SPARTAN: RAPID TRAJECTORY ANALYSIS VIA PSEUDOSPECTRAL METHODS

Marco Sagliano[1], **David Seelbinder**[1], **and S. Theil**[1]

(1)*German Aerospace Center, Robert Hooke Str. 7, 28359, Bremen, Germany*

1 ABSTRACT

We present SPARTAN: a tool developed by DLR able to transcribe multi-phase optimal-control problems by using pseudospectral methods. The tool analyses the problem provided in continuous form, transforms it into an efficient finite-dimensional representation, which is supplied to state-of-the-art nonlinear programming solvers. The solution is then back-converted into a continuous form and can be analysed to verify that the first-order necessary conditions coming from the application of the Pontryagin Minimum Principle are satisfied. Moreover, an automatic validation of the results is performed by using Dormand-Prince schemes to verify that the optimized states accurately satisfy the equations of the motion. In this paper the main features of SPARTAN are described, and some numerical results are provided to confirm the validity of the implemented approach.

2 INTRODUCTION

The continuous development of transcription methods and the parallel improvement of the Central Processing Units (CPUs) has led to a drastic development of algorithms to numerically attack optimal-control problems (OCPs) [1, 2]. Among these algorithms Pseudospectral Methods gained attention due to their straightforward implementation and the properties they exhibit, like a pseudospectral (i.e., quasi-exponential) convergence of the discrete, finite-dimensional solution to the continuous one when the number of nodes is increased, and the capability to accurately estimate the costates of the underlying continuous problem [3]. This capability is always underestimated, but especially important as it gives the possibility to verify the necessary conditions for optimality through the analysis of the Hamiltonian of the system, providing therefore a theoretical bridge between direct methods, where the states and controls are discretized, and the cost function is optimized by using gradient-based methods, and indirect methods [4], which focus on deriving, and possibly solving the multi-point boundary values problems associated with the application of the Pontryagin's Minimum Principle. It is known that not all the pseudospectral methods exhibit the remarkable properties of exact mapping between the costates of the optimal control problem, and the Lagrange multipliers computed through by the Nonlinear Programming (NLP) solver. SPARTAN (Simple Pseudospectral Algorithm for Rapid Trajectory ANalysis), a tool developed by the German Aerospace Center implements the flipped Radau pseudospectral method, and therefore can exploit this property in virtue of the Covector Mapping Theorem (CMT) [5].

This mapping capability is exploited in SPARTAN to verify a-posteriori that the computed numerical solution satisfies the first-order necessary conditions. Specifically, Secs. 3 and 4 focus on the definition of the continuous optimal control problem and its corresponding transcription by means of pseudospectral methods, respectively. Section 5 shows the results obtained for three different examples emphasizing different aspects of SPARTAN. Finally, some conclusions about this work and SPARTAN are drawn in Sec. 6.

3 OPTIMAL CONTROL PROBLEM

The general optimal control problem that SPARTAN can deal with is the Bolza problem, where the cost function $J(t, \boldsymbol{x}, \boldsymbol{u})$ can include both a terminal cost $\Phi(t, \boldsymbol{x})$ and a running cost, given by the integral of a function $\Psi(t, \boldsymbol{x}(t), \boldsymbol{u}(t))$. Moreover, some problems might be subject to an intrinsic multiphase nature. This is the case for instance of multi-stage ascent systems, which exhibit discontinuities in key-variables like mass and thrust. This type of problem can be cast into SPARTAN by introducing a generic phase $p \in [1, \ldots, P]$, where P is the total number of phases. The problem can therefore be formulated as follows [6].

Minimize the cost functional

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

associated with the trajectory of the multiphase dynamic system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}^{(p)}\big(\boldsymbol{x}(t), \boldsymbol{u}(t), t\big), \qquad p = 1, \ldots, P, \tag{2}$$

subject to the path constraints

$$\boldsymbol{g}_{min}^{(p)} \leq \boldsymbol{g}^{(p)}\big(\boldsymbol{x}(t), \boldsymbol{u}(t), t\big) \leq \boldsymbol{g}_{max}^{(p)}, \qquad p = 1, \ldots, P, \tag{3}$$

the event constraints

$$\boldsymbol{\phi}_{min}^{(p)} \leq \boldsymbol{\phi}^{(p)}\big(\boldsymbol{x}(t_0^{(p)}), t_0^{(p)}, \boldsymbol{x}(t_f^{(p)}), t_f^{(p)}\big) \leq \boldsymbol{\phi}_{max}^{(p)}, \qquad p = 1, \ldots, P, \tag{4}$$

and the phase linkage conditions

$$\Delta\boldsymbol{x}_{min}^{(p)} \;\leq\; \Delta\boldsymbol{x}^{(p)} = \boldsymbol{x}(t_0^{(p+1)}) - \boldsymbol{x}(t_f^{(p)}) \;\leq\; \Delta\boldsymbol{x}_{max}^{(p)}, \qquad p = 1, \ldots, P-1, \tag{5}$$

The domain of the functions expressed in equations Eq. (1) through Eq. (5) are defined accordingly,

$$\begin{aligned}
\Phi \quad &: \mathbb{R}^{n_x} \times \mathbb{R} \quad \times \mathbb{R}^{n_x} \times \mathbb{R} \quad \to \mathbb{R} \\
\Psi \quad &: \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \quad \to \mathbb{R} \\
\boldsymbol{f}^{(p)} \quad &: \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \quad \to \mathbb{R}^{n_x} \\
\boldsymbol{g}^{(p)} \quad &: \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R} \quad \to \mathbb{R}^{n_g^{(p)}} \\
\phi^{(p)} \quad &: \mathbb{R}^{n_x} \times \mathbb{R} \quad \times \mathbb{R}^{n_x} \times \mathbb{R} \quad \to \mathbb{R}^{n_\phi^{(p)}} \\
\Delta\boldsymbol{x}^{(p)} \quad &: \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \quad \to \mathbb{R}^{n_x}
\end{aligned}$$

where n_x is the dimension of the state vector; n_u is the dimension of the control vector; $n_g^{(p)}$ is the number of simultaneous path constraints on phase p; and $n_\phi^{(p)}$ is the number of event constraints at the beginning or end of phase p.

The phases are in this context assumed to be sequential and the interface times from one phase to the next are assumed to be identical $t_0^{(p+1)} = t_f^{(p)},\ p = 1, \ldots, P-1$. This formulation, although seemingly not completely generic, is actually sufficient to account for a vast amount of optimal control problems.

