1731032.

[8] A. Wächter and L. Biegler. On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming. *Mathematical programming*, 106:25–57, March 2005. doi: 10.1007/s10107-004-0559-y.

[9] D. Garg. *Advances in Global Pseudospectral Methods for Optimal Control*. PhD thesis, University of Florida, 2011.

[10] Marco Sagliano and Stephan Theil and Vincenzo D'Onofrio and Michiel Bergsma. SPARTAN: A Novel Pseudospectral Algorithm for Entry, Descent, and Landing Analysis. In *Advances in Aerospace Guidance, Navigation and Control*, pages 669–688. Springer International Publishing, dec 2017. doi: 10.1007/978-3-319-65283-2_36.

[11] P. Davis and P. Rabinowitz. *Methods of Numerical Integration*. Academic Press, Orlando, 1984. ISBN 9780122063602.

[12] Fariba Fahroo and I. Michael Ross. Advances in Pseudospectral Methods for Optimal Control. In *AIAA Guidance, Navigation, and Control Conference and Exhibit, Honolulu, USA, 2008*, pages 1–23, 2008. doi: 10.2514/6.2008-7309.

[13] M. Sagliano, S. Theil, M. Bergsma, V. D'Onofrio, L. Whittle, and G. Viavattene. On the Radau Pseudospectral Method: theoretical and implementation advances. *CEAS Space Journal*, 9 (3):313–331, June 2017. doi: 10.1007/s12567-017-0165-5.

[14] J.-P. Berrut and L. N. Trefethen. Barycentric Lagrange Interpolation. *SIAM Review*, 46(3):501–517, jan 2004. doi: 10.1137/s0036144502417715.

[15] I. M. Ross and F. Fahroo. Legendre Pseudospectral Approximations of Optimal Control Problems. *Springer*, 295:327–342, 2003. doi: 10.1007/978-3-540-45056-6_21.

[16] D. E. Kirk. *Optimal Control Theory, An Introduction*. Dover Publications Inc., 2004. ISBN 0486434842.

[17] P. E. Gill, E. Wong, W. Murray, and M. A. Saunders. User's Guide for SNOPT Version 7.6: Software for Large-Scale Nonlinear Programming. *University of California, San Diego*, January 2017.

[18] Y. Kawajir, C. Laird, S. Vigerske, and A. Wächter. Introduction to IPOPT: A tutorial for downloading, installing, and using IPOPT. *Chicago Northwestern University*, April 2015.

[19] K. S. G. Anglim, Z. Zhang, and Q. Gao. Minimum-Fuel Optimal Trajectory For Reusable First-Stage Rocket Landing Using Particle Swarm Optimization. *World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, 11(5):981–990, 2017. doi: 10.5281/ZENODO.1130268.

[20] V. D'Onofrio. Implementation of Advanced Differentiation Methods for Optimal Trajectory Computation. Master's thesis, Università Degli Studi Di Napoli "Federico II", 2014.