Generalized hp Pseudospectral Convex Programming for Powered Descent and Landing

Marco Sagliano*
German Aerospace Center, Bremen, Germany, 28359

I. Introduction

One of the pivotal concepts for a more affordable access to space is reusability. This paradigm shift is led by SpaceX [1], with its successful Falcon 9 program. At the same time, Blue Origin is constantly working on the development of the New Glenn reusable rocket [2]. Governmental agencies are now moving their first steps in the same direction. An example is given by CALLISTO (Cooperative Action Leading to Launcher Innovation in Stage Toss-back Operations), a trilateral project jointly developed by DLR, JAXA, and CNES [3]. The recovery of the first stage includes the capability to react to off-nominal conditions and uncertainties during the entry, descent and landing phases in real-time. This ambitious objective requires the ability to compute valid solutions very rapidly. This objective is the focus of an emerging sub-field of Guidance and Control, known as *Computational Guidance*[4].

Since the beginning of the Apollo era, several methods have been proposed. In the consequently so-called *Apollo guidance* [5], used for the Moon landing, an acceleration profile is computed according to the initial and final (desired) position and velocity. The method solves for the desired terminal conditions, but it is neither optimal in terms of propellant consumption, nor allows the inclusion of further constraints. Another popular approach is the gravity turn [6, 7]. The algorithm is defined by having the thrust direction parallel and opposite to the velocity vector during the powered descent phase. One of the drawbacks is that the correct execution of the algorithm (and therefore the achievement of the desired final conditions) depends on the initial states, and therefore, requires further modifications to be used in different scenarios [8].

A paradigm shift was experienced with the development of convex optimization [9], a class of methods which allow to obtain in real-time optimal solutions for all those problems satisfying some criteria (that is, for all those problems subject to convex constraints). In the field of Entry, Descent, and Landing (EDL) a breakthrough was represented by the development of the lossless convexification for the Mars powered descent problem [10]. The algorithm optimizes the consumption of propellant mass, and allows for the inclusion of multiple constraints, such as glideslope limits. This class of methods has been extended to deal with low-thrust scenarios [11, 12], unmmanned path planning problems [13], rendezvous and proximity operations [14, 15] as well as with entry scenarios with [16] and without thrust control [17, 18]. The real-time capability of convex optimization-based methods has been further boosted by developing highly customized solvers [19], and successful flight results were obtained [20].

A completely different perspective is provided by pseudospectral methods, which represent a different way to

*Research Engineer, GNC Department, AIAA Member transcribe optimal control problems (OCPs), including the powered descent guidance problem [21]. Pseudospectral methods are characterized by the use of non-uniform distributions of discrete nodes, which generally lead to more accurate results if compared with standard transcriptions [22]. The resulting nonlinear programming (NLP) problem is solved with one of the well-known off-the-shelf NLP packages, such as SNOPT [23] or IPOPT [24]. The drawback is represented by the fact that pseudospectral methods cannot in general guarantee to find a solution in polynomial time. Moreover, pseudospectral methods, as any other algorithm based on the use of NLP solvers, compute only local optima, and for complex problems they might require a good initial guess.

Chebyshev polynomials were used for a first step towards the hybridization of pseudospectral methods and convex optimization in the pioneering work of Acikmese et Al. [25]. However, in this work the polynomials were only used for interpolating the controls. This choice implies that neither the properties associated with the use of non-uniform distributions of nodes, nor the dedicated differential and integral operators were exploited. A full pseudospectral-convex hybridization formulation, based on flipped Radau pseudospectral method and Lobatto pseudospectral method has been proposed in our previous work [26]. In this case the accuracy of the solutions, based on a full exploitation of the properties of orthogonal polynomials led to very accurate results, but for larger number of nodes the Central Processing Unit (CPU) time was larger than standard methods. The reason is the structure of the underlying matrices, which are less sparse, leading therefore to larger computation times. In this note we propose a strategy to mitigate this effect by generalizing the previous pseudospectral-convex optimization in the frame of the broader family of hp schemes. This technique is very popular in the Finite-Element method community [27], and was already implemented in other optimization packages [28]. When these methods are employed the matrix representing the transcription of the system dynamics shows a quasi-diagonal structure. Therefore, more efficient numerical routines are used for its manipulation. This increased efficiency translates into a faster computation with limited effects on the accuracy of the solution.

The remainder of this note is organized as follows. Section II briefly reminds the properties of pseudospectral methods, with a focus on the hp flipped Radau pseudospectral method. Section III describes the proposed *hp* pseudospectral method, together with its reduction to the *h* and *p* cases. We show some numerical examples for the powered descent and landing problem, together with the characterization of the accuracy and the computational time of the method for the Mars landing scenario in Sec. IV. Moreover, we assess the effects of the choice of the *h* and *p* terms on the performance of the method. This assessment leads to a simple strategy for the automatic selection of *h* and *p* given the total number of nodes. Finally, Sec. V presents some concluding remarks about this note.

II. Pseudospectral Methods

A. Optimal Control Problem

Let us consider OCPs for autonomous systems in the so-called Bolza form. Given a state vector $\mathbf{x}(t) \in \mathbb{R}^{n_s}$, a control vector $\mathbf{u}(t) \in \mathbb{R}^{n_c}$, the scalar functions $\Phi(\mathbf{x}, \mathbf{u})$ and $\Psi(\mathbf{x}, \mathbf{u})$, the dynamics of the system described by $\mathbf{f}(\mathbf{x}, \mathbf{u}) \in \mathbb{R}^{n_s}$, and the vector $\mathbf{g}(\mathbf{x},\mathbf{u}) \in \mathbb{R}^{n_g}$ we can formulate the problem as follows:

$$\text{minimize} \quad J = \Phi\left[\mathbf{x}\left(t_f\right), \mathbf{u}\left(t_f\right)\right] + \int_{t_0}^{t_f} \Psi\left[\mathbf{x}(t), \mathbf{u}(t)\right] dt \tag{1}$$

subject to

$$\begin{gathered}
\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}\right) \\
\mathbf{g}_L \leq \mathbf{g}\left(\mathbf{x}, \mathbf{u}\right) \leq \mathbf{g}_U \\
\mathbf{x}_L \leq \mathbf{x}(t) \leq \mathbf{x}_U, \mathbf{u}_L \leq \mathbf{u}(t) \leq \mathbf{u}_U \\
\mathbf{x}_0 = \mathbf{x}(t_0), \quad \mathbf{x}_f = \mathbf{x}(t_f)
\end{gathered} \tag{2}$$

We can see the presence of the endpoint function, represented by the Mayer term, and an integral form, that is, the *Lagrange* term. We deal with problems with bounded states and controls, and there might be path constraints, represented by the function $\mathbf{g}\left(t, \mathbf{x}, \mathbf{u}\right)$, which can further limit the solution space. Moreover, initial and final conditions might be constrained as well. Equations (1)-(2) represent a generic continuous optimal control problem. This representation is useful to describe a wide variety of problems, and several researchers are constantly working on the convexification of sub-categories of this type of problems. Some relevant examples in that sense can be found in [13, 29, 30].

B. Pseudospectral Methods

Optimal control problems can be solved by using indirect methods and direct methods. In the first case the Pontryagin Minimum Principle is used, and the problem is transformed into an equivalent multiple-point boundary-value problem. Direct methods are based on the discretization (or *transcription*) of the OCP, which generates a finite-dimensional NLP problem. Pseudospectral methods represent a particular area of interest in the frame of the wider class of direct methods. Examples of tools implementing pseudospectral methods include DIDO [31], GPOPS [32], and SPARTAN [21]. Pseudospectral methods benefit from the following properties:

- "Spectral" (i.e., quasi-exponential) convergence of the NLP solution to the OCP solution when the number of nodes employed is increased (and the problem is smooth)
- Runge phenomenon is avoided

While in our previous work [26] we focused on the pure form of the flipped Radau pseudospectral method, in this Note we develop the hp flipped Radau Pseudospectral method (or hp-fRPm). The benefits of this choice will be clear in the next sections, while further details on the flipped Radau pseudospectral method can be found in [21].