It is relevant to notice that, with the exception of the phase-linkage conditions, the constraints of a multiple phase optimal control problem are always phase-wise decoupled.

4 TRANSCRIPTION BASED ON MULTIPHASE FLIPPED RADAU METHOD

In this section we briefly describe the transcription process according to the flipped Radau pseudospectral method. This process covers all the steps required to move from a so-called *infinite-dimensional* problem, associated with continuous functions, to a *finite-dimensional* representation, that is cast as a nonlinear programming problem (NLP), and can therefore be dealt with one of state-of-the-art NLP solvers available. The transcription involves the continuous domain of the problem, a differential operator to approximate the left-hand side of the equations of motion, as well as an integral operator used to properly represent the running cost. In the following subsections these steps will be described.

4.1 Discretization

The discretization deals with the selection of the finite number of points to represent the optimal control problem. Given a phase discretized at $N+1$ points along the independent variable, a convenient assumption to make is that the continuous solution will be well approximated by an explicit polynomial function of degree N. Then, the smooth polynomial function of degree N which passes through the $N+1$ points can be found via *Lagrange polynomial interpolation*,

$$\boldsymbol{x}(t) \approx \boldsymbol{X}(t) = \sum_{i=0}^{N} \boldsymbol{X}(t_i) L_i(t) \qquad L_i(t) = \prod_{\substack{j=0 \\ j \neq i}}^{N} \frac{t - t_j}{t_i - t_j}$$

where $\boldsymbol{x}(t)$ is the real solution, $\boldsymbol{X}(t)$ is the continuous polynomial approximation, t_i is the independent variable associated with the i^{th} sample point and L_i is the Lagrange interpolating polynomial associated with the i^{th} sample point.

Furthermore, if the solution is well approximated by the N degree polynomial $\boldsymbol{X}(t)$, the first derivative of the solution, with respect to the independent variable can also be estimated based on the properties of Lagrange polynomials, as in [7]:

$$\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} \approx \frac{\mathrm{d}\boldsymbol{X}(t)}{\mathrm{d}t} = \sum_{i=0}^{N} \boldsymbol{X}(t_i) \frac{\mathrm{d}L_i(t)}{\mathrm{d}t} \tag{6}$$

In discrete terms, Eq. (6) is equivalent to the matrix-vector multiplication between a *differentiation matrix* and the column vector of the sampled points $\boldsymbol{X}_i$,

$$\dot{\boldsymbol{X}}_k = \sum_i \boldsymbol{D}_{ki} \boldsymbol{X}_i, \quad i = 0, 1, 2, \ldots, N, \quad k \subseteq i \tag{7}$$

where the elements of the differentiation matrix are defined as:

$$\boldsymbol{D}_{ki} = \dot{L}_i(t_k) = \sum_{\substack{l=0 \\ l \neq i}}^{N} \frac{1}{t_i - t_l} \prod_{\substack{j=0 \\ j \neq i,l}}^{N} \frac{t_k - t_j}{t_i - t_j} \tag{8}$$

The operation expressed in Eq. (7) computes the derivative of $\boldsymbol{X}$ at the points t_k as linear combination of the state $\boldsymbol{X}$ evaluated at the points t_i.

Similarly, assuming once again that the solution is well approximated by a polynomial function, it is possible to compute a definite integral over a specified interval of the independent variable according to a Gaussian quadrature, simply by evaluating the argument function at the discrete points and computing a weighted sum, as in [8]:

$$\int_{-1}^{1} f(t)\,\mathrm{d}t = \sum_{k=1}^{N} w_k f(t_k) \tag{9}$$

The three discrete operations (the Lagrange polynomial interpolation, the derivative operator by differentiation matrix multiplication, and the integration by Gaussian quadrature) above described can be more or less accurate according to a wiser or poorer choice of sample points t_k along the domain of the independent variable, respectively.

4.2 Flipped Radau polynomial roots

Due to the Runge phenomenon, the worst possible choice of sample points for polynomial interpolation is an equidistant grid [8, 9]. In contrast, 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 degree:

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

where $P_N(\tau)$ is the Legendre polynomial of order N.

Figure 1 shows the comparison between an equidistant grid and the distribution of the flipped Radau polynomial roots for reference. One peculiarity of the roots of the Radau polynomial is that they are not symmetric with respect to the origin.

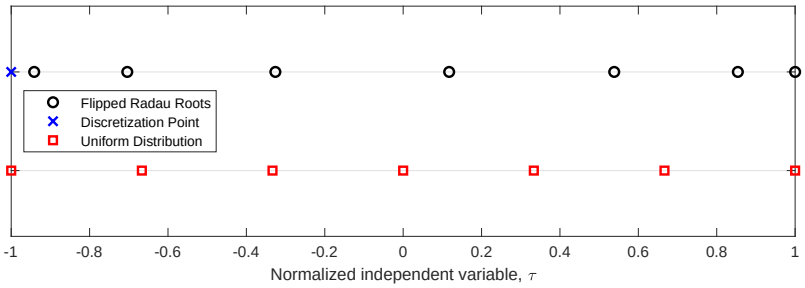

Figure 1: Comparison of node distributions for $N = 7$ in the domain $[-1\ 1]$. A uniform distribution is shown in red squares. The roots of the flipped Radau polynomial are shown in black circles, an extra discretization point at $\tau = -1$ is shown as a blue cross.

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, as in [8, 9]:

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

where τ_i is the i^{th} abscissa of the direct Radau roots; N is the degree of the Radau polynomial; and $P_{N-1}(\tau_i)$ is the Legendre polynomial of degree $N-1$ evaluated at τ_i. The Gaussian quadrature

associated with the flipped Radau method is accurate for polynomial functions of degree up to $2N - 2$.

Because the roots of the flipped Radau polynomial do not include the left limit of the interval ($\tau = -1$) there will be no weight associated with this point, and therefore this point cannot be collocated. However the point at $\tau = -1$ can be used as an auxiliary discretization sample for the calculus of the derivative at the collocation points. Also, this additional discretization point allows the specification of the state at the beginning of the trajectory (initial condition).

Letting $i = 0, 1, 2, \ldots, N$ be the indices corresponding to the discretization points, and $k = 1, 2, \ldots, N$ be the indices corresponding to the collocated points, the resulting matrix $\boldsymbol{D}$ from Eq. (8) will be rectangular of size $N \times (N + 1)$.

Moreover, by mapping the domain of the independent variable, $t \in \left[t_0,\ t_f\right]$, into the normalized domain of $\tau \in \left[-1,\ 1\right]$, as follows [9].

