Perturbed Lambert's Problem Solver based on Differential Algebra Optimization

Paolo Panicucci*
Institut Supérieur de l'Aéronautique et de l'Espace - SUPAERO, 10 Avenue Edouard Belin, 31400, Toulouse, France

Vincent Morand† and Denis Hautesserres‡
Centre National des Etudes Spatiales, 18 Avenue Edouard Belin, 31400, Toulouse, France

Classical Lambert's problem is an astrodynamical problem implemented in industrial and scientific software to solve this specific two-boundary value problem under the hypothesis of a Keplerian dynamic. To design optimal trajectories or to compute initial guesses for least-square orbit determination problems, it must be solved and implemented. Earlier works develop numerical and analytical techniques to solve the classical Lambert's problem in the mono-revolution and multi-revolution cases. On the one hand, this results in fast-computing and efficient methods that are employed in state-of-the-art software. On the other, the dynamical model is simplistic and the actual final position of the satellite differs by several kilometers once the revolution number increases. A way to obtain a more close-to-reality solution is to consider a more complete dynamical model by taking into account orbital perturbing forces such as aerodynamic drag and perturbing gravity potential. As previously-introduced algorithms do not consider a perturbed dynamics, this paper develops an optimization algorithm based on Taylor Differential Algebra to solve the perturbed Lambert's problem. The operations defined in the algebra allow the computation of the polynomial approximation of the final state propagation as a function of the initial state to be computed. This polynomial expansion is used to reduce the final position error as in thrust-region optimization. A wide range of numerical simulation is performed in order to have a clear view of the algorithm performances. Test cases have been chosen between the main orbit families (LEO, MEO, GEO, HEO and GTO) in order to have a complete and clear overview of the developed algorithm. Moreover the influence of the polynomial order is studied and a preference expansion order is selected to maximize the performance index. Obtained results are promising and further developments are proposed to increment algorithm performances.

I. Introduction

The classical Lambert's problem (CLP) is a well-known problem in astrodynamics consisting in deducing the initial velocity of the spacecraft $\boldsymbol{v}_{in}$ knowing the initial position $\boldsymbol{r}_{in}$, the final positions $\boldsymbol{r}_{fin}$ and the time interval Δt between the two. This two-point boundary value problem has fascinated scholars since the 18th century and has been intensively threated in most of the classical astrodynamics reference books [1–3]. A recent advance in high-performing computation of the CLP solution has

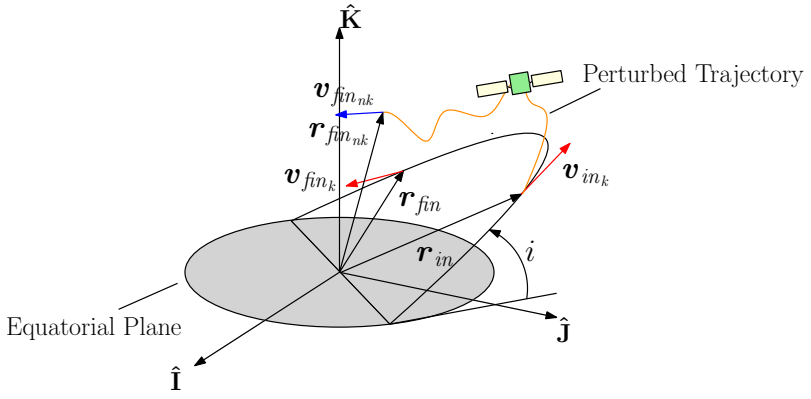


Figure 1 Difference in propagation between the CLP and a PLP (modified from [4]).

been presented by Izzo [5] providing machine-precision results with only two iterations in the mono-revolution case.

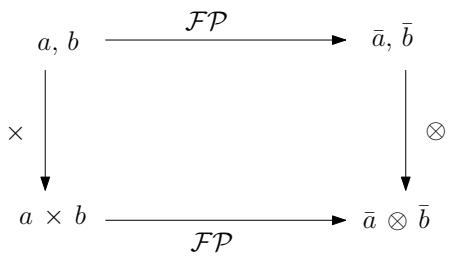
*PhD Candidate, ISAE-Supaero, DISC, paolo.panicucci(hat)isae-supaero.fr
†Space Flight Dynamics Engineer, CNES, ISL department, vincent.morand(hat)cnes.fr
‡Space Flight Dynamics Engineer, CNES, ISL department, denis.hautesserres(hat)cnes.fr