C. Hp Pseudospectral Methods

Pseudospectral methods use discrete, non-uniform domain coming from the use of Legendre-related polynomials. The original time domain of our problem $[t_0,\ t_f]$ is mapped against the so-called *pseudospectral time* $[-1,\ 1]$. To compute a more accurate solution the number of nodes collocated in this interval is increased, which implies that we are increasing the degree of the polynomials used to approximate the continuous variables of the original problem, for instance, by using p nodes. Since p is the only parameter we are controlling, we can state that the global flipped Radau pseudospectral method is a *p-method*. However, one can think to break the time domain in sub-domains, and to *locally* collocate the equations of motion in each segment. In this case there will be two parameters to define, which are the number of segments h, and the number of nodes per segment p. The definition of these two parameters is the idea behind the so-called *hp methods*, which are very popular in finite-element methods [27] as well as in computational fluid dynamics [33]. Hp methods were successfully applied to optimal control, by using adaptive schemes, which allowed for automatic mesh refinements aimed at improving the accuracy of the solutions [32, 34]. Note that p can be a vector $p \in \mathbb{R}^h$, where each element $p_1, p_2, \ldots, p_h$ identifies the degree of the polynomial used for the segment $1, 2, \ldots, h$. However, throughout this paper since no automatic mesh-refinement is involved, we will use a constant value p for all the segments.

A double notation is now introduced to differentiate the segments and the nodes within a segment. Subscripts i refer to the i^{th} node in a given segment, while superscripts j define the j^{th} segment. Therefore,

$$\mathbf{X}_i^j \quad \mathbf{U}_i^j \quad \mathbf{G}_i^j, \quad \begin{matrix} j \in [1, \ldots, n] \\ \\ i \in [1, \ldots, p] \end{matrix} \tag{3}$$

identify the generic state, control and constraint at the i^{th} node of the j^{th} segment. Accordingly, we will have n time segments, defined as

$$[t_0^j,\ t_f^j], \quad j = [1, \ldots, n] \tag{4}$$

and the generic element of the overall time vector is t_i^j. With this notation we can transcribe Eqs. (1)-(2) according to the hp flipped Radau pseudospectral method:
Minimize (or maximize) the cost function J, for $i = 1, \ldots, p$, and $j = 1, \ldots, n$.

$$J = \Phi\left[\mathbf{X}_p^n, \mathbf{U}_p^n\right] + \frac{t_p^n - t_0^1}{2n} \sum_{j=1}^{n} \sum_{i=1}^{p} w_i \Psi\left[\mathbf{X}_i^j, \mathbf{U}_i^j\right] \tag{5}$$

subject to

$$\begin{gathered} \mathbf{F}_i^j = \mathbf{D}^j \cdot \begin{bmatrix} \mathbf{X}_0^j & \mathbf{X}_1^j & \ldots & \mathbf{X}_p^j \end{bmatrix} - \tfrac{t_p^n - t_0^1}{2n} \mathbf{f}\left(\mathbf{X}_i^j, \mathbf{U}_i^j\right) = \mathbf{0} \\ \mathbf{g}_L \leq \mathbf{G}\left(\mathbf{X}_i^j, \mathbf{U}_i^j\right) \leq \mathbf{g}_U \\ \mathbf{x}_L \leq \mathbf{X}_i^j \leq \mathbf{x}_U \\ \mathbf{u}_L \leq \mathbf{U}_i^j \leq \mathbf{u}_U \end{gathered} \tag{6}$$

where $\mathbf{D}^j$ is a differentiation matrix needed to transform the differential equations of the OCP into a set of algebraic constraints [21, 22]. Note that since we chose to have the same number of nodes p in each segment, the matrix $\mathbf{D}$ is the same for all the segments; therefore $\mathbf{D}^j = \mathbf{D}$ holds, and the index j can be dropped. In addition to Eqs. (5)-(6), the so-called *linking conditions* are required, that is

$$\begin{aligned} t_p^{j-1} &= t_0^j \\ \mathbf{X}_p^{j-1} &= \mathbf{X}_0^j \end{aligned} \quad , \quad j = [2, \ldots, n] \tag{7}$$

Eqs. (5)-(7) describe the hp pseudospectral algorithm, which will be combined with convex optimization, leading to the proposed generalized hp pseudospectral convex method.

III. Hp Pseudospectral Convex Optimization

In this section we present the hp pseudospectral convex framework that is potentially able to generate real-time capable optimal solutions for the Mars powered descent and landing phase, as described in [10, 35, 36]. Note that also in this context we neglect the aerodynamic forces generated during the Mars descent.

A. Mars Powered Descent and Landing

In 2012 NASA's rover Curiosity successfully landed on the martian surface [37]. During the final descent and landing phases the retrorockets were used to counteract Martian gravity and ensure the proper conditions for a soft touchdown. An elegant formulation of this problem was proposed by Acikmese and Ploen [10]. Specifically, the corresponding OCP can be stated in the following convex form. We are interested in minimizing the fuel mass required for the landing, formulated through the use of a slack variable σ:

$$\text{minimize} \quad J = \int_{t_0}^{t_f} \sigma(t) dt \tag{8}$$

The dynamics is described by the following set of equations:

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \mathbf{u} + \mathbf{g} \\ \dot{z} &= -\alpha\sigma \end{aligned} \tag{9}$$

with the constraint $\|\mathbf{u}(t)\|_2 \leq \sigma(t)$. The minimization of the integral of σ is equivalent to minimize the fuel consumption, which is the original cost function in the nonlinear formulation described in [10]. $\mathbf{r} \in \mathbb{R}^3$ is the position vector, and $\mathbf{v} \in \mathbb{R}^3$ represents the velocity vector, both expressed in a surface-fixed reference frame, depicted in Fig. 1. Note that the variable z in the last equation is the natural logarithm of the mass, defined consistently with previous works to overcome the nonlinearity caused by mass consumption [10, 26, 35]. The lander is equipped with $n = 6$ thrusters, having a cant angle $\phi = 27$ degrees and able to provide a thrust $\mathbf{T} \in \mathbb{R}^3$. For each component we can define

$$T_i = T_{max} n \widehat{T}_i \cos\phi, \ i = x, y, z \tag{10}$$

with T_{max} equal to 3.1 kN. Note that $\widehat{T}$ obeys the following constraint:

$$\widehat{T}_l \leq \left\| \widehat{T} \right\| \leq \widehat{T}_u \tag{11}$$

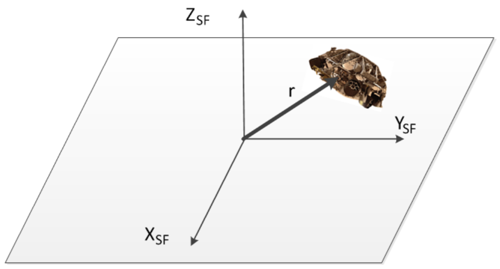


Fig. 1 Surface-fixed reference frame.

where $\widehat{T}_l = 0.3$ and $\widehat{T}_u = 0.8$, and the link between $\mathbf{T}$ and $\mathbf{u}$ is given by

$$\mathbf{u} = \frac{\mathbf{T}}{m} \tag{12}$$

Equation (12) and the definition of z imply that Eq. (11) can be expressed in terms of σ and z as

$$\rho_l e^{-z(t)} \leq \sigma(t) \leq \rho_u e^{-z(t)} \tag{13}$$

with the terms ρ_l and ρ_u equal to the minimum and the maximum values of $\|\mathbf{T}\|$. This expression is approximated by using second-order Taylor expansion and first-order Taylor expansion for the lower and the upper boundaries, respectively.