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2} \qquad \tau = \frac{2}{t_f - t_0}t + \frac{t_f + t_0}{t_f - t_0}, \tag{12}$$

The integral operation from Eq. (9) can be generalized for an arbitrary interval of integration:

$$\int_{t_0}^{t_f} f(t)\,\mathrm{d}t = \frac{t_f - t_0}{2}\int_{-1}^{1} f\big(t(\tau)\big)\,\mathrm{d}\tau = \frac{t_N - t_0}{2}\sum_{k=1}^{N} w_k f\big(t_k\big) \tag{13}$$

where $t_k = t(\tau_k)$ and $t_N = t_f$.

The sampling illustrated in Fig. 1 for the Radau roots is applied to each phase individually because each phase is smooth and differentiable, and because the phases are decoupled from one another (derivatives and integral operators act only about the respective phase).

4.3 Constraints in discrete algebraic form

By taking advantage of the discrete operators discussed in section 4.1 and the properties associated with the flipped Radau discretization from Sec. 4.2, a list of algebraic constraints can be formulated that will be an equivalent expression of the initial continuous optimal control problem from section 3, and which can be used as input to the nonlinear solver.

The discrete cost functional is expressed as

$$J^N = \Phi\Big[\boldsymbol{X}_0^{(1)}, t_0^{(1)}, \boldsymbol{X}_N^{(P)}, t_N^{(P)}\Big] + \sum_{p=1}^{P}\frac{t_N^{(p)} - t_0^{(p)}}{2}\sum_{k=1}^{N^{(p)}} w_k^{(p)}\Psi\Big[\boldsymbol{X}_k^{(p)}, \boldsymbol{U}_k^{(p)}, t_k^{(p)}\Big], \quad k = 1, \ldots, N^{(p)} \tag{14}$$

The integral term (Lagrange cost) turns into the sum of weighted evaluations of the argument function Ψ, as expressed in Eq. (13), and the sum across phases is performed. The number of nodes, N, can be specified for each phase individually, and that implies that the quadrature weights might also change from phase to phase.

Next, we compute the differentiation matrix associated with the nodes of the Radau polynomial, Eq. (8), and by performing the operation stated in Eq. (7), equality constraints can be formulated such that the dynamic defects are asserted to be zero, as in [10, 11]:

$$\boldsymbol{F}^{(p)} = \frac{t_N^{(p)} - t_0^{(p)}}{2} f_k^{(p)} \left[\boldsymbol{X}_k^{(p)}, \boldsymbol{U}_k^{(p)}, t_k^{(p)} \right] - \sum_{i=0}^{N^{(p)}} \boldsymbol{D}_{ki}^{(p)} X_i^{(p)} = \boldsymbol{0}, \tag{15}$$

$$i = 0, 1, \ldots, N^{(p)},\ k = 1, \ldots, N^{(p)}$$

where the summation term simply represents the inner product between the discrete vector X_i and the k^{th} row of the differentiaton matrix $\boldsymbol{D}$.

Regarding the path constraints and the event constraints, these are expressed in general terms as inequality constraints, and the only requirement is that they lie within the user-specified boundaries.

$$\boldsymbol{g}_{min}^{(p)} \leq \boldsymbol{g}^{(p)} \left[\boldsymbol{X}_k^{(p)}, \boldsymbol{U}_k^{(p)}, t_k^{(p)} \right] \leq \boldsymbol{g}_{max}^{(p)} \tag{16}$$

$$\boldsymbol{\phi}_{min}^{(p)} \leq \boldsymbol{\phi}^{(p)} \left[\boldsymbol{X}_0^{(p)}, t_0^{(p)}, \boldsymbol{X}_N^{(p)}, t_N^{(p)} \right] \leq \boldsymbol{\phi}_{max}^{(p)} \tag{17}$$

The linkage conditions are the only set of constraints which introduces algebraic coupling of variables between different phases, in practice, these constraints only represent additional boundary conditions that must be satisfied and which relate the states of one phase to the next. Linkage conditions are therefore, a source of discontinuities in the problem, and they do not influence the continuous optimality conditions. The linkage conditions in discrete algebraic form are expressed as:

$$\Delta \boldsymbol{x}_{min}^{(p)} \ \leq \ \Delta \boldsymbol{X}^{(p)} = \boldsymbol{X}_0^{(p+1)} - \boldsymbol{X}_N^{(p)} \ \leq \ \Delta \boldsymbol{x}_{max}^{(p)} \tag{18}$$

4.4 Adjoined vector of decision variables, vector of constraints and Jacobian matrix

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

$$\boldsymbol{XU} = \begin{bmatrix} X_0^\intercal & X_1^\intercal & U_1^\intercal & X_2^\intercal & U_2^\intercal & \ldots & X_N^\intercal & U_N^\intercal \end{bmatrix} \tag{19}$$

be the concatenated state-control decision vector associated with every node of a given phase p. Notice that there is no control vector associated with the very first node $i = 0$, again this is because this node is not collocated. The resulting size of this row vector is $1 \times [n_x + N^{(p)}(n_x + n_u)]$.

The complete vector of decision variables can be expressed as:

$$\boldsymbol{X}_{NLP} = \begin{bmatrix} \boldsymbol{XU}^{(1)} & \boldsymbol{XU}^{(2)} & \ldots & \boldsymbol{XU}^{(P)} & t_N^{(1)} & t_N^{(2)} & \ldots & t_N^{(P)} \end{bmatrix}^\intercal \tag{20}$$

resulting in a column vector of size $[P(n_x + 1) + (n_x + n_u) \sum_{p=1}^{P} N^{(p)}] \times 1$.

Further, taking the algebraic constraints expressed in section 4.3, we may express the list of constraints as follows:

$$\boldsymbol{C}_{NLP} = \begin{bmatrix} J^N & \boldsymbol{F}^{(1)} & \ldots & \boldsymbol{F}^{(P)} & \boldsymbol{g}^{(1)} & \ldots & \boldsymbol{g}^{(P)} & \boldsymbol{\phi}^{(1)} & \ldots & \boldsymbol{\phi}^{(P)} & \Delta \boldsymbol{X}^{(1)} & \ldots & \Delta \boldsymbol{X}^{(P-1)\ T} \end{bmatrix} \tag{21}$$

The vectors $\boldsymbol{X}_{NLP}$ and $\boldsymbol{C}_{NLP}$ describe the nonlinear programming problem at hand. Thus, it is possible to build a Jacobian matrix that will aid the nonlinear solver into finding a solution.

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 valuable for the non linear solver.

