POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Application of hybrid differential dynamic programming in orbital elements for low thrust trajectory optimisation

Laurea Magistrale in Space Engineering - Ingegneria Spaziale

Author: Simone Carminati

Advisor: Prof. Camilla Colombo

Co-advisor: Dr. Marco Nugnes

Academic year: 2022-2023

1. Introduction

Thanks to the technological development in the electric propulsion, the space sector is exploiting the use of low-thrust systems to an ever-widening range of mission types. In fact, they are increasingly being chosen as propulsion solutions for both interplanetary and planetary transfers. The main characteristics of such engines are a high specific impulse, and a low level of thrust that they can deliver. Consequently, the time of flight associated to a low-thrust trajectory is considerably longer than the chemical thrusters counterpart.

In this context, it is no longer possible to consider approximations such as impulsive manoeuvres to model the dynamics, and the latter requires a formulation that includes continuous thrust. The consequence from a numerical optimisation point of view is a significant increase in the variables involved, and the design of such trajectories turns into the solution of a highly non-linear large-scale optimal control problem.

For this purpose, several optimisation algorithms exist, generally divided into two macro-families: direct and indirect methods. One approach that is relevant in this context is Differential Dynamic Programming (DDP), which combines the advantages of both families (in short, accuracy on one hand, robustness on the other) with a limited computational cost. Indeed, given a number N of nodes to discretise a trajectory, direct methods typically have a computational cost proportional to N^3, whereas the one associated to DDP is proportional to N. Despite its potential, most of the times this method has been used for orbital trajectory design, the problem has been formulated as a function of classical Cartesian coordinates as state variables; in fact its application by exploiting different sets of variables has found little use in this field.

The aim of this thesis work is to adapt this state-of-the-art optimisation method, called Hybrid Differential Dynamic Programming (HDDP) [2], to a new set of variables often used in celestial mechanics, namely orbital elements. This set has the typical feature to evolve more smoothly and regularly rather than Cartesian coordinates. Thanks to this property, the coupling between orbital elements and DDP can prove particularly useful in speeding up the convergence and ensuring the success of the method even when dealing with problems particularly sensitive to variables' oscillation, as it is the case of a low-thrust multi-revolution transfer.

Some practically relevant test cases in the context of low-thrust trajectory design are used to validate the aforementioned hypotheses.

2. Mathematical background

2.1. Optimal control problem

A dynamical system, can be described in two alternative forms, expressed by the two systems of equations:

$$\dot{\boldsymbol{x}}_k = \boldsymbol{f}_k(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}; t_k) \tag{1}$$

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}_k(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{w}) \tag{2}$$

where $\boldsymbol{x}_k = [x_1, ..., x_n]^T$ is a vector of n variables that represents the state vector, $\boldsymbol{u}_k = [u_1, ..., u_m]^T$ is a m-dimensional vector representing the control input, and $\boldsymbol{w} = [w_1, ..., w_p]^T$ is the vector of "static" controls, i.e. time-independent parameters that can affect the problem.
Solving an optimal control problem, given a dynamical system, consists in finding the control law $\boldsymbol{u}(t_k)$ such that the objective function J is minimised:

$$J := \sum_{j=0}^{N} L_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + \phi(\boldsymbol{x}_{N+1}) \tag{3}$$

where ϕ is a function that depends only by the final state, whereas L is a merit function depending on all the remaining stages and respective controls of the trajectory. The general non linear optimal control problem can include some constraints that shall be respected: stage-constraints $\boldsymbol{g}$, which limits the trajectory in a prescribed region, and final state constraints $\boldsymbol{\psi}$, which represents the desired final state of the path.

2.2. Differential dynamic programming

Differential dynamic programming is a modern way to face a large-size optimal control problem. It consists of an iterative method that produces consecutive trajectories associated with a gradually improved value of an objective function J to be minimized, until a local minimum is reached. The method is composed by two main phases: a backward recursion and a forward propagation. In the backward recursion necessary optimality conditions are solved to select an optimal control law starting from a first guess policy and its trajectory, in order to minimise the objective function. In the forward propagation the optimal control law is used to compute the new improved trajectory and the new values for the cost function and constraints' violation.