As the spacecraft motion is perturbed, the CLP solution does not provide a satisfying initial velocity $\boldsymbol{v}_{in}$ to be used in initial orbit determination (IOD) solver, such as the ones developed at CNES. This is due to the non-linear behavior that the Equations of Motion (EOMs) show for long propagation time or highly non-linear orbits, such as Tundra or GTO, with error between the perturbed final position and the CLP final position greater than $10^4\ km$. Moreover, as a close-to-reality initial guess for IOD algorithms is needed, it seems straightforward to consider a more complex and complete problem: the perturbed Lambert's problem (PLP). The schematic representation of this situation is shown in Figure 1 where problem geometrical definition and notations are shown.

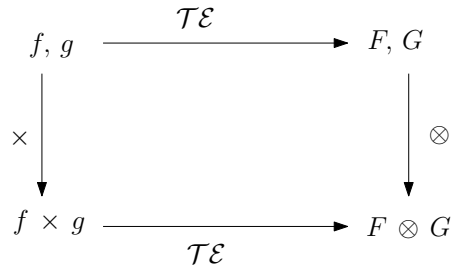
The main limitation in solving the PLP is due to the computation of dynamical model derivatives to be used in the iterative optimization algorithm. Software-implemented solutions are based on finite difference approximation of the dynamical model Jacobian or, when possible, on the analytical derivatives computation. On the one hand, the first approach can be used with all types of dynamical model but it implies a high computation time as the EOMs must be solved multiple times in order to compute the Jacobian. On the other hand, the second technique allows the direct gathering of the derivative values but it requires to infer analytically the derivative expressions. Recently, an alternative approach has been developed that is based on the computation - in a computer environment - of the Taylor expansion of a function up a to a given order: Taylor Differential Algebra (TDA). Therefore, this method allows the exact computation of all derivatives up to a desired order as if an analytical gathering had been performed and by keeping computational time limited. The first TDA application has been the computation of parametric maps in optical systems where Taylor expansions of particle trajectories have been gathered [6]. Recent work has used TDA techniques in space mechanics to solve parametric equations or to take into account uncertainty propagation [4, 7–12].

In this paper, an iterative optimization strategy has been developed in a Differential Algebra (DA) framework to solve the PLP. The optimization can be classified among the shooting iterative methods with a thrust-region approach. After a brief introduction of the TDA, the main elements of the PLP solver are presented: the considered dynamical model, the optimization algorithm and the chosen initial condition. Finally, the numerical results obtained from a large number of test cases are presented and commented on in order to understand the algorithm performances.

II. Taylor Differential Algebra



(a) Commutative diagram for real numbers. $\mathcal{FP}$ is the floating point number conversion operation.



(b) Commutative diagram for functions. $\mathcal{TE}$ is the Taylor expansion conversion operation.

Figure 2 Commutative diagrams for a generic operation $\times$ and a generic conversion operation (modified from [6])

In this section a general overview of the TDA is presented. Interested readers can consult References 6 and 8 for further details. The basic idea of the TDA is to extend the algebraic operation on numbers to function in a computer environment. In a strict sense, neither numbers nor functions can be *exactly* represented in a computer because of the finite amount of information that a machine manages. From a real number point of view, they are approximated by the floating point numbers in a way that the diagram in Figure 2a commutes either by firstly applying the operation $\times$ on the real number and then converting the result to floating point numbers or by firstly converting to the floating point numbers and then applying the adjoint operation $\otimes$. Analogously, a commuting diagram, in Figure 2b, can be defined with a generic function in such a way that the Taylor expansion coefficients extraction and the operation $\times$ on function commutes thanks to the definition of an adjoint operation $\otimes$. Moreover, by introducing multiplication, scalar multiplication and addition, the defined mathematical structure turns into a commutative algebra and, by equipping the algebra of a analytic differentiation and integration, the formal definition of a differential algebra is established [6].