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 a constraint (row) with respect to a decision variable (column). Mathematically the Jacobian is expressed as:

$$\textbf{Jac} = \nabla_{\boldsymbol{X}_{NLP}} \boldsymbol{C}_{NLP} = \nabla \boldsymbol{C}_{NLP} = \begin{bmatrix} \nabla J^{N} \\ \nabla \boldsymbol{F}^{(1:P)} \\ \nabla \boldsymbol{g}^{(1:P)} \\ \nabla \boldsymbol{\phi}^{(1:P)} \\ \nabla \Delta \boldsymbol{X}^{(1:P-1)} \end{bmatrix} \tag{22}$$

The format of the $\boldsymbol{X}_{NLP}$ vector from Eq. (20) dictates the sparsity pattern of the Jacobian matrix. Figure 2 illustrates the pattern of the Jacobian for an example problem with four phases: 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).

It is relevant to note that the diagonals on the linkage conditions rows are, invariably, positive and negative identity matrices with the dimension of the state vector, $\pm \boldsymbol{I}_{n_x}$, connecting one phase to the next. Also, it is noticeable in Fig. 2, that the constraints of different phases are decoupled from one another, in other words, decision variables of one phase do not affect the constraints of any other phase, thus the Jacobian matrix is naturally very sparse.

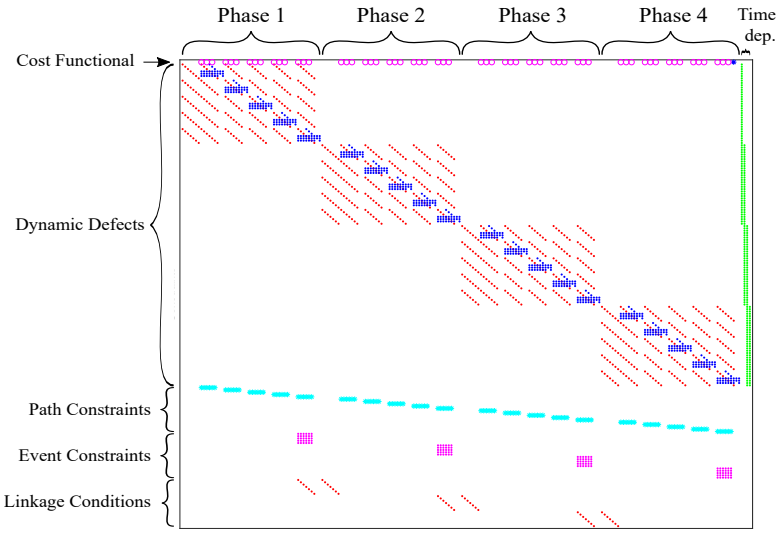

Figure 2: Sparsity pattern of the Jacobian matrix for an example problem with 4 phases, 5 collocation points on each phase. 4 open terminal times. One scalar path constraint applying to every phase and a terminal constraint vector applying to every phase.

In conclusion the NLP to be solved after the transcription process is as follows: we are interested to determine the state vector of Eq. (20) which minimizes Eq. (14), and satisfies the constrants given by Eqs. (15), (16), (17), and (18). In the next section we will see some numerical examples of this approach.

5 NUMERICAL EXAMPLES

In this section three numerical examples are provided to illustrate some features of SPARTAN. Specifically, a minimum time solution for a double integrator, the belly-flop maneuver of the Starship, and a low-thrust orbit-raising trajectory are computed.

5.1 Minimum-Time Optimal Maneuver

The first example shows its multi-phase capability by implementing a minimum-time problem with a system subject to the classical double-integrator dynamics. This example is widely adopted to describe features of optimal-control solutions [4]. The system is modeled as follows.

$$\text{minimize} \quad J = t_F \tag{23}$$

subject to

$$\begin{array}{rcl} \dot{x}_1^p &=& x_2^p \\ \dot{x}_2^p &=& u^p \end{array} \quad p \in [1,2] \tag{24}$$

with boundary conditions

$$\begin{array}{rcl} x_1^1(t_0^1) &=& 1 \\ x_2^1(t_0^1) &=& 2 \\ x_1^2(t_F^2) &=& 0 \\ x_2^2(t_F^2) &=& 0 \end{array} \tag{25}$$

We propose to model the system with two phases, with controls subject to the following *equality constraints*.

$$\begin{array}{rcl} u^1(t) &=& -1 \\ u^2(t) &=& +1 \end{array} \tag{26}$$

This choice is made for demonstration purposes: from optimal control theory we know that for this problem the solution will switch at a given time t^* from -1 to +1. The modeling of Eq. (26) is therefore chosen to highlight the capability of the tool to correctly identify the switching time. The solution is depicted in Fig. 3, while the states and controls are depicted individually in Fig. 4.

We can see that SPARTAN correctly detects both the switching and the final times, equal to 3.7321 and 5.4641 s, respectively. Note that the same solution would be obtained by modeling the problem with a unique phase, but the type of modeling proposed here shows higher accuracy as it intrinsically embeds the discontinuity associated with the true optimal solution of the problem. Moreover, it provides accurate insights about the nature of the optimal solution itself. More advanced examples can be found in [6], showing the Falcon 9 boostback and descent sequence, and in [12], where both the ascent and the lunar landing sequence of the Apollo 11 mission are described.

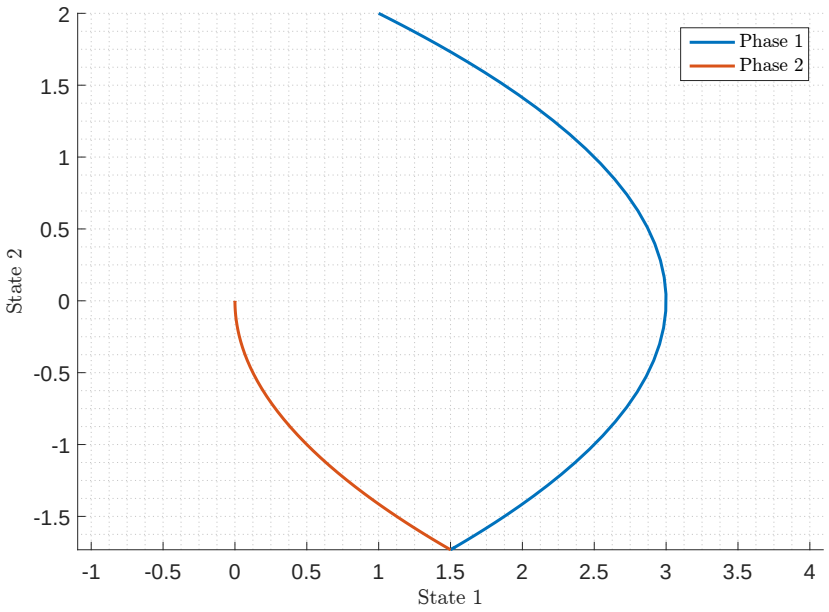

Figure 3: Multi-phase double integrator dynamics solved with SPARTAN: resulting state-space trajectory