2.2.1. Backward sweep

This process is based on Bellman's optimality principle [3], the process is based on bellman's optimality principle, which is formulated mathematically in the recursive Hamilton-Jacobi-Bellman equation:

$$J_k(\boldsymbol{x}_k) := L_k(\boldsymbol{x}_k, \boldsymbol{u}_k) + J^*_{k+1}(\boldsymbol{x}_{k+1}) \tag{4}$$

where J_k is defined as the cost-to-go function, a variable that represent the cost value of the trajectory starting from node k to the final one; J_{k+1} is the optimal value of the cost-to-go function associated to the optimal path that goes from node $k+1$ to the final one; L_k is a the stage cost function.
The DDP optimisation technique is based on the quadratic expansions of all the variables in Eq. 4 starting from a reference control law and its associated trajectory. The quadratic expansion overcomes the well-known problem in dynamic programming denoted as "curse of dimensionality" that is the impossibility to apply Bellman's principle due to the fast increase in the number of variables and functions to be stored to solve the problem. The optimal control variation minimising the second-order approximation is obtained considering the first order derivative $\frac{d}{d(\delta u_k)} J_k = 0$, which in the unconstrained case has the form:

$$\delta \boldsymbol{u}_k = \boldsymbol{A}_k + \boldsymbol{B}_k \delta \boldsymbol{x}_k \tag{5}$$

The minimisation of the cost-to-go function at each stages provides the coefficients $\boldsymbol{A}_k$, $\boldsymbol{B}_k$ that must stores for each time instant in the backward regression. The optimal control variation in Eq. 5 is replaced in Eq. 4 to retrieve the optimal cost function at stage k and its partial derivatives. Furthermore, during this process also some constant terms appear in the formulation, which represents the expected reduction ER of the cost-to-go function J_k due to the control variation. At this point the process moves to the next step and repeat itself up to the initial point of the trajectory.

2.2.2. Forward propagation

Once the coefficients of the feedback control policy variation are retrieved for all the stages N and the final value of the expected reduction ER_0 is computed, the first phase shall be considered completed and the propagation can start. Basically it consists in a forward process where, at each stage k, the new control law and the next point in the trajectory are defined. This is done following these recursive formulas, imposing that $\delta \boldsymbol{x}_0 = \boldsymbol{0}, \delta \boldsymbol{u}_0 = \boldsymbol{A}_0$.

$$\begin{cases} \boldsymbol{u}_k = \bar{\boldsymbol{u}}_k + \epsilon \boldsymbol{A}_k + \boldsymbol{B}_k(\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k) \\ \boldsymbol{x}_{k+1} = \boldsymbol{F}(\boldsymbol{x}_k, \boldsymbol{u}_k) \end{cases} \tag{6}$$

where ϵ is a parameter which is initially set equal to 1, and then progressively decreased if needed. At this point it is required to check whether the trajectory just calculated is "close enough" to the previous one, to be sure that the second-order expansion approximation is not violated. At the end of the forward propagation the values of J_{new} and J are compared to check if the variation of the cost-function is similar to the value of the expected reduction ER_0 computed on the whole reference trajectory. If this is not the case, the parameter ϵ is halved and the process is repeated in order to generate a solution closer to the reference one. Otherwise, the new control law and the new trajectory are set as the reference ones and another backward regression can start.

2.3. Hybrid differential dynamic programming

Hybrid differential dynamic programming represents the state-of-the art for DDP. It consists in the inclusion of some robust mathematical techniques to overcome some limitations associated to the DDP algorithm such as:

- the inclusion of constraints in the formulation
- correction of the Hessian $J_{uu,k}$ in case it is not positive definitive which is a second order necessary condition for the minimisation process

2.3.1. Trust region

Trust region is a robust mathematical technique that fulfils a dual purpose in the algorithm. The first consists of replacing the classical line search method during the forward propagation with a technique that is recognised in the literature as being more numerically stable. The second consists in introducing an efficient method capable of guaranteeing the positive definite condition of the Hessian. The trust region method is able to define both the optimal control law coefficient and satisfy the state constraints associated to the maximum magnitude that the constant term $\boldsymbol{A}_k$ can have. In this way the control variation at each step is restricted in a certain region around the reference one, enhancing numerical stability during the backward sweep and granting that the hessian remain positive definite. Operatively, the method consists in solving the following subproblem, called trust region quadratic subproblem TRQP$(J_{u,k}, J_{uu,k}, \Delta)$, at each time step k:

$$\begin{aligned} &\min_{\delta \boldsymbol{u}_k} (J_{u,k}\delta \boldsymbol{u}_k + \frac{1}{2}\delta \boldsymbol{u}_k^T J_{uu,k}\delta \boldsymbol{u}_k) \\ &\text{s.to} \quad ||\boldsymbol{D}\delta \boldsymbol{u}_k|| \leq \Delta \end{aligned} \tag{7}$$

where Δ is the current trust region radius and $\boldsymbol{D}$ is a positive definite scaling matrix. The solution to this subproblem consists in an iterative procedure that produces an adequate shift of $J_{uu,k}$, in order to obtain a solution that respects the magnitude constraint. The solution provide the shifted Hessian $\widetilde{J}_{uu,k}$ that is used to compute the coefficients of the feedback control variation policy. Indeed, given the general form of the control variation, which includes the presence of static parameters and final state constraints:

$$\delta \boldsymbol{u}_k = \boldsymbol{A}_k + \boldsymbol{B}_k \delta \boldsymbol{x}_k + \boldsymbol{C}_k \delta \boldsymbol{w} + \boldsymbol{D}_k \delta \boldsymbol{\lambda} \tag{8}$$

the final coefficients are retrieved in this way:

$$\begin{cases} \boldsymbol{A}_k = -\widetilde{J}_{uu,k}^{-1} J_{u,k} \\ \boldsymbol{B}_k = -\widetilde{J}_{uu,k}^{-1} J_{ux,k} \\ \boldsymbol{C}_k = -\widetilde{J}_{uu,k}^{-1} J_{uw,k} \\ \boldsymbol{D}_k = -\widetilde{J}_{uu,k}^{-1} J_{u\lambda,k} \end{cases} \tag{9}$$

2.3.2. Constraint handling

HDDP manages both stage constraints and final state constraints, but in two different ways.
Stage constraints are treated during the backward regression using the range-space active set method, which assesses whether a constraint is active at the current stage k and if so, linearises the latter and modifies the coefficients obtained

from the Eq. 9 in order to ensure its fulfilment. Final state constraints instead, are taken into account thanks to an augmented lagrangian approach, which reformulates the problem including some weights related to the violation of the constraints inside the objective function. This does not grant its exact fulfilment, but ensures that at each iteration, as the original cost function is minimised, the violation of the constraint itself is also gradually reduced. Operatively, it consists in modifying ϕ in the following way:

$$\widetilde{\phi} = \phi + \boldsymbol{\lambda}^T \boldsymbol{\psi} + \sigma ||\boldsymbol{\psi}||^2 \tag{10}$$

where now $\boldsymbol{\lambda}$ are variables which are modified at each iteration at the end of the backward process, in order to maximize J, solving the TRQP$(-J_\lambda, -J_{\lambda\lambda}, \Delta)$. In this way the variation $\delta\boldsymbol{\lambda}$ can be computed and, consequently, also the control law. In HDDP, a variation of the penalty parameter σ is also implemented in the event of an increase in the final constraints violation.

2.3.3. STM approach

An other important aspect introduced by the HDDP is the use of STMs for the evaluation of the optimal cost function partial derivatives. Before its introduction, the derivatives that appears in the formulation are related to the discrete transition function $\boldsymbol{F}$ which, for complex dynamical system expressed by means of differential system $\boldsymbol{f}$, is difficult to retrieve analytically. This implies complex procedures to approximate the discrete transition function and compute the derivatives, introducing several approximations. The STM approach, allows to compute the derivatives directly by the knowledge of $\boldsymbol{f}$ without pass necessarily by $\boldsymbol{F}$. This can be done, providing the analytical form of the STMs, whenever possible, or by integrating the system of equation:

$$\begin{cases} \dot{\boldsymbol{\Phi}}_k^1 = \boldsymbol{f_X} \boldsymbol{\Phi}_k^1 \\ \dot{\boldsymbol{\Phi}}_k^2 = \boldsymbol{f_X} \bullet \boldsymbol{\Phi}_k^2 + \boldsymbol{\Phi}_k^{1T} \bullet \boldsymbol{f_{XX}} \bullet \boldsymbol{\Phi}_k^1 \end{cases} \tag{11}$$

subjected to the initial condition $\boldsymbol{\Phi}_k^1(t_k) = \boldsymbol{I}_{n+m+p}$ and $\boldsymbol{\Phi}_k^2(t_k) = \boldsymbol{0}_{n+m+p}$.