$$\rho_l e^{-z_l} \left[1 - (z - z_l) + \frac{1}{2}(z - z_l)^2\right] \leq \sigma(t) \leq \rho_u e^{-z_u} \left[1 - (z - z_u)\right] \tag{14}$$

Finally, the centers of expansion z_l and z_u are computed according to

$$\begin{aligned} z_l &= log(m_0 - \alpha \rho_l t), \\ z_u &= log(m_0 - \alpha \rho_u t), \end{aligned} \tag{15}$$

The Martian gravity vector is defined as $\mathbf{g} = [0,\ 0,\ -3.7114]$ m/s^2. The initial mass m of the lander is equal to 1905 kg, and its value is updated by using $\alpha = \frac{1}{I_{sp} g_e \cos\phi}$, where $I_{sp} = 225$ s is the specific impulse of the thrusters, and $g_e = 9.807$ m/s^2 is the Earth's gravitational constant. The time of flight t_f is assigned and equal to 78 s. This assumption is necessary to express the dynamics in linear form. Note that free final time can be included by adopting a linesearch algorithm, by exploiting the fact that the optimal mass consumption is unimodal with respect to the time of flight, and by consecutively solving convex subproblems [10]. The inclusion of this feature in the current class of proposed algorithm is kept as future research topic. Initial and final positions and velocities are listed in Table 1. Safer trajectories can be generated by including the glideslope constraint:

Table 1 Boundary conditions for the Mars descent and landing scenario.

Parameter	Value	Units
$r(t_0)$	[2000 0 1500]	m
$r(t_f)$	[0 0 0]	m
$v(t_0)$	[100 0 −75]	m/s
$v(t_f)$	[0 0 0]	m/s

$$\left[\frac{r_z(t)}{\sqrt{r_x^2(t)+r_y^2(t)}}\right] \geq \tan\widetilde{\theta}_{alt}, \quad \widetilde{\theta}_{alt} = 4 \text{ deg} \tag{16}$$

This constraint ensures that during its descent the lander moves within a cone having a semi-angle equal to $90 - \widetilde{\theta}_{alt}$ degrees, and therefore does not decrease its altitude without a corresponding reduction of the lateral position error while reaching the target landing point.

B. States and Controls Discretization

The first step for the application of the proposed method to the powered landing problem is the determination of the discrete timesteps, and the state vector representing the solution. For each of the n segments we can compute the p roots of the flipped Radau-Legendre polynomials as defined in [21]. Note that in each of the n segments we have $p + 1$ discrete nodes. However, the first point is not collocated. Moreover, we have $n - 1$ link conditions defined in Eq. (7). Therefore, we want to determine the solution only in np points, which correspond to the discrete set of pseudospectral times $\tau_i \in [-1,\ 1]$. These nodes are converted into physical time-steps by using the first of the affine transformations defined in [21], adapted to the j^{th} segment,

$$t_i^j = t_f^{j-1} + \frac{t_f^j - t_0^j}{2}\tau_i + \frac{t_f^j + t_0^j}{2}, \; i = 0,\ldots,p, \; j = 1,\ldots,n, \quad t_f^0 \triangleq 0 \tag{17}$$

The discrete time vector is of course non-uniform in virtue of the nature of the hp pseudospectral transcription. For the states and the controls we propose to use the following vector:

$$\mathbf{X} = \begin{bmatrix} \mathbf{r}_1^1 & \mathbf{v}_1^1 & z_1^1 & \mathbf{u}_1^1 & \sigma_1^1 & \ldots & \mathbf{r}_p^n & \mathbf{v}_p^n & z_p^n & \mathbf{u}_p^n & \sigma_p^n \end{bmatrix}^T \tag{18}$$

Note that the initial conditions ($\mathbf{r}_0$, $\mathbf{v}_0$, and $\mathbf{z}_0$) and the initial controls ($\mathbf{u}_0$ and σ_0) are excluded from the definition of $\mathbf{X}$, consistently with the fact that the initial node of the fRPm is *not collocated*.

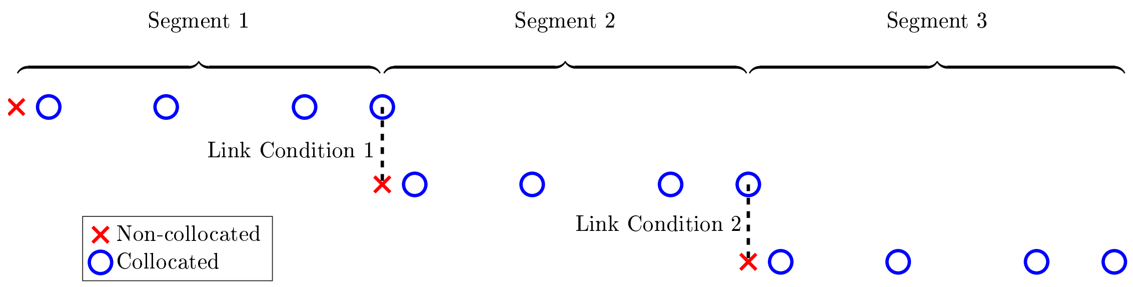


Fig. 2 Linking conditions between segments in the hp collocation scheme.

C. Cost Function

The vector **c** representing the cost function will be a vector having dimensions $np(n_s + n_c) \times 1$, in which only np elements, corresponding to the σ_i^j values, are different from zero. Therefore we have

$$c_k = \begin{cases} \dfrac{t_f^n - t_0^1}{2n} w_i, & \begin{array}{l} i = 1, \dots, p \\ j = 1, \dots, n \\ k = ij(n_s + n_c) \end{array} \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

where w_i are the Radau quadrature weight defined in [21], and t_0^1 and t_f^n are the initial and final times of the entire mission profile, assumed to be known. Note that for each of the segments the first node is non-collocated. However, the same node is the last collocated node of the previous segment, as depicted in Fig. 2. Therefore, its weight will simply correspond to w_p.

D. Dynamics

If we define the continuous state vector as

$$\mathbf{x_c} = [\mathbf{r} \quad \mathbf{v} \quad z]^T \tag{20}$$

and the control as

$$\mathbf{u_c} = [\mathbf{u} \quad \sigma]^T \tag{21}$$

the dynamics of Eq. (9) has the following state-space representation,

$$\mathbf{A_c} = \begin{bmatrix} O_{3\times3} & I_3 & 0 \\ O_{3\times3} & O_{3\times3} & O_{3\times1} \\ O_{1\times3} & O_{1\times3} & 0 \end{bmatrix}, \quad \mathbf{B_c} = \begin{bmatrix} O_{3\times3} & 0 \\ O_{3\times3} & 0 \\ O_{1\times3} & -\alpha \end{bmatrix} \tag{22}$$

where $O_{n_1 \times n_2}$ and I_{n_3} are the zero matrix of dimensions n_1 and n_2 and the identity matrix of dimensions n_3, respectively. In the standard transcription the matrices $\mathbf{A_c}$ and $\mathbf{B_c}$ were converted into their discrete counterparts $\mathbf{A_d}$ and $\mathbf{B_d}$, whereas with the hp pseudospectral convex framework we can skip this transformation, and directly use $\mathbf{A_c}$ and $\mathbf{B_c}$.