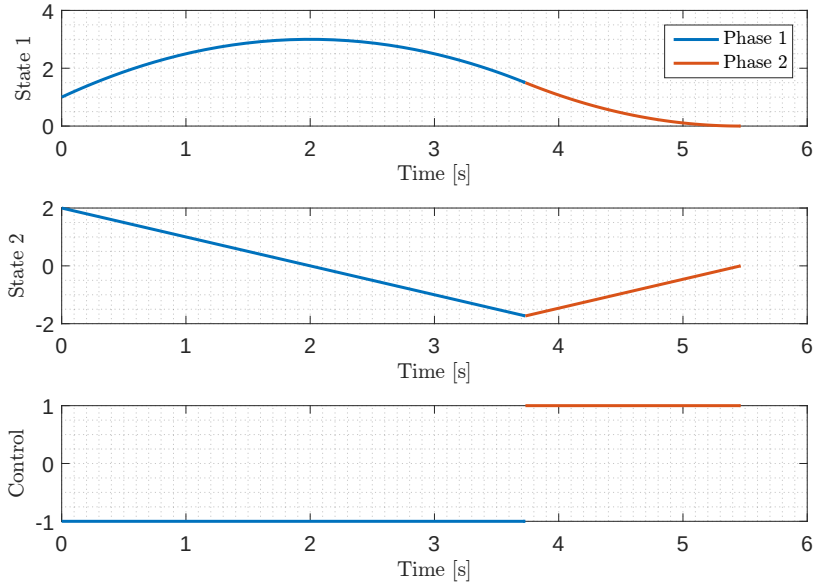

Figure 4: Multi-phase double integrator dynamics solved with SPARTAN: states and controls

5.2 Starship Belly Flop Maneuver

A second example to show the rapid prototyping capabilities of SPARTAN is the implementation of the Starship's *belly flop* maneuver, required to reduce the velocity of the spacecraft through the larger aerodynamic drag caused by exposing a much wider surface. At about 1000 m of altitude the vehicle begins an aggressive maneuver leading its attitude from being horizontal to vertical in order to perform the pinpoint landing maneuver. The motion of the vehicle has been modeled in 2-D, by considering two translational and one rotational degrees of freedom. The body is modeled as a uniform rod having known initial mass m_0 of 100 tons and a length L of 50 m [13]. With no better information it is moreover assumed that the center of mass is at 50% distance from the raptor engines, and that a maximum gimbal angle δ of 20° is allowed. The raptor generates a maximum thrust T_{max} equal to 2210 kN with a throttle capability u defined between 40% and 100%, and has a specific impulse I_{sp} of 380 s. We consider the motion in the plane and therefore the 3-DoF dynamics is captured by the state vector

$$\boldsymbol{x} = \begin{bmatrix} x & y & v_x & v_y & \theta_z & \omega_z & m \end{bmatrix} \tag{27}$$

where x and y are the position components, v_x and v_y the corresponding velocities, θ_z and ω_z the angle and its angular rate around the z axis and m the mass of the starship. We aim at minimizing the mass consumption during the landing maneuver, therefore the cost function is defined simply as follows.

$$\text{minimize} \quad J = m(t_F) \tag{28}$$

For the dynamics we have only one phase, captured by the following differential equations,

$$\begin{aligned} \dot{x} &= v_x \\ \dot{y} &= v_y \\ \dot{v}_x &= -T_{max}\frac{u\sin(\theta+\delta)}{m} \\ \dot{v}_y &= -T_{max}\frac{u\cos(\theta+\delta)}{m} - g \\ \dot{\theta}_z &= \omega_z \\ \dot{\omega}_z &= M_z/I_z \\ \dot{m} &= -T_{max}\frac{u}{I_{sp}g_0} \end{aligned} \tag{29}$$

where the control signals are the throttle level u and the gimbal angle δ, and the torque induced by them is the following expression.

$$M_z = -\frac{L}{2}T_{max}u\sin\delta \tag{30}$$

The boundary conditions to be satisfied are

$$\begin{aligned} x(t_0) &= 0 \ \text{m} & x(t_F) &= 0 \ \text{m} \\ y(t_0) &= 1000 \ \text{m} & y(t_F) &= 0 \ \text{m} \\ v_x(t_0) &= 0 \ \text{m/s} & v_x(t_F) &= 0 \ \text{m/s} \\ v_y(t_0) &= -80 \ \text{m/s} & v_y(t_F) &= 0 \ \text{m/s} \\ \theta(t_0) &= \tfrac{\pi}{2} \ \text{rad} & \theta(t_F) &= 0 \ \text{rad} \\ \omega(t_0) &= 0 \ \text{rad/s} & \omega(t_F) &= 0 \ \text{rad/s} \\ m(t_0) &= 100000 \ \text{kg} & m(t_F) & \quad \text{free} \end{aligned} \tag{31}$$

with final time t_F to be determined by the optimizer.

The resulting trajectory is visible in Fig. 5, while the controls and the corresponding mass profile are depicted in Fig. 6.