III. The Dynamical Model

In this section the dynamical model used in the initial condition propagation is presented. The EOMs consider the central acceleration of the planet, the atmospheric drag and the perturbation due to a non-spherical body [13]:

$$\ddot{\boldsymbol{r}} = -\frac{\mu}{|\boldsymbol{r}|^3}\boldsymbol{r} + \frac{1}{2}\frac{C_D A}{m}\varrho\,|\boldsymbol{v}_{rel}|\,\boldsymbol{v}_{rel} + \nabla_{\boldsymbol{r}}\Phi \qquad \text{where} \qquad \varrho = \varrho_0\, e^{-\frac{(|\boldsymbol{r}|-r_e)}{h}} \tag{1}$$

where μ is the planetary gravitational parameter, $\boldsymbol{r}$ is the position vector, C_D is the satellite drag coefficient, A is satellite effective drag area, $\boldsymbol{v}_{rel}$ is satellite relative velocity with respect to the atmosphere, m is the satellite mass, ϱ is the atmospheric density, ϱ_0 is the sea-level reference density, r_e is the Earth equatorial radius, h is the scale height, $\nabla_{\boldsymbol{r}}$ is the nabla operator in Cartesian coordinates and Φ is the perturbation gravity potential.
Moreover, the perturbation gravity potential is expressed as follows [14]:

$$\Phi = -\frac{\mu}{|\boldsymbol{r}|}\left[\sum_{k=2}^{n_{\text{zo}}}\left(\frac{r_e}{|\boldsymbol{r}|}\right)^k J_k P_n(\sin\varphi) - \sum_{k=2}^{n_{\text{zo}}}\sum_{m=1}^{n_{\text{te}}}\left(\frac{r_e}{|\boldsymbol{r}|}\right)^k \left[C_k^m \cos m\lambda + S_k^m \sin m\lambda\right] P_k^m(\sin\varphi)\right] \tag{2}$$

where $(\varphi,\ \lambda)$ are the satellite geocentric latitude and longitude. The integer n_{zo} is the zonal truncation degree and the integer n_{te} is the tesseral truncation order. The parameters J_k, C_k^m, S_k^m denote the zonal and tesseral coefficients of the spherical harmonic expansion and their numerical values are chosen accordingly with the GRIM4S4 gravity model. Finally, $P_k(x)$ is the Legendre polynomial of degree k and $P_k^m(x)$ is the associated Legendre polynomial of degree k and order m.
From a computational point of view, the perturbations are computed by considering some important assumptions:

- The satellite geocentric latitude and longitude are computed by considering a flattened Earth with an equatorial radius r_e and a flattening parameter f.
- The atmosphere is spherically symmetrical and rotates at the same angular rate as the Earth.
- The atmospheric density parameters are constant in time.
- The perturbation gravity acceleration is computed with the algorithm given in [15].

The constant numerical values are reported in Table 1.

$m\ [kg]$	$A\ [m^2]$	$C_D\ [-]$	$\varrho_0\ [kg\,m^{-3}]$	$h\ [m]$	$a_e\ [km]$	$f\ [-]$	$n_{\text{zo}},\ n_{\text{te}}\ [-]$
1000	10	2.2	1.225	7000	6378.138	$3.353\,10^{-3}$	6

Table 1 Constants numerical value for the PLP force model.

IV. DA Thrust-Region Algorithm

In this section the optimization algorithm is exposed and commented on. The main idea is based on the possibility of extracting information from the Taylor expansion of the initial velocity $\boldsymbol{v}_{in}$ as a function of $\Delta\boldsymbol{r}^n$, i.e. the difference at step n between the non-Keplerian final position at step n $\boldsymbol{r}^n_{fin_{nk}}$ and the targeted final position $\boldsymbol{r}_{fin}$. Firstly the initial conditions, i.e. the known initial position $\boldsymbol{r}_{in}$ and the initial velocity at step n $\boldsymbol{v}^n_{in}$, are initialized in a DA framework:

$$\begin{cases}[\boldsymbol{r}_{in}] = \bar{\boldsymbol{r}}_{in} + \delta\boldsymbol{r}_{in}\\ [\boldsymbol{v}^n_{in}] = \bar{\boldsymbol{v}}^n_{in} + \delta\boldsymbol{v}^n_{in}\end{cases} \tag{3}$$

where the bar superscript denotes the constant part of the expansion and δ its derivatives up to order k.
Secondly, a TDA non-Keplerian propagation is performed to find the DA map between the non-Keplerian final position at step n and the initial conditions. Successively $\Delta\boldsymbol{r}^n$ is gathered:

$$\left[\Delta\boldsymbol{r}^n_{fin}\right] = \left[\boldsymbol{r}^n_{fin_{nk}}\right] - \left[\boldsymbol{r}_{fin}\right] = \Delta\bar{\boldsymbol{r}}^n_{fin} + \mathcal{M}_{\Delta\boldsymbol{r}^n_{fin}}\left(\delta\boldsymbol{r}_{in},\ \delta\boldsymbol{v}^n_{in}\right) \tag{4}$$

where $\mathcal{M}_g\,(f,h)$ is the mapping of g as a function of f and h and the δg is the variation of g with respect to the polynomial center. The map is then augmented with the $\delta\boldsymbol{r}_{in}$-identity map in order to allow the inversion:

$$\begin{bmatrix}\delta\Delta\boldsymbol{r}^n_{fin}\\ \delta\boldsymbol{r}_{in}\end{bmatrix} = \begin{bmatrix}\mathcal{M}_{\Delta\boldsymbol{r}^n_{fin}}\\ \mathcal{I}_{\boldsymbol{r}_{in}}\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{r}_{in}\\ \delta\boldsymbol{v}^n_{in}\end{bmatrix} \qquad \Rightarrow \qquad \begin{bmatrix}\delta\boldsymbol{r}_{in}\\ \delta\boldsymbol{v}^n_{in}\end{bmatrix} = \begin{bmatrix}\mathcal{M}_{\Delta\boldsymbol{r}^n_{fin}}\\ \mathcal{I}_{\boldsymbol{r}_{in}}\end{bmatrix}^{-1}\begin{bmatrix}\delta\Delta\boldsymbol{r}^n_{fin}\\ \delta\boldsymbol{r}_{in}\end{bmatrix} \tag{5}$$

The obtained map is the evaluated for $\delta\boldsymbol{r}_{in} = \boldsymbol{0}$ and $\Delta\boldsymbol{r}^n_{fin} = \Delta_\alpha\bar{\boldsymbol{r}}^n_{fin} = \alpha\Delta\bar{\boldsymbol{r}}^n_{fin}$ where α is the chosen step and $\Delta\bar{\boldsymbol{r}}^n_{fin}$ is the constant value of the Taylor polynomial expansion computed as the distance between $\boldsymbol{r}_{fin_{nk}}$ and $\boldsymbol{r}_{fin}$. Thus:

$$\begin{bmatrix}\delta\boldsymbol{r}_{in}\\ \delta\boldsymbol{v}^n_{in}\end{bmatrix} = \begin{bmatrix}\mathcal{M}_{\Delta\boldsymbol{r}^n_{fin}}\\ \mathcal{I}_{\boldsymbol{r}_{in}}\end{bmatrix}^{-1}\begin{bmatrix}\Delta_\alpha\bar{\boldsymbol{r}}^n_{fin}\\ \boldsymbol{0}\end{bmatrix} \tag{6}$$

By considering the second row and combining it with Equation 3:

$$\bar{\boldsymbol{v}}^{n+1}_{in} = \bar{\boldsymbol{v}}^n_{in} + \mathcal{M}_{\delta\boldsymbol{v}^n_{in}}(\Delta_\alpha\boldsymbol{r}^n_{fin},\ \boldsymbol{0}) \tag{7}$$

The optimization is finally stopped if the norm of the final position difference $\left|\Delta\tilde{\boldsymbol{r}}_{fin}^{n}\right|$ is smaller than a given tolerance ε or if a maximum imposed number of iteration n_{max} is reached.

A. Convergence Radius Estimation and Step Choice

In the previously exposed algorithm a function computing the required step is called on and, for the sake of clarity, this subsection aims to describe the algorithm used to evaluate the optimization step.
In order to find the convenient step, an estimation of the Taylor series convergence radius must be performed. In order to find an estimator, the following procedure has been developed based on the error estimation of a multi-variable polynomial found in [7]. The main idea is based on the estimation of the convergence radii for all the variables in the polynomial and then approximating the global convergence radius with the smallest of all these values by considering a safety factor to avoid overestimation due to approximations. Let $P(\boldsymbol{x})$ be a multi-variable polynomial that is factorized for each variable x_j $\forall j \in [1,\, n]$ as follows:

$$P(\boldsymbol{x}) = \sum_{m=0}^{k} x_j^m \, Q_{j,\,m}(x_1,\, \ldots,\, x_{j-1},\, x_{j+1},\, \ldots,\, x_n) \tag{8}$$