In our case we build the residuals of the differential equations as

$$\dot{\mathbf{x}}_\mathbf{c}(t) = \mathbf{A_c x_c}(t) + \mathbf{B_c u_c}(t) + \mathbf{B_c g} \tag{23}$$

since $\dot{\mathbf{x}}_{\mathbf{c},i} \cong \mathbf{D}_{i,\dots} \cdot \mathbf{x_c}$ we can write that for each segment j and node i

$$\mathbf{D}_{i,\dots} \cdot \mathbf{x_c}^j(t) - k_t \mathbf{A_c x_c}^j(t) - k_t \mathbf{B_c u_c}_i^j(t) = k_t \mathbf{B_c g} \tag{24}$$

holds. This relationship, evaluated in the p nodes, leads to the following definitions. In each segment ($j = 1, \dots, n$) we have

$$\mathbf{A^j_{dyn}} = \begin{bmatrix} D_{1,1}I_{n_s} - k_t A_c & -k_t B_c & \dots & \dots & D_{1,p}I_{n_s} & O_{n_s \times n_c} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ D_{p,1}I_{n_s} & O_{n_s \times n_c} & \dots & \dots & D_{p,p}I_{n_s} - k_t A_c & -k_t B_c \end{bmatrix}, \quad \mathbf{b^j_{dyn}} = \begin{bmatrix} -D_{1,0}x_p^{j-1} + k_t B_c g \\ \vdots \\ \vdots \\ -D_{p,0}x_p^{j-1} + k_t B_c g \end{bmatrix} \tag{25}$$

The term k_t is computed as $(t_f^n - t_0^1)/2n$, according to the definition of Eq. (5). Note that the knowledge of the initial conditions is exploited to construct the vector $\mathbf{b^1_{dyn}}$ through the first column of the matrix $\mathbf{D}$, representing the discrete, non-collocated point corresponding to x_0, while the linking conditions are implicitly guaranteed by including the elements x_p^{j-1} in the definition of the vectors $\mathbf{b^j_{dyn}}$.

E. Final Conditions

Arbitrary final conditions can be met by imposing further terms as linear constraints. Supposing that all the six components on position and velocity are constrained to some values $\mathbf{r_f}$, $\mathbf{v_f}$, we can impose them by defining a further matrix $\mathbf{A_{fc}}$, and a further vector $\mathbf{b_{fc}}$ as

$$\begin{aligned} \mathbf{A_{fc}} &= \begin{bmatrix} O_{(n_s-1)\times n_s} & O_{(n_s-1)\times n_c} & \dots & \dots & I_{(n_s-1)} & O_{(n_s-1)\times n_c+1} \end{bmatrix} \\ \mathbf{b_{fc}} &= \begin{bmatrix} \mathbf{r_f} & \mathbf{v_f} \end{bmatrix}^T \end{aligned} \tag{26}$$

Remark 1 Note that the number of rows of $\mathbf{A_{fc}}$ and elements of $\mathbf{b_{fc}}$ are in this case equal to $n_s - 1$ because the final mass is not constrained.

The linear system representing the dynamics and the final conditions is therefore given by the following condition

$$\mathbf{AX = b} \tag{27}$$

where

$$\begin{aligned} \mathbf{A} &= \begin{bmatrix} \mathbf{A^1_{dyn}}^T & \dots & \mathbf{A^n_{dyn}}^T & \mathbf{A_{fc}}^T \end{bmatrix}^T \\ \mathbf{b} &= \begin{bmatrix} \mathbf{b^1_{dyn}}^T & \dots & \mathbf{b^n_{dyn}}^T & \mathbf{b_{fc}}^T \end{bmatrix}^T \end{aligned} \tag{28}$$

F. Constraints

Finally, constraints can be included as specified in Sec.VI.A of our previous paper, and are here omitted for brevity. The entire problem is expressed as

$$\text{minimize} \quad J = \mathbf{c}^T \mathbf{X} \tag{29}$$

subject to Eqs. (27), (16), (14) in discrete form, together with the conic constraint $\|\mathbf{u}\|_2 \leq \sigma$. The initial state is known, and the initial control is extrapolated from the control history once that the problem is solved. The computation of the initial control completes the solution with all the missing information.

G. Reduction to *h*- and *p*-method

In this subsection we will show how the *hp* framework reduces to the *h*- and to the *p*-methods when the number of collocated nodes or segments are reduced to 1, respectively.

Reduction to *p*-method

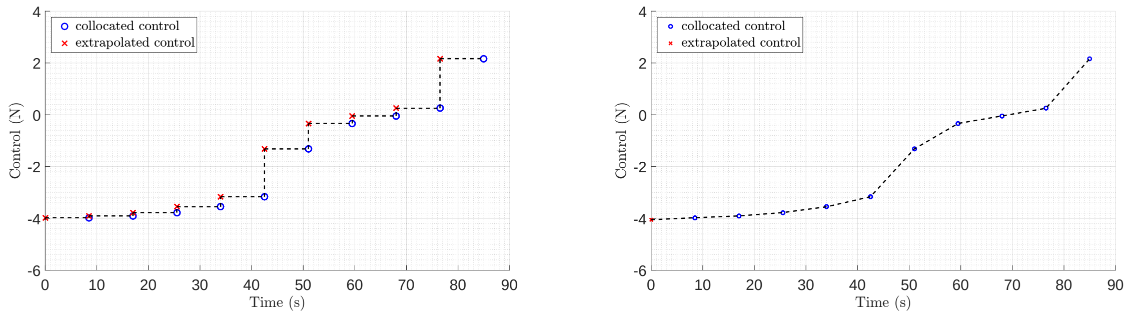
The reduction to the *p*-method is described in detail in [21]. By setting $n = 1$ the cost function of Eq. (19), as well as all the other constraints defined in Eq. (27) become equal to the ones of [26].

Reduction to *h*-method

This case is more interesting. We will have several segments, and for each segment we set $p = 1$. With this definition there is a unique root of the associated flipped Legendre Radau polynomial, that is $\tau = 1$, while the non-collocated node is of course $\tau = -1$. The differentiation matrix becomes

$$D = \begin{bmatrix} -1/2 & 1/2 \end{bmatrix} \tag{30}$$

while the Radau-Gauss quadrature weight (associated with the node $\tau = 1$) is equal to 2. With these definitions we can observe that the discrete differentiation matrix becomes a standard finite-difference central scheme. As previously mentioned, the initial control is extrapolated in the first segment, and provided by the linking conditions for the other segments. Interestingly, for this specific case this means that only the value at the end of each segment is computed. If we interpret each node linking two segments as the first, non-collocated node of the following segment, the *h*-scheme translates into a constant control signal in each segment, that is, a zero-order hold approximation, which is exactly the scheme proposed by Acikmese in [10]. Alternatively, one can interpret the linking points as the last collocated node of each segment. In this case the control history can be represented as first-order hold, which is the choice adopted for the implemented method. The two interpretations are represented in Figs. 3(a) and 3(b). For what regards the cost function, since we are considering n segments, each with 1 collocated node, the unique weight w_1 of the node at $\tau = 1$ is equal to



(a) Zero-order hold (b) First-order hold

Fig. 3 *h*-method - interpretation of the control law as zero-order hold or first-order hold according to the definition of the linking conditions.

2. The flipped Gauss Radau quadrature rule becomes

$$\int_{t_0}^{t_f} f(t)dt \cong \frac{t_f - t_0}{2n} \sum_{j=1}^{n} \sum_{i=1}^{p} w_i f(t_i^j) = \Delta t \sum_{j=1}^{n} f(t_1^j) \tag{31}$$

where Δt is simply equal to $\frac{t_f - t_0}{n}$, that is exactly the choice adopted for the discretization of the cost function in [10]. Therefore, the flipped *hp* pseudospectral framework in its *h* form reduces to a transcription based on equi-spaced nodes for the discretization, central differences for the approximation of the derivatives, and a simple mid-point rule for the integral operator. Therefore the proposed framework can be interpreted as a *morphing transcription*, which gives us a degree of freedom to balance the accuracy of the solution (driven by increasing the nodes in each segment), and the CPU time, which can be controlled by choosing the values of p and h.