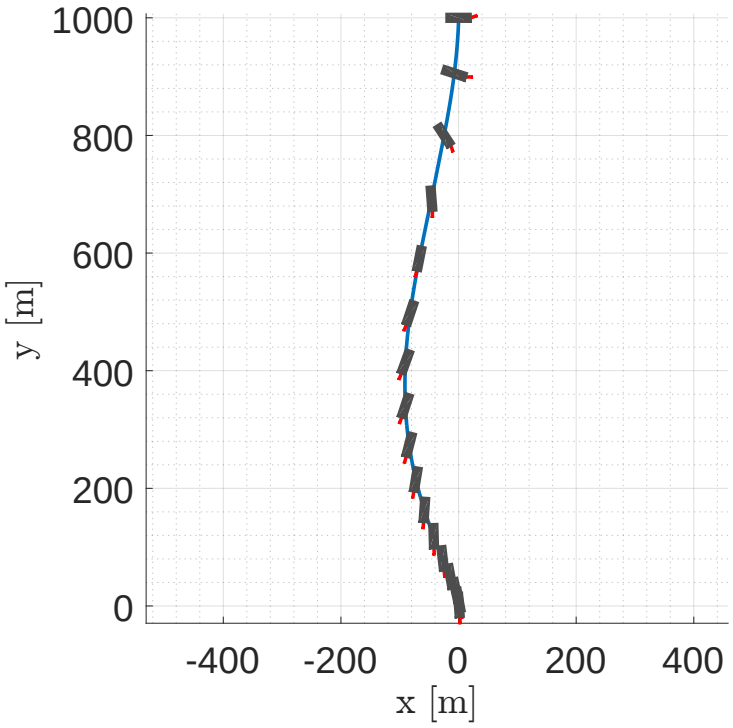

Figure 5: Starship Belly flop maneuver obtained with SPARTAN

A visual comparison with the video of the starship SN10 descent [14] suggests that the profile computed with SPARTAN is indeed a good approximation of the maneuver, confirming the validity of this method to rapidly prototype solutions to complex maneuvers like the starship one. From the analysis of Fig. 6 we can observe that the maneuver is indeed compatible with the operative limits of the Raptor engine and the gimbal angle. With the current assumptions about 7500 kg of fuel are sufficient to perform the maneuver, even though no further comments can be made on the validity of the mass profile without reconstructing the ascent part of the mission as well.

In terms of validation capabilities of the solution the discrepancy between the optimized states computed by SPARTAN and the propagated results are visible in Fig. 7. The maximum error observed in terms of position is in the order of 5 m in terms of position and 1 m/s for what regards the velocity, confirming the high accuracy of the computed open-loop solution.

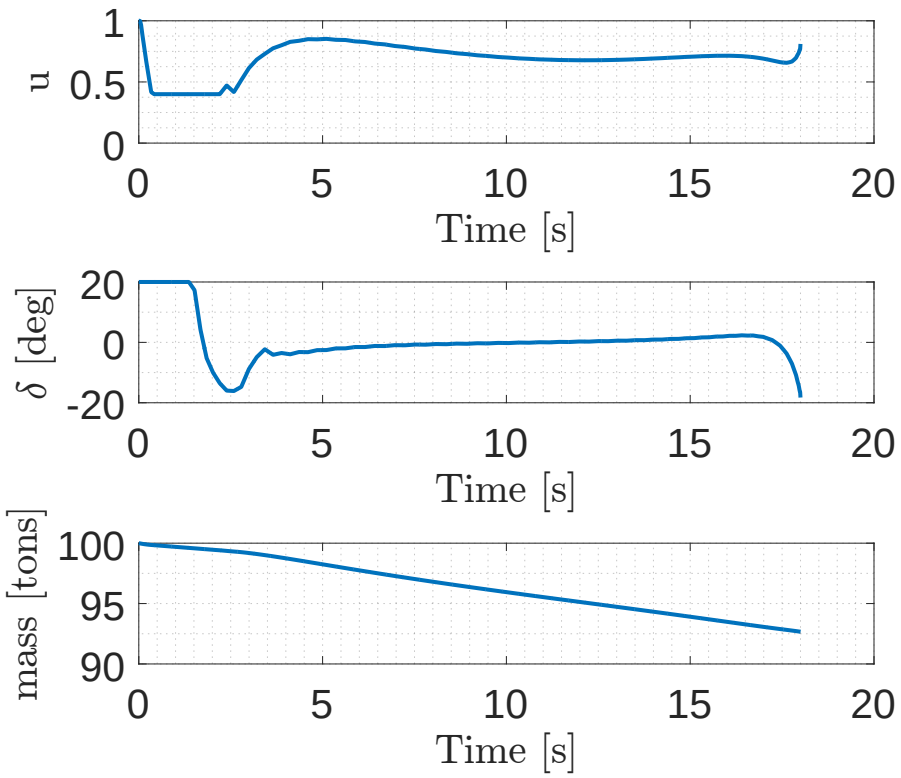

Figure 6: Starship controls and mass profiles obtained with SPARTAN

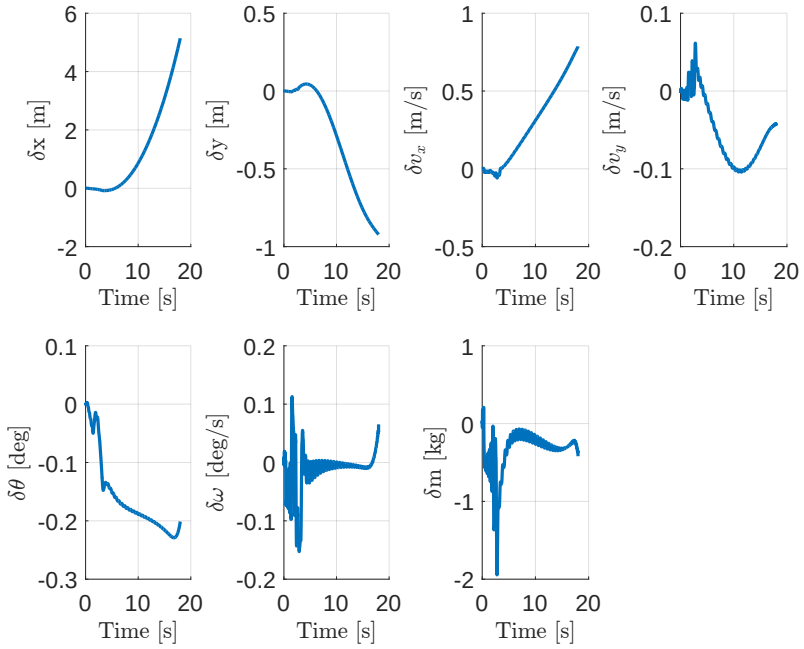

Figure 7: Validation of SPARTAN solutions: discrepancy between Runge-Kutta and SPARTAN states

5.3 Low-Thrust Orbit Transfer

A third representative example is a low-thrust optimal trajectory generation, based on the scenario described by Ross et Al. [15]. We are interested to generate a valid trajectory to raise the orbit of a spacecraft described in polar coordinates by the state vector

$$\boldsymbol{x} = \begin{bmatrix} r & \theta & v_r & v_t \end{bmatrix} \tag{32}$$

where r and θ are the radius and the phase angle of the spacecraft, while v_r and v_t are the radial and tangential components of its velocity. We are interested to perform a minimum-time maneuver, and therefore the cost function is once again Eq. (23), while the one-phase dynamics is given by the following set of nonlinear, coupled equations.

$$\begin{array}{rcl} \dot{r} &=& v_r \\ \dot{\theta} &=& \frac{v_t}{r} \\ \dot{v}_r &=& \frac{v_t^2}{r} - \frac{1}{r^2} + u_r \\ \dot{v}_t &=& -\frac{v_r v_t}{r} + u_t \end{array} \tag{33}$$