where $Q_{j,\,m}$ is a polynomial not depending on the variable x_j. By denoting the coefficient $q_{j,\,m}$ of the polynomial $Q_{j,\,m}$, the norm of each polynomial $Q_{j,\,m}$ is the sum of the absolute value of all the coefficients:

$$S_{j,\,m} = \sum_{q_{j,\,m} \in Q_{j,\,m}} \left|q_{j,\,m}\right| \tag{9}$$

From the computation of $S_{j,\,m}$, a series $\left(S_{j,\,m}\right)_{m\in[1,\,k]}$ is generated.
Furthermore, as exposed in [7], the polynomial norm that has just been defined decays exponentially by increasing the number m because of the Taylor's Theorem. From this consideration the value of $S_{j,\,k+1}$ can be inferred by logarithmic regression.
Form this polynomial norm, the radius of convergence associated with the j^{th} variable and the m^{th} term of the series is gathered:

$$\varrho_j^m = \frac{1}{\sqrt[m]{S_{j,\,m}}} \tag{10}$$

Successively, the convergence radius associated with the j^{th} variable is estimated to be smallest value among the last two convergence radius estimations [16]:

$$\varrho_j = \min\left\{\varrho_j^k,\, \varrho_j^{k+1}\right\} \tag{11}$$

By computing the value of ϱ_j for all the j the Estimated Trust-Region (ETR) is deduced. The ETR is an ellipsoid centered in the polynomial center providing information about the thrust region in each direction.

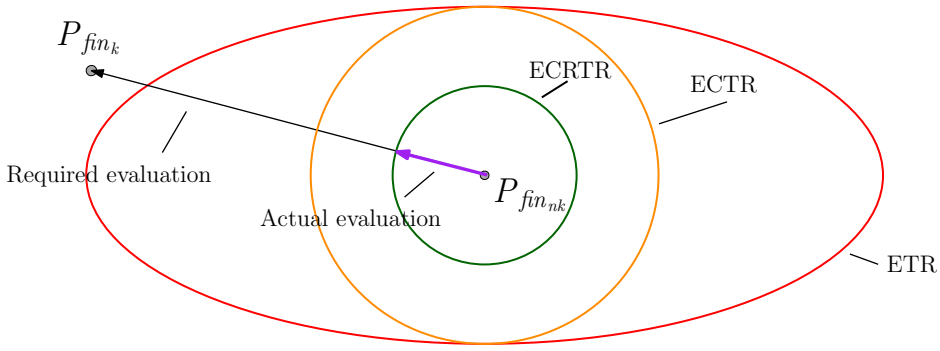


Figure 3 Sketch of the step choice in 2 dimensions.

The measure of the global convergence radius is computed by considering the minimal value among all the ϱ_j:

$$\varrho = \min_j\left\{\varrho_j\right\} \tag{12}$$

From this definition the Estimated Circular Trust-Region (ECTR) is inferred. The ECTR is the largest hypersphere contained in the ETR. Finally, in order to have a conservative estimation of the convergence radius, a safety factor η is added to contract

the ECTR - recommended value interval is $\eta \in [0.2, 0.6]$. This final hypersphere is labeled Estimated Circular Reduced Trust-Region (ECRTR). The step is then computed:

$$\alpha = \eta \frac{\varrho}{|\boldsymbol{x}_{eval}|} \tag{13}$$

where x_{eval} is the evaluation point. In Figure 3 a 2D example of the exposed procedure is shown.
A final remark must be made: the present algorithm is used only if the polynomial degree k is greater than 1 as the constant term of the series is not used for the logarithmic regression. In a linear case, the estimation is simply given by:

$$\varrho = \min_j \left\{ \varrho_j^1 \right\} \tag{14}$$

Moreover, in case of a Taylor polynomial vector of size v, the procedure must be repeated for all the polynomials and the global convergence radius is computed as the smallest value among the v Taylor models. Thus:

$$\begin{cases} \varrho = \min_{j,l} \left\{ \varrho_{j,l}^k,\ \varrho_{j,l}^{k+1} \right\} & \text{if } k > 1 \\ \varrho = \min_{j,l} \left\{ \varrho_{j,l}^1 \right\} & \text{otherwise} \end{cases} \tag{15}$$

where $\varrho_{j,l}^m$ is the estimated convergence radius associated with the l^{th} model, the j^{th} variable and the m^{th} coefficient of the factorization.