IV. Numerical Examples

In this section we describe some numerical examples for the scenario described in Sec. III.A. First we compare the results coming from the nominal scenario with the ones obtained by using a nonlinear programming approach, and solved by using SPARTAN, an in-house algorithm developed by the German Aerospace Center [21]. All the cases have been run with Matlab 2015b on a laptop having an i7-368 processor with a clock frequency of 2.6 GHz. To assess the reliability of the method, we performed a Monte-Carlo campaign with 1000 runs. Finally, an analysis of the accuracy and the CPU time of the solution when the parameters h and p are varied is carried on. This analysis will suggest a criterion for an optimal choice of the h and p terms given the total number of nodes.

A. Nominal case: comparison with NLP-based approach

In this first case we computed the solution for $n = 7$, $p = 7$ (which leads to a total of 50 discrete nodes, including the initial non-collocated node). The solution is depicted in Figs. 4-6, and compared with the one obtained by SPARTAN.

We can see from Fig. 4, (showing position, velocity, acceleration and controls) that the solution is fully consistent

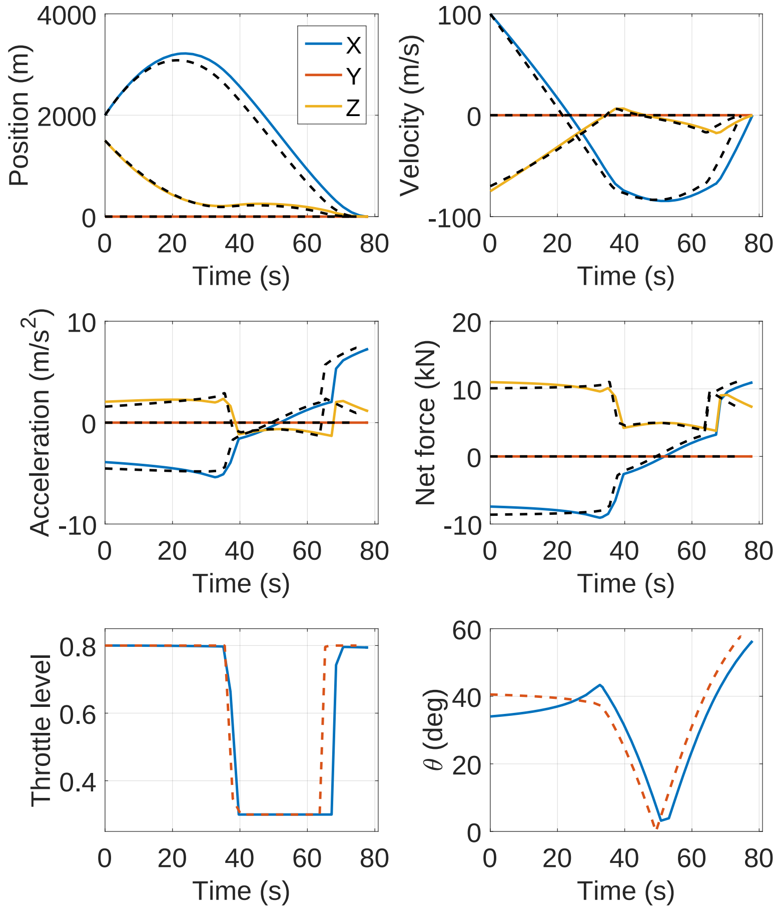


Fig. 4 Solution obtained with hp pseudospectral convex method: states and controls - The dotted lines represent the NLP-based solution obtained with SPARTAN.

with the results obtained in the original formulation [10]. The glideslope constraint is also fully satisfied (Fig. 5(a)). The final mass consumption is equal to 399.5 kg, and the mass profile is depicted in Fig. 5(b). Figure 6 shows the control history with respect to time. The original non-convex control constraints, together with the equality condition $\Gamma = \|\mathbf{T}\|$, with $\Gamma = m\sigma$ (prescribed by the application of the Pontryagin's minimum principle to this specific problem) are satisfied. The main difference with respect to the results obtained with SPARTAN resides in the fact that SPARTAN is able to optimize the time of flight as well, leading to a $t_f = 74.6$ s, and to a slightly different profile for position and velocity. The difference in terms of mass consumption is about 14 kg. This difference is reduced to 10.2 kg when a fixed time equal to 78 s is assigned in SPARTAN as well. A deeper analysis showed that the remaining difference is a direct consequence of the Taylor series of Eq. (15) used to approximate the upper and lower boundaries on thrust described by Eq. (13), while in SPARTAN Eq. (11) is directly employed. Nevertheless, the shape of the solution is very

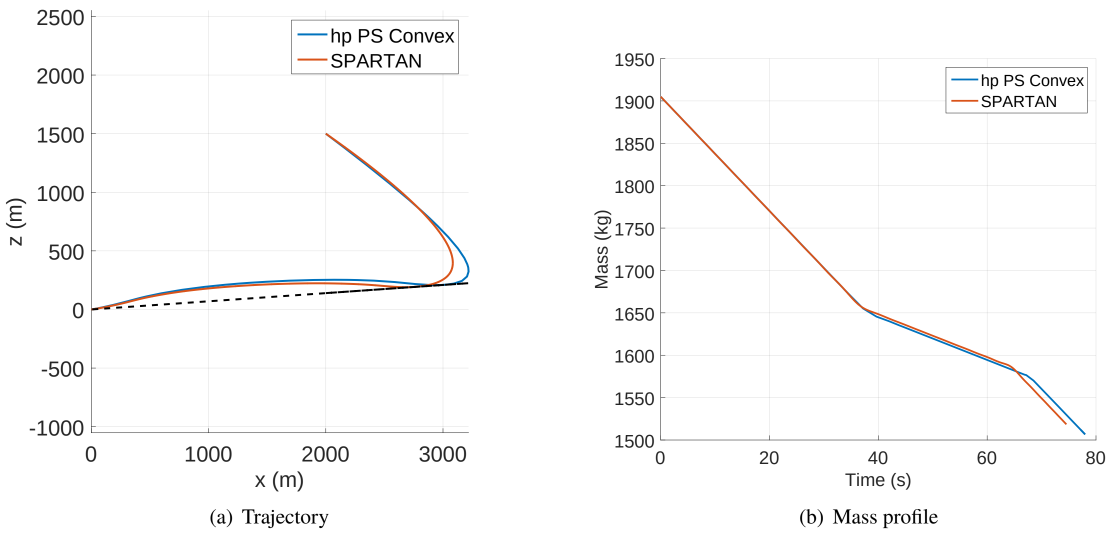


Fig. 5 Solution obtained with hp pseudospectral convex method: trajectory and mass profile.