The controls are given by the radial and tangential accelerations u_r, u_t, constrained to be within the following squared box.

$$\begin{array}{ll} |u_r| \leq 0.01 & \text{LU/TU}^2 \\ |u_t| \leq 0.01 & \text{LU/TU}^2 \end{array} \tag{34}$$

The boundary conditions to be satisfied are given by

$$\begin{array}{rclllrcll} r(t_0) &=& 1 & \text{LU} & & r(t_F) &=& 4 & \text{LU} \\ \theta(t_0) &=& 0 & \text{rad} & & \theta(t_F) & & & \text{free} \\ v_r(t_0) &=& 0 & \text{LU/TU} & & v_r(t_F) &=& 0 & \text{LU/TU} \\ v_t(t_0) &=& 1 & \text{LU/TU} & & v_t(t_F) &=& 0.5 & \text{LU/TU} \end{array} \tag{35}$$

where LU and TU represent properly defined non-dimensional variables, built on the assumption that the gravitational parameter is equal to 1 LU^3/TU^2 and the initial radius is equal to 1 LU. The resulting trajectory is visible in Fig. 8, and perfectly matches the reference results [15]. Moreover, the automatic validation confirms that the propagated states agree very well with the propagated ones (Fig. 9).

An interesting aspect to observe is that the flipped Radau method relies on the results of the Covector Mapping Theorem, that states that there is a mapping between the optimality conditions of the original continuous OCP and the ones of the corresponding NLP. This mapping is exploited to accurately reconstruct the dual variables of the original problem, which can be exploited to verify that the computed solution is indeed a candidate optimal one. For the example here indeed the application of the Pontryagin Minimum Principle dictates that the switching functions for the two controls u_r and u_t are the dual variables associated with the radial and tangential vectors λ_{v_r} and λ_{v_t}. By applying the CMT SPARTAN computes all the dual variables associated with the problem, and these can be inspected to verify indeed the optimality of the solution from a theoretical perspective. A practical example is giben by Figs. 10 and 11. For Fig. 10 we can observe that the switches in the control signal u_r perfectly match the sign changes of the dual variable λ_{v_r}, confirming the optimality of the solution. In Fig. 11 we can observe that the variable λ_{v_t} is constantly negative, and no switches occur. Consistently, the control u_t is positive all the time, and equal to 0.01 LU/TU^2. This type of information provides further strength to the numerical results obtained by SPARTAN since it combines the straightforward application of a direct method like pseudospectral collocation and the analysis capabilities which are closer to the indirect methods-based approaches.

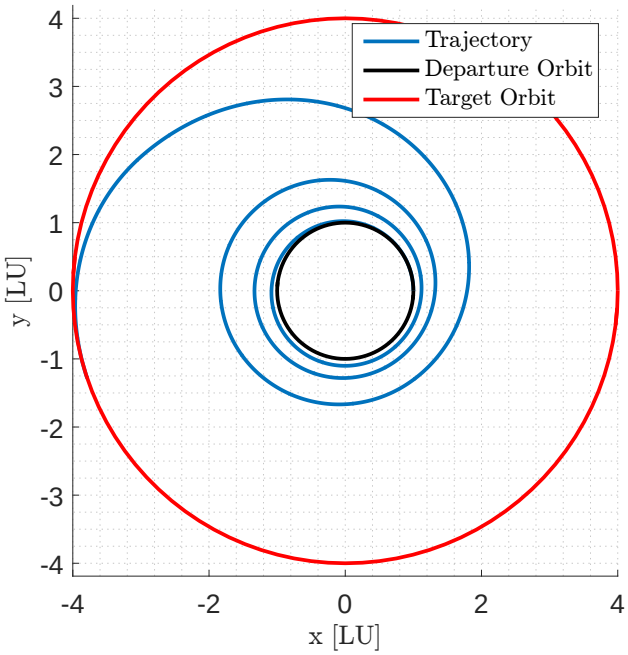

Figure 8: Example of low-thrust solution obtained with SPARTAN - Trajectory

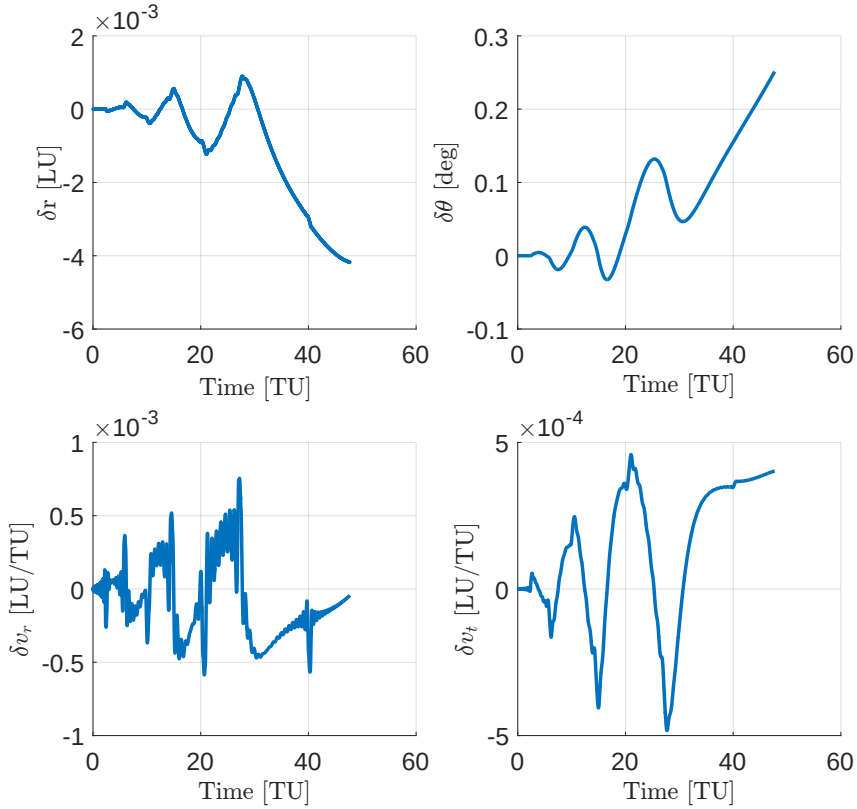

Figure 9: Validation of SPARTAN solutions: discrepancy between Runge-Kutta and SPARTAN states