3. HDDP with orbital elements

The algorithm presented up to this point is now being used to study problems formulated as a function of orbital elements. The structure of the optimiser remains the same, however, attention must be paid to certain aspects due to the intrinsic nature of the new set of variables, as also mentioned in [1].

- This set is defined by variables that have very different orders of magnitude (the semi-major axis is typically much larger than the other elements). This causes the matrices involved to be ill-conditioned and scaling becomes of paramount importance for the method to be successful.
- The second is represented by the choice of the elements exploited.
  The classical Keplerian orbital element has a strong physical meaning, representing dimension, shape and orientation in space of the current orbit, but are affected by several singularity conditions. In cases when $e = 0$, $e = 1$ and $i = 0$, which are often situations of interest for practical applications, the system dynamics diverges, causing the failure of the method. Every time the path approaches these singularity conditions, even without reaching them, the behaviour of the optimiser worsens, slowing down and sometimes completely preventing convergence.
  Modified equinoctial elements on the other hand, does not present any singularity and in general and generally behave more regularly than the previous set, but their evolution is not easy to interpret as they have no physical meaning.

Once the critical points of these new variables have been presented, however, the advantages shall also be underlined. Firstly, as already mentioned, the time evolution of these elements is more regular than the Cartesian coordinates, which greatly facilitates the optimiser's task. Secondly, the dynamics of such elements are typically reported in a reference system $[\hat{\boldsymbol{t}}, \hat{\boldsymbol{n}}, \hat{\boldsymbol{h}}]$, which may also be more convenient for defining an initial control law that is more intuitive and closer to the final desired target.

4. Test cases

In this thesis four test cases are investigated in order to proof different aspects of the presented trajectory design methodology.

4.1. Direct transfer

The first case involves the use of classical Keplerian elements as variables and deals with a continuous thrust transfer without complete revolutions in the planetary domain. This test case is used to evaluate the robustness of the method, considering the significant difference between initial guess and final target orbit. Once the first optimal solution has been found, the specific impulse was then modified through a continuation scheme to make the propulsive solution viable. Finally, for sake of completeness, a sensitivity analysis was then done by varying the time of flight. This last analysis proved useful in order to evaluate the behavior of the optimiser as we approach a singularity situation.

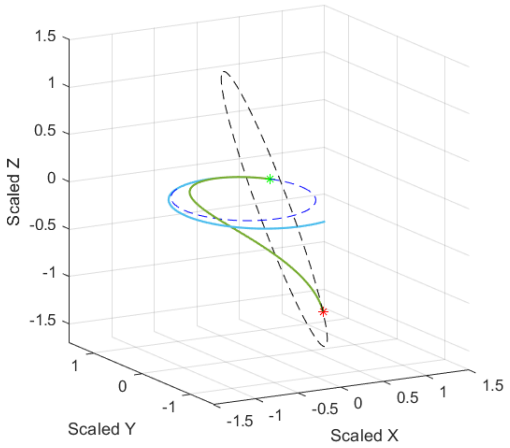

Figure 1: Comparison initial guess and optimal path

4.2. Earth-Mars rendezvous

This test can be found in several other papers dealing with trajectory optimisation and has also been treated here as a comparison parameter with the previous literature. Stage constraints on the maximum deliverable thrust are also introduced, differently from the previous case, in order to highlight how effectively these constraints are handled by the method. Modified equinoctial elements are used as set of variables to avoid singularities. The results obtained from this case study suggests a significant improvement in the convergence speed if the new variables are used.