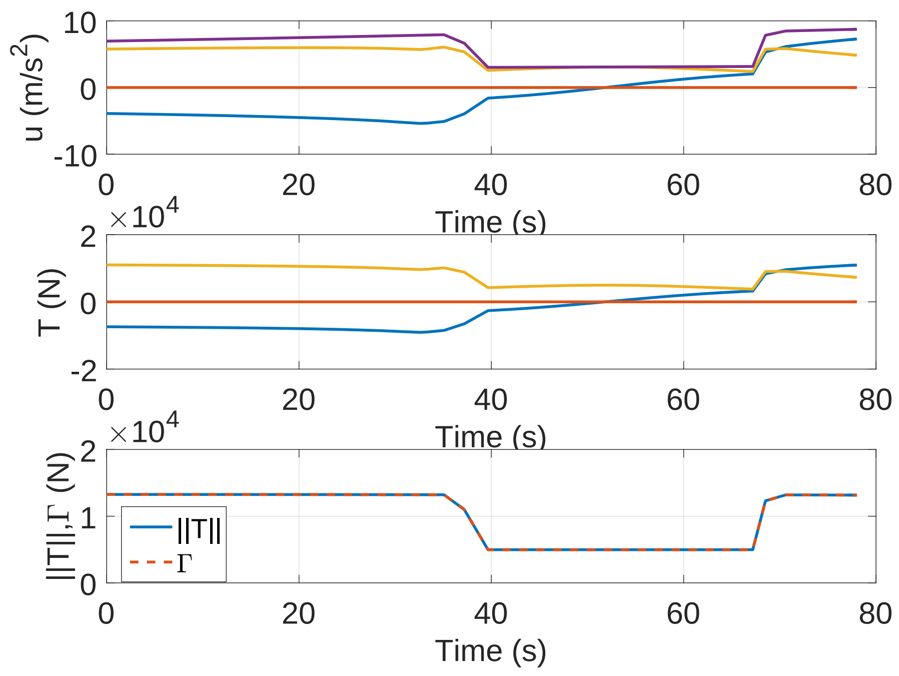


Fig. 6 Solution obtained with hp pseudospectral convex method - control components.

similar, and confirms the validity of the proposed hp pseudospectral convex approach as potentially real-time capable alternative to nonlinear programming-based techniques.

Table 2 Monte-Carlo analysis - uncertainties.

Parameter	Dispersion	Distribution	Units
$\|\|\mathbf{r}(t_0)\|\|$	±25	Uniform	m
$\|\|\mathbf{v}(t_0)\|\|$	±10	Uniform	m/s
$m(t_0)$	±1	Uniform	kg
t_f	±3	Uniform	s

B. Monte-Carlo Campaign

In this section we show the results associated with a Monte-Carlo campaign (1000 cases) to test the reliability of the method. Perturbations in terms of initial position, velocity, mass, and final time are included. The ranges and the type of uncertainty for each of the variables included in the analysis are listed in Table 2. States and controls are depicted in Fig. 7. The resulting perturbed trajectories are shown in Fig. 8(a), while the mass consumption profiles are plotted in Fig. 8(b).

Remark 2 Note that in the Figs. 7-8 only the first 25 cases are shown for a better visualization of the results.

From the analysis of Figs. 7 we can observe that all the solutions satisfy the constraints as in the nominal case. Despite the relatively large variations in terms of initial velocities, the scheme is able to steer the lander towards the prescribed point in every case analyzed. The envelope of corresponding trajectories is depicted in Fig. 8(a), where not only two-dimensional cases, but also full three-dimensional cases are considered. Since the final time is not optimal, and significant variations of initial position and velocity are introduced, there is a large variation of total mass consumption. In fact, the optimal value of mass spent during the landing varies between 388 and 420 kg (Fig. 8(b)).

C. Accuracy Comparison

Finally, a more systematic analysis of the trade-off between accuracy and CPU time can be observed in Table 3. Results are compared in terms of landing errors due to the accuracy of the different transcriptions, and in terms of CPU time required to compute a solution. Since we are interested to measure the effectiveness of the different transcriptions when we change the parameters h and p the landing errors are computed as the difference between the optimal trajectory and the propagated solution that we compute by propagating the optimal controls. This approach allows to directly observe the landing error due to the transcription only without taking other effects into account. The CPU times are the time required by ECOS [38] to solve the corresponding SOCP problem. For a better characterization of the CPU times, every case has been computed 10 times, and the CPU times averaged. Four campaigns of runs have been considered, each of them with a total number of nodes fixed, and equal to 25, 50, 75 and 100, respectively. For each of these campaigns the number of segments and nodes are varied to assess the effects on the CPU time as well.

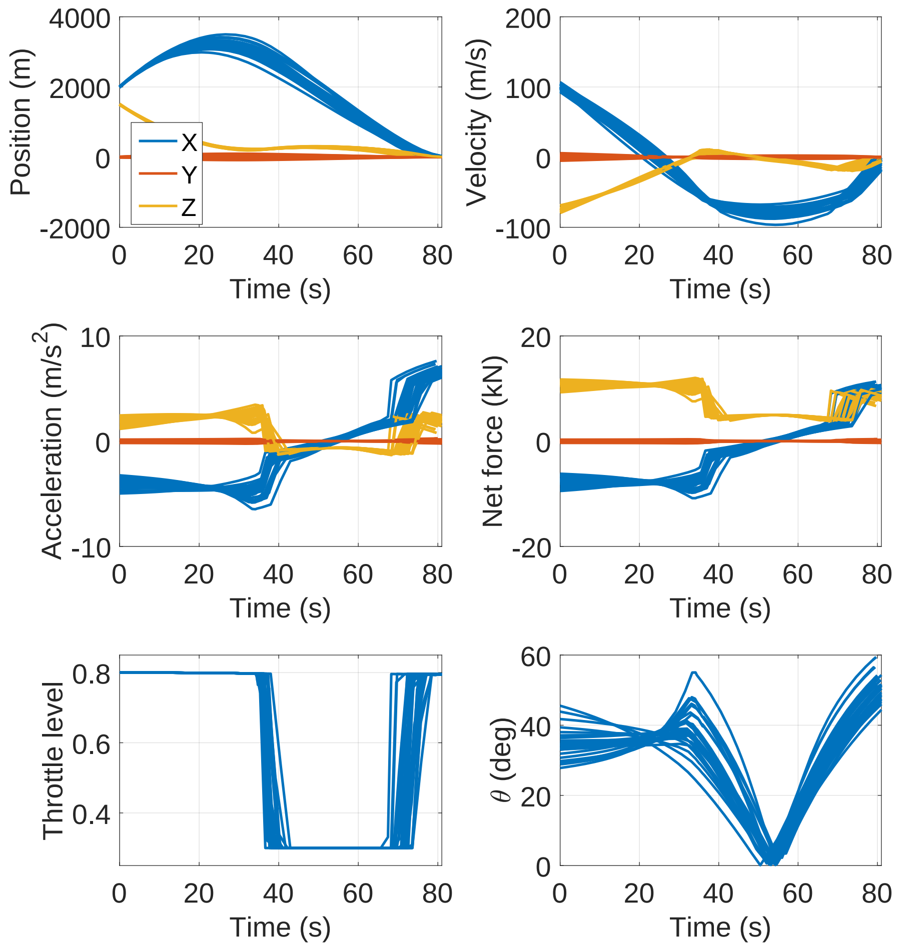
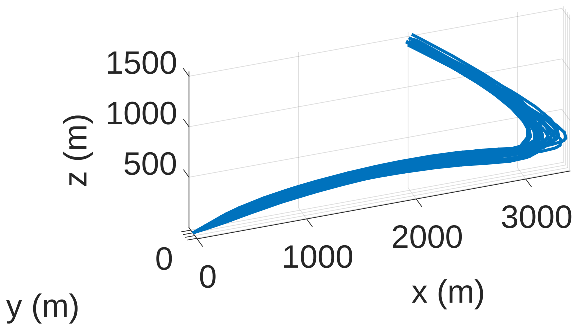
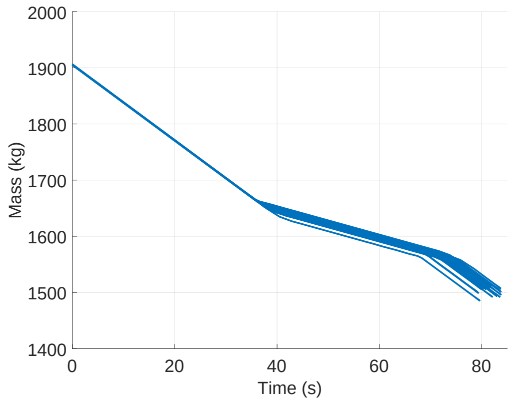