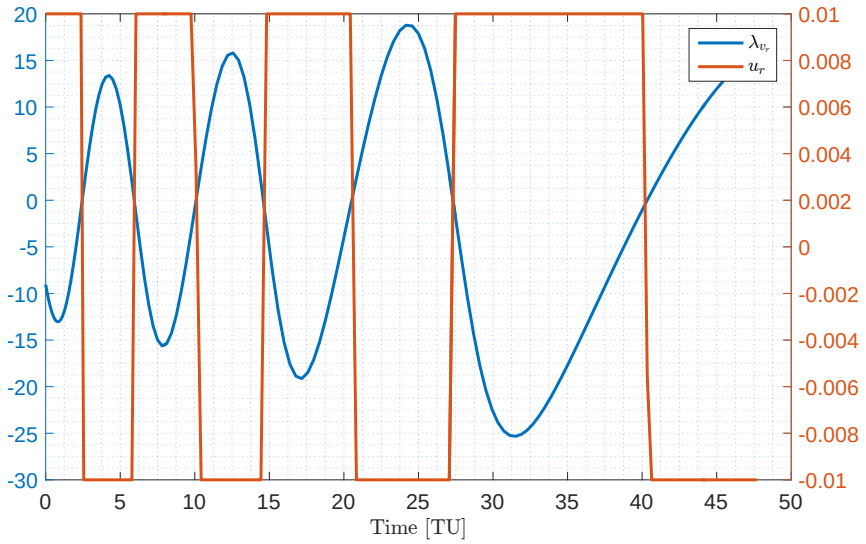

Figure 10: Control structure u_r detected by SPARTAN

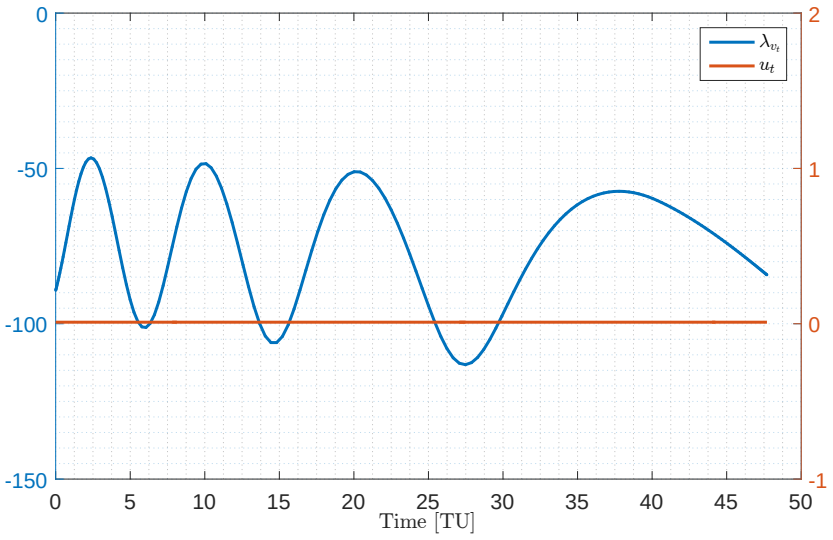

Figure 11: Control structure u_t detected by SPARTAN

6 CONCLUSIONS

This paper highlights some of the features of SPARTAN, a tool developed by DLR to transcribe and solve multiphase optimal control problems through the application of the flipped Radau pseudospectral method. The tool is meant for optimal solution rapid prototyping, as demonstrated by the implementation of the starship's belly flop maneuver, and provides further functionalities to analyse the optimality of the solution through the application of the covector mapping theorem, as demonstrated by the low-thrust example analysed here. Moreover, built-in functionality to numerically check that the original differential equations are satisfied is implemented to further increase the capability of analysing the computed solutions.

SPARTAN will be released as open-source solution to provide a tool able to cope with a large range of space (and potentially non-space related) scenarios.

REFERENCES

[1] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, New York, 1999.

[2] J. T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, 2nd ed.* SIAM, Philadelphia, 2010.

[3] I. M. Ross. *A Primer on Pontryagin's Principle in Optimal Control*. Second. Collegiate Publishers, 2015. ISBN: 0984357114.

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

[5] I. M. Ross. "A Historical Introducion to the Covector Mapping Principle". In: *AAS / AIAA Astrodynamics Specialist Conference, Tahoe, NV, USA*. AAS 05-332. 2005.

[6] José V. Garrido and Marco Sagliano. "Ascent and Descent Guidance of Multistage Rockets via Pseudospectral Methods". In: *AIAA Scitech 2021 Forum*. American Institute of Aeronautics and Astronautics, Jan. 2021. DOI: 10.2514/6.2021-0859.

[7] Jean-Paul Berrut and Lloyd N. Trefethen. "Barycentric Lagrange Interpolation". In: *SIAM Review* 46.3 (Jan. 2004), pp. 501–517. DOI: 10.1137/s0036144502417715.

[8] 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*. 2008, pp. 1–23. DOI: 10.2514/6.2008-7309.

[9] M. Sagliano et al. "On the Radau Pseudospectral Method: theoretical and implementation advances". In: *CEAS SPACE Journal* (2017). Ed. by Springer. DOI: 10.1007/s12567-017-0165-5. Accepted.

[10] I. Michael Ross and Fariba Fahroo. "Legendre Pseudospectral Approximations of Optimal Control Problems". In: *Springer* 295 (2003), pp. 327–342. DOI: 10.1007/978-3-540-45056-6_21.

[11] A. V. Rao et al. "GPOPS: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using the Gauss Pseudospectral Method". In: *ACM Transactions on Mathematical Software* 37.2 (Apr. 2010), pp. 1–39. DOI: 10.1145/1731022.1731032.

[12] Marco Sagliano. “Apollo 11 Reloaded: Optimization-based Trajectory Reconstruction”. In: *AIAA Scitech 2021 Forum*. American Institute of Aeronautics and Astronautics, Jan. 2021. DOI: 10.2514/6.2021-1344.

[13] Thomas Godden. *How SpaceX lands Starship (sort of)*. 2021. URL: https://thomas-godden.medium.com/how-spacex-lands-starship-sort-of-ee96cdde650b.

[14] SpaceX. *SN10 Bellyflop-landing [1080-30 fps]*. 2021. URL: https://www.youtube.com/watch?v=WIoqSHnSn3o.

[15] I. Michael Ross, Qi Gong, and Pooya Sekhavat. “Low-Thrust, High-Accuracy Trajectory Optimization”. In: *Journal of Guidance, Control, and Dynamics* 30.4 (July 2007), pp. 921–933. DOI: 10.2514/1.23181.