V. Optimization Initial Condition

In this section the initial condition for the PLP solver is expounded. Two type of initial conditions are considered:

1) The first optimization initial condition is the solution of the CLP between the initial position $\boldsymbol{r}_{in}$ and the final position $\boldsymbol{r}_{fin}$. It is computed with the Izzo's method [5] and labeled from now on $\boldsymbol{v}_{in_k}$.
2) The second optimization initial condition, labeled $\boldsymbol{v}'_{in_k}$, is the solution of the CLP between the initial condition $\boldsymbol{r}_{in}$ and the non-Keplerian final position at the first step $\boldsymbol{r}^1_{fin_{nk}}$. Both CLP solutions are computed with the Izzo's method.

The vector $\boldsymbol{v}'_{in_k}$ is considered as the default optimization initial condition and, only in case of not convergence, the first optimization initial condition is considered. In Figure 4 the difference between the two optimization initial conditions is shown.

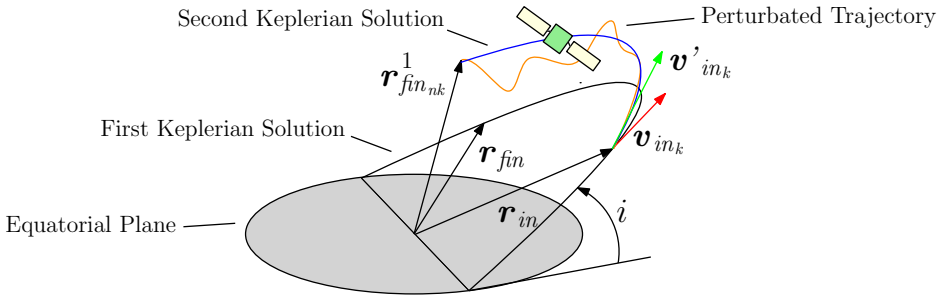


Figure 4 Difference between the two considered optimization initial conditions, i.e. $\boldsymbol{v}_{in_k}$ (in red) and $\boldsymbol{v}'_{in_k}$ (in green).

VI. Simulations

The expounded method has been tested on several orbits in order to understand its features and to circumscribe its validity. The performed section is organized as follows: firstly an overview of the chosen orbits to test is expounded and secondly numerical results are presented.

A. Test Cases

The considered test cases are listed in Table 2 where the orbit is defined by providing the classical orbital parameters. The chosen satellites cover a wide range of orbital conditions that can be cataloged in five main families:

1) Low Earth Orbit (LEO), mainly influenced by the atmospheric drag throughout their lifetime.
2) High Elliptical Orbit (HEO), perturbed principally by the Moon - not considered in this paper and considered as a future development - and secondly by the gravity potential perturbation close to the perigee.
3) Geosynchronous Transfer Orbit (GTO), altered by the drag force close to the perigee and by gravity potential because of their high eccentricity.

4) Medium Earth Orbit (MEO) and Geostationary Earth Orbit (GEO), weakly affected by gravity potential perturbations and solar radiation pressure - not considered in this work.

Satellite	a $[km]$	e $[-]$	i $[^\circ]$	Ω $[^\circ]$	ω $[^\circ]$	M $[^\circ]$	Orbit Class
E-Grasp	10495.1363	0.319671884	63.4	0.0	180.0	0.0	LEO
Proba 3	36943.0	0.8111	59.0	84.0	188.0	0.0	LEO
Molniya	26554.0	0.72	63.4	0.1	280.0	0.0	HEO
Tundra	42164.0	0.825	63.4	0.0	270.0	0.0	HEO
XMM-Newton	65648.3	0.816585	67.1338	0.0	270.0	0.0	HEO
Integral	87941.0	0.856052	54.0	0.0	0.0	0.0	HEO
SimbolX	106247.136454	0.75173	5.2789	89.341	-179.992	0.0	HEO
Standard GTO	24688.1363	0.716538	5.0	0.0	0.0	0.0	GTO
Super GTO	51528.1363	0.870398	30.0	0.0	0.0	0.0	GTO
SSTO	56640.6363	0.878124	45.3	0.0	0.0	0.0	GTO
ATV	6586.1775	0.00328	51.6	153.48	-21.395	215.24	LEO
CryoSat	7100.4651	0.00252	92.029	-37.185	107.492	51.202	LEO
EyeSat	7078.0	0.0	98.18	0.0	0.0	0.0	LEO
Jason 1	7254.0729	0.06216	66.974	-74.818	-241,05	179.7259	LEO
LEO	6831.5723	0.001357	51.6	224.8	280.1	66.5	LEO
Prisma FFIORD	13479.858	0.002	98.0	290.0	0.0	0.0	LEO
Proba 2	7106.137	0.000039	98.3	91.364	-1.423	180.0	LEO
SPOT-Like	7081.139	0.0158	98.0	164.02	0.0	0.0	LEO
Galileo	29995.22529	0.001040	56.0009	0.0	0.0	0.0	MEO
T2C	42830.90787	0.001956	5.6622	160.7292	249.2383	199.3432	GEO
Standard GEO	42170.0	0.001	0.00001	0.0	0.0	0.0	GEO