Fig. 7 Monte-Carlo analysis of hp pseudospectral convex method - states and controls, x component in blue, y component in red, z component in yellow (1000 cases).

h-methods are generally faster. For instance, if we look at the case with 25 total nodes the *h*-method takes 36.9 ms to compute a valid solution. However, they are characterized by a large error (in this case equal to 287 m in terms of position error, and 18.5 m/s for what regards the velocity). The proposed hp framework drastically reduces the error. For instance, when we consider 3 segments with 8 nodes each the error drops to 20 m and 0.6 m/s, at a relative increase of the CPU time (78.5 ms). In other words, a twice longer CPU time required to compute a solution reduces the error by a factor of 14 times in terms of landing position and 30 times in terms of landing velocity. Similar trends can be observed when a total number of nodes equal to 50, 75 or 100 is considered. Results suggest that one should not use *p*-methods. Even if they are very accurate, for relatively larger number of nodes the time required to compute a solution becomes much larger than *h*- and *hp*-methods. Instead, some combinations of *hp*-schemes work very well, and suggest



(a) Trajectories



(b) Mass profiles

Fig. 8 Monte-Carlo analysis of hp pseudospectral convex method - trajectories and mass consumption (1000 cases).

that they provide an excellent trade-off policy between accuracy and CPU time. For instance, when we have 50 nodes, if we use 7 segments and 7 nodes per segment results are faster than the h-methods (75.9 ms against 80.1 ms), and more accurate as well, with an error of about 20 m obtained with hp-transcription and 118 m when we use the h-method. Same conclusions can be drawn for the velocity. In general these results suggest that it is better to adopt an hp method with a smaller number of nodes than a pure h method with a larger number of nodes. For instance, even if we would use 75 or 100 nodes with the h-transcription the results obtained by using 50 hp nodes would still be both more accurate and faster. It is interesting to note that even 25 node, split into 12 segments containing 2 collocated nodes each (that is, a quasi-h method), are already three times more accurate than an h-method with 100 nodes, and almost twice faster. Finally, note that the proposed method is always much faster than SPARTAN, which takes about 0.4, 3.3, 7.7, and 20.0 s to solve the cases with 25, 50, 75 and 100 nodes, respectively. These results are in line with the expectations, that is, the proposed pseudospectral convex framework is much faster than the traditional pseudospectral NLP-based methods. Results suggest that in this case the saving in CPU time ranges from 5-10 times for the case of 25 nodes to 10-320 times for the case associated with 100 nodes, giving a wide possibility of trade-off between CPU time and accuracy, which can be performed according to the specific architecture present onboard.

Table 3 Comparison of accuracy and CPU times.

Method	Nodes $hp + 1$	p	h	CPU time (μ, ms)	CPU time (1σ, ms)	Landing error pos (m)	Landing error vel (m/s)
h	25	1	24	36.9	64.0	286.7114	18.4522
hp	25	2	12	57.7	51.0	18.5052	1.4484
hp	25	3	8	74.1	0.4	89.0568	4.5097
hp	25	4	6	63.9	12.3	89.8285	2.7811
hp	25	6	4	65.4	9.4	81.1201	1.9056
hp	25	8	3	78.5	6.3	20.4619	0.6164
hp	25	12	2	46.8	0.6	50.9700	1.2186
p	25	24	1	68.8	0.7	23.7977	0.5902
h	50	1	49	80.1	17.8	117.9673	9.0420
hp	50	7	7	75.9	0.7	20.2320	0.6338
p	50	49	1	382.5	3.0	5.7291	0.1419
h	75	1	74	122.4	7.2	75.3325	5.9808
hp	75	2	37	129.7	76.5	76.5179	1.8261
hp	75	37	2	802.7	553.4	5.4904	0.1323
p	75	74	1	2124.9	1411.9	2.6610	0.0633
h	100	1	99	99.5	35.0	54.9366	4.4727
hp	100	3	33	143.9	76.8	27.5502	0.6603
hp	100	9	11	192.4	3.2	15.4182	0.4590
hp	100	11	9	247.9	1.8	10.9618	0.4699
hp	100	33	3	1753.9	5.9	1.2345	0.0385
p	100	99	1	2632.4	1214.0	1.4230	0.0349

V. Conclusions

In this work a new hybrid framework consisting of hp-pseudospectral methods and convex optimization has been proposed. The purpose was to reduce the computational gap with respect to standard convex schemes applied to descent and landing scenarios, while increasing the accuracy of the solution computed by the optimization process beneath them. Results confirm that the proposed technique leads to a significant reduction of the CPU times with respect to the previous generation of pseudospectral convex methods, while being at the same time more accurate than standard convex optimization. It was shown how the method can be interpreted as generalized transcription method, able to reduce to the standard convex form or to the full pseudospectral method according to the choice of h and p, leading therefore to a unified transcription method.

The method was compared with results obtained by using an NLP-based approach, and its reliability was assessed by running Monte Carlo campaigns with multiple uncertainties in terms of position, velocity, mass and time of flight. The trade-off provided by the morphing transcription makes this class of methods an appealing alternative to the standard convex approaches for powered descent and landing scenarios, as the one described in this paper, and confirms its high-accuracy, and at the same time its potential as real-time capable method for this type of applications.

References

[1] space.com, "http://www.space.com/31420-spacex-rocket-landing-success.html," , 2015. URL `http://www.space.com/31420-spacex-rocket-landing-success.html`, retrieved on 19-Nov-2018.

[2] blueorigin.com, "Introducing New Glenn," , 2018. URL `https://www.youtube.com/watch?v=BTEhohh6eYk`, retrieved on 19-Nov-2018.

[3] Klevanski, J., Ecker, T., Riehmer, J., Reimann, B., Dumont, E., and Chavagnac, C., "Aerodynamic Studies in Preparation for CALLISTO - Reusable VTVL Launcher First Stage Demonstrator," *69th International Astronautical Congress (IAC), Bremen, Germany*, 2018.

[4] Lu, P., "Introducing Computational Guidance and Control," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 2, 2018, pp. 193–193. doi:10.2514/1.g002745, URL `https://doi.org/10.2514/1.G002745`.

[5] Bennett, F., "Lunar descent and ascent trajectories," *Aerospace Sciences Meetings, American Institute of Aeronautics and Astronautics*, 1970. doi:10.2514/6.1970-25, URL `http://dx.doi.org/10.2514/6.1970-25`.

[6] Jungmann, J., "Gravity turn trajectories through planetary atmospheres," American Institute of Aeronautics and Astronautics, 1967. doi:10.2514/6.1967-596, URL `http://dx.doi.org/10.2514/6.1967-596`.

[7] McInnes, C. R., "Gravity-Turn Descent from Low Circular Orbit Conditions," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 1, 2003, pp. 183–185. doi:10.2514/2.5033, URL `http://dx.doi.org/10.2514/2.5033`.

[8] Ingoldby, R. N., "Guidance and Control System Design of the Viking Planetary Lander," *Journal of Guidance, Control, and Dynamics*, Vol. 1, No. 3, 1978, pp. 189–196. doi:10.2514/3.55763, URL `http://dx.doi.org/10.2514/3.55763`.

[9] Boyd, S., and Vandenberghe, L., *Convex Optimization*, 2004.

[10] Acikmese, B., and Ploen, S. R., "Convex programming approach to powered descent guidance for mars landing," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 5, 2007, pp. 1353–1366. doi:10.2514/1.27553, URL `https://arc.aiaa.org/doi/10.2514/1.27553`.