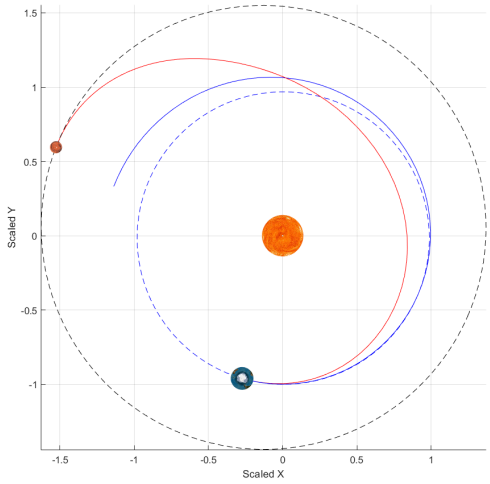

Figure 2: Comparison between initial guess and optimal solution

4.3. Multi-revolution transfer

This study case is particularly relevant for multi-revolution applications. Indeed, this kind of transfer is challenging to deal with Cartesian coordinates because of the great sensitivity of the state variables. An approach exploiting orbital elements can open the door to the possibility of effectively optimising even trajectories with a large number of revolutions.
This test case considers a 45 revolutions transfer from an inclined eccentric orbit to a GEO. To fully evaluate performances in such a case study, multiple dynamics were written as a function of different independent variables exploiting the Sundman transform. This operation allows to review the way the discretization of the orbit occurs and make it more homogeneous than the one obtained in case of time is used as independent variable.
It is important to emphasise that this transformation can be used if and only if the state variables are expressed in terms of orbital elements. For this reason, there is an additional advantage of using such variables in the multi-revolution case.

	Independent Variable		
	t	**E**	θ
Final Mass [kg]	659.516	652.274	655.441
TOF [days]	18.4	23.3	23.5
N_{rev} [-]	45	45	45
Final θ [deg]	246.04	80.37	38.67
J^* [N·s]	0.889246	0.889269	0.846263
$\boldsymbol{N_{iter}}$ [-]	**78**	**36**	**42**

Table 1: Most relevant data associated with the 3 different optimal solutions

4.4. OneWeb case

To conclude, it has been decided to apply the method presented to a real low-thrust mission with large number of revolutions currently under development. The real application selected consider the orbit raising strategy of the OneWeb satellite's constellation, that pass from an initial almost circular orbit at a mean altitude of 592 km above the Earth's surface, to a circular one located at 1200 km altitude. Given the specifics of the equipped engine, for this work a transfer of 400 revolutions has been selected as a primary choice.
The method converges in a small number of iterations, producing a feasible optimal trajectory. This is an important result that highlights the potential of the method because of the impossibility to achieve an optimal solution by using classical Cartesian coordinates.

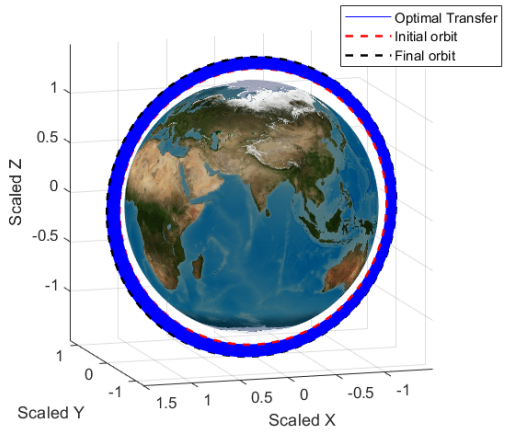

Figure 3: Representation of the optimal solution, the initial and final target orbits

5. Conclusions

As mentioned in the introduction, the focus of this thesis is on the coupling between HDDP method and orbital elements as state representation of the dynamics. The purpose behind this study was to assess the improvements and limitations imposed by such a coupling, and in particular to evaluate the possible extension of the optimisation method to a class of problems that are extremely sensitive to variables' oscillation and at the same time which are considered to be an hot topic nowadays, the multi-revolution low thrust transfers.
As confirmed by numerical results, an increase in convergence speed and the possibility of dealing with problems with a very large number of revolutions involved has been verified for such a combination.
More investigations are required to limit the computational effort related to the STMs calculation in order to speed up the algorithm, the efficient introduction of perturbations into the dynamics equations, and the possibility of dealing with multiphase dynamics, which were not explored in detail in this study.

References

[1] Nugnes M. Colombo C. Low-thrust trajectory optimisation through differential dynamic programming method based on keplerian orbital elements. *70th International Astronautical Congress (IAC), Washington D.C., United States, paper C1.1.2*, 2019.

[2] Russell R.P. Lantoine G. A hybrid differential dynamic programming algorithm for constrained optimal control problems. part 1: Theory. *J. of Optimization Theory and Applications, vol 154. pp 382-417.*, 04 2012. DOI: 10.1007/s10957-012-0039-0. URL: https://doi.org/10.1007/s10957-012-0039-0.

[3] Bellman R. *Dynamic Programming*. Princeton University, 1957. ISBN: 978-0691079516.