Table 2 Orbital parameters and orbit type of the considered test cases.

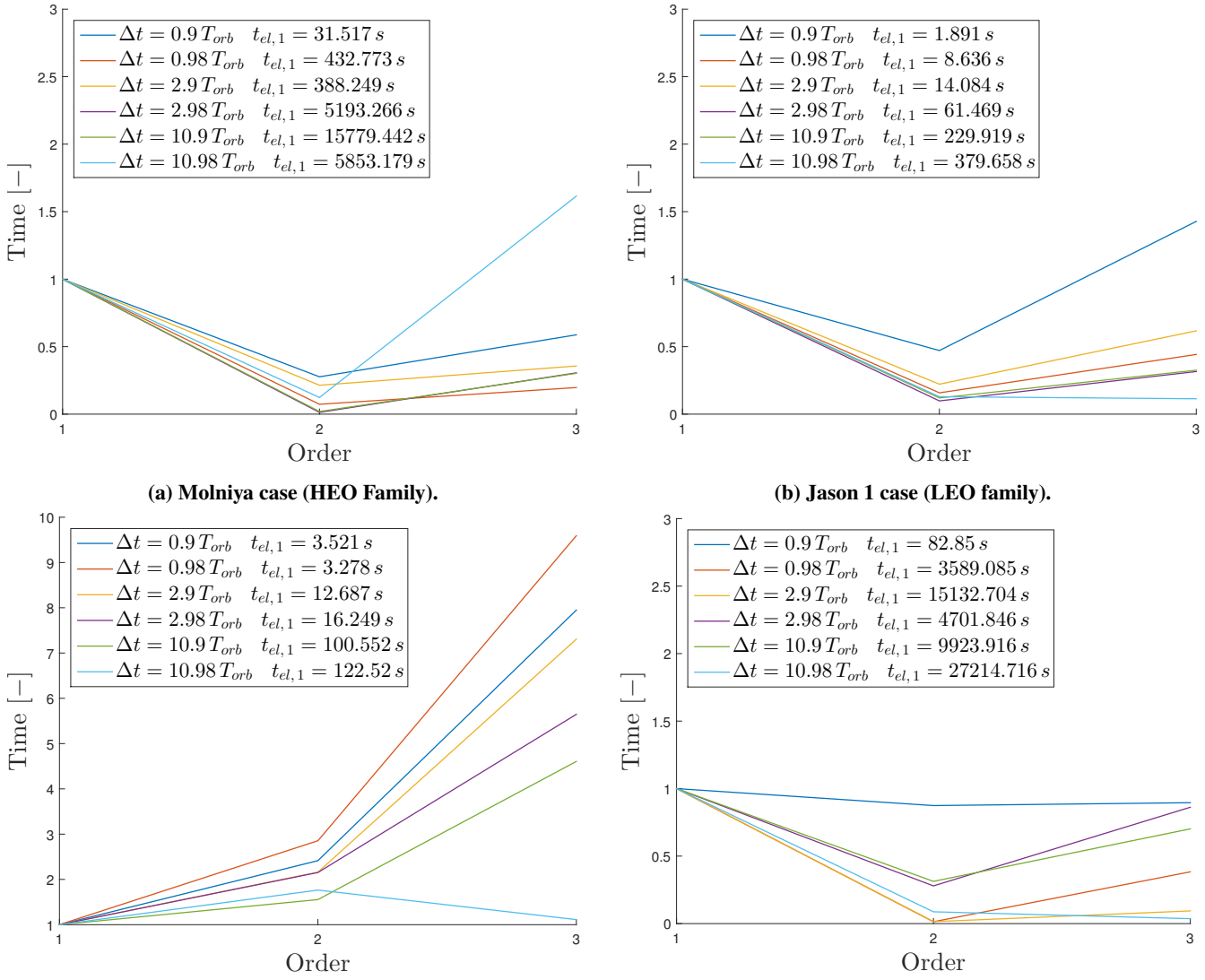
The initial conditions are propagated for a time span Δt whose expression is given by:

$$\Delta t = (\gamma + R)\ T_{\text{orb}} \tag{16}$$

where T_{orb} is the orbital period, $\gamma = \{0.9,\ 0.98\}$ is the orbit sweeping percentage and $R = \{0,\ 2,\ 10\}$ - considered as an input data of the algorithm - is the number of orbit revolutions. It is remarkable that in operational problems the value of R is not known and the algorithm must be run several times in order to compute the most reasonable value of R. On the one hand, the values of R have been chosen to study the optimization algorithm in case of short, medium and long propagation and, on the other hand, the selected values of γ have been designated to consider a well-conditioned and an ill-conditioned problem. The case $\gamma = 0.98$ is ill-conditioned as the initial and the final position are close to each other.
In the following subsection the results are presented where the PLP solver's constants have been chosen as follows: n_{max} has been chosen in order to achieve convergence, $\eta = 0.5$, $dt = 30\,s$ and $\varepsilon = 1\,m$ where ε is the stopping absolute tolerance, η is the step-size contraction factor, dt is the integration step and n_{max} is the maximal iterations number. The chosen absolute tolerance ε is extremely low if compared with operational problems but it has been chosen to test the algorithm robustness.

B. Numerical results

In the present section the batch of simulations is presented. All the simulations have been performed on an Intel® Core™ i3-2120 @3.30 GHz with 2 physical cores and 4 logic ones running on Microsoft Windows 7 Enterprise.
Firstly, a preference Taylor expansion order is deduced by looking at computational times. In Figures 5a - 5d the normalized time with respect to order 1 elapsed time is shown and it is clear that order 2 is the more convenient to be exploited.
As a consequence, the reported simulations consider polynomials of order 1 and 2. Therefore, the performance parameter τ_{PLP} is defined as the rapport between the order 2 computational time $t_{\text{el},\,2}$ and the order 1 one $t_{\text{el},\,1}$:



(a) Molniya case (HEO Family).

(b) Jason 1 case (LEO family).

(c) Standard GEO case (GEO family).

(d) Super GTO case (GTO family).

Figure 5 Comparison of elapsed time with respect to polynomial order for different orders for an archetype of the four main orbit families. Computing time is normalized with respect to the order 1 value (reported in the legend).

$$\tau_{\text{PLP}} = \frac{t_{\text{el, 2}}}{t_{\text{el, 1}}} \tag{17}$$

The complete batch of results is presented in Tables 4 and 5 where elapsed time and number of performed iterations are shown. By looking at the results, it is deducible that in the well-conditioned case convergence is achieved both with order 1 and 2 even if the order 2 has an non-negligible improvement as far as the performance index is concerned.

Furthermore it is remarkable that the initial condition $\boldsymbol{v}_{in}^{0}$ plays a central role as, in most ill-conditioned cases, the convergence is achieved just with a first Keplerian initial condition $\boldsymbol{v}_{in_k}$. In some cases - like SSTO with $\Delta t = 0.98\,T_{\text{orb}}$ - the problem is given by the fact that the second Keplerian initial condition $\boldsymbol{v}'_{in_k}$ is not close to the true solution. In other cases - such as Super GTO with $\Delta t = 0.98\,T_{\text{orb}}$ - both initials conditions do not lead to convergence as the problem has multiple solutions and the $\boldsymbol{v}_{in}^{n}$ jumps in the optimized state space until divergence is obtained. Finally, the non-uniqueness of the solution is clear in some cases as the optimization converges to a different velocity from the one used to generate the final position $\boldsymbol{r}_{fin}$. In these cases - like EyeSat with $\Delta t = 10.98\,T_{\text{orb}}$ - the use of a order 2 polynomial allows the convergence to the targeted solutions because more information about dynamical model derivatives is available.

In Table 3 the numerical values of parameter τ_{PLP} are listed. It can be