[11] Wang, Z., and Grant, M. J., "Minimum-Fuel Low-Thrust Transfers for Spacecraft: A Convex Approach," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 54, No. 5, 2018, pp. 2274–2290. doi:10.1109/TAES.2018.2812558.

[12] Tang, G., Jiang, F., and Li, J., "Fuel-Optimal Low-Thrust Trajectory Optimization Using Indirect Method and Successive Convex Programming," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 54, No. 4, 2018, pp. 2053–2066. doi:10.1109/TAES.2018.2803558.

[13] Zhang, Z., Wang, J., and Li, J., "Lossless convexification of nonconvex MINLP on the UAV path-planning problem," *Optimal Control Applications and Methods*, Vol. 39, No. 2, 2018, pp. 845–859. doi:10.1002/oca.2380, URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/oca.2380`.

[14] Liu, X., and Lu, P., "Robust Trajectory Optimization for Highly Constrained Rendezvous and Proximity Operations," No. 0 in Guidance, Navigation, and Control and Co-located Conferences, American Institute of Aeronautics and Astronautics, 2018. doi:10.2514/6.2013-4720, URL `https://doi.org/10.2514/6.2013-4720`.

[15] Lu, P., and Liu, X., "Autonomous trajectory planning for rendezvous and proximity operations by conic optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 2, 2013, pp. 375–389. doi:10.2514/1.58436, URL `https://arc.aiaa.org/doi/pdf/10.2514/1.58436`.

[16] Liu, X., "Fuel-Optimal Rocket Landing with Aerodynamic Controls," *Journal of Guidance, Control, and Dynamics*, 2018, pp. 1–13. doi:10.2514/1.g003537, URL `https://doi.org/10.2514/1.G003537`.

[17] Wang, Z., and Grant, M. J., "Hypersonic Trajectory Optimization by Sequential Semidefinite Programming," AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2017. doi:10.2514/6.2017-0248, URL `http://dx.doi.org/10.2514/6.2017-0248`.

[18] Liu, X., Shen, Z., and Lu, P., "Entry trajectory optimization by second-order cone programming," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 2, 2015, pp. 227–241. doi:10.2514/1.G001210, URL `https://arc.aiaa.org/doi/pdf/10.2514/1.G001210`.

[19] Dueri, D., Acikmese, B., Scharf, D. P., and Harris, M. W., "Customized Real-Time Interior-Point Methods for Onboard Powered-Descent Guidance," *Journal of Guidance, Control, and Dynamics*, 2016, pp. 1–16. doi:10.2514/1.g001480, URL `http://dx.doi.org/10.2514/1.G001480`.

[20] Scharf, D. P., Acikmese, B., Dueri, D., Benito, J., and Casoliva, J., "Implementation and Experimental Demonstration of Onboard Powered-Descent Guidance," *Journal of Guidance, Control, and Dynamics*, 2016, pp. 1–17. doi:10.2514/1.g000399, URL `http://dx.doi.org/10.2514/1.G000399`.

[21] Sagliano, M., Theil, S., Bergsma, M., D'Onofrio, V., Whittle, L., and Viavattene, G., "On the Radau pseudospectral method: theoretical and implementation advances," *CEAS Space Journal*, Vol. 9, No. 3, 2017, pp. 313–331. doi:10.1007/s12567-017-0165-5, URL `https://doi.org/10.1007/s12567-017-0165-5`.

[22] Ross, I. M., Sekhavat, P., Fleming, A., and Gong, Q., "Pseudospectral Feedback Control: Foundations, Examples and Experimental Results," *AIAA Guidance, Navigation, and Control Conference, Keystone, USA,*, 2006. doi:10.2514/6.2006-6354.

[23] Gill, P. E., Murray, W., and Saunders, M. A., *User's Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming*, University of California, San Diego, USA, 2008.

[24] Wächter, A., and Biegler, L. T., "On the implementation of an interior-point filter linesearch algorithm for large-scale nonlinear programming," *Math. Program. 106(1) , Springer-Verlag, New York, 2006*, 2006.

[25] Acikmese, A. B., and Ploen, S., "A Powered Descent Guidance Algorithm for Mars Pinpoint Landing," Guidance, Navigation, and Control and Co-located Conferences, American Institute of Aeronautics and Astronautics, 2005. doi:10.2514/6.2005-6288, URL `http://dx.doi.org/10.2514/6.2005-6288`.

[26] Sagliano, M., "Pseudospectral Convex Optimization for Powered Descent and Landing," *Journal of Guidance, Control and Dynamics, Accepted*, Vol. 41, 2017, pp. 320–334. doi:10.2514/1.G002818, URL `https://arc.aiaa.org/doi/pdf/10.2514/1.G002818`.

[27] Melenk, J. M., *hp-Finite Element Methods for Singular Perturbations*, Springer, 2004.

[28] Rao, A. V., "A Survey of Numerical Methods for Optimal Control," *AAS/AIAA Astrodynamics Specialist Conference, AAS Paper 09-334, Pittsburgh, PA, August 10 - 13*, 2009.

[29] Harris, M. W., and Acikmese, B., "Maximum Divert for Planetary Landing Using Convex Optimization," *Journal of Optimization Theory and Applications*, Vol. 162, No. 3, 2013, pp. 975–995. doi:10.1007/s10957-013-0501-7.

[30] Harris, M. W., and Acikmese, B., "Lossless convexification of non-convex optimal control problems for state constrained linear systems," *Automatica*, Vol. 50, No. 9, 2014, pp. 2304 – 2311. doi:https://doi.org/10.1016/j.automatica.2014.06.008, URL `http://www.sciencedirect.com/science/article/pii/S0005109814002362`.

[31] Elissar, "Description of DIDO Optimal Control software," , June 2015. URL `http://www.elissarglobal.com/`.

[32] Rao, A. V., Benson, D. A., L., D. C., Patterson, M. A., Francolin, C., I., S., and Huntington, G. T., "GPOPS: A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using the Gauss Pseudospectral Method," *ACM Transactions on Mathematical Software*, Vol. 37, No. 2, 2010, pp. 1–39. doi:10.1145/1731022.1731032.

[33] Karniadakis, G., and Sherwin, S., *Spectral/hp element methods for computational fluid dynamics*, Oxford University Press, 2013.

[34] Darby, C. L., Hager, W. W., and Rao, A. V., "An hp-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Applications and Methods*, Vol. 32, No. 4, 2011, pp. 476–502. doi:10.1002/oca.957, URL `http://dx.doi.org/10.1002/oca.957`.

[35] Blackmore, L., Acikmese, B., and Scharf, D. P., "Minimum-Landing-Error Powered-Descent Guidance for Mars Landing Using Convex Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 4, 2010, pp. 1161–1171. doi:10.2514/1.47202, URL `http://dx.doi.org/10.2514/1.47202`.

[36] Mao, Y., Szmuk, M., and Acikmese, B., "A Tutorial on Real-time Convex Optimization Based Guidance and Control for Aerospace Applications," *2018 Annual American Control Conference (ACC)*, 2018, pp. 2410–2416. doi:10.23919/ACC.2018.8430984.

[37] Martin, M. S., G.Mendeck, Brugarolas, P. B., Singh, G., and Serricchio, F., "In-Flight Experience of the Mars Science Laboratory Guidance, Navigation, and Control System for Entry, Descent, and Landing," *9th International ESA Conference on Guidance, Navigation, and Control Systems*, 2014.

[38] Domahidi, A., Chu, E., and Boyd, S., "ECOS: An SOCP Solver for Embedded Systems," *European Control Conference*, 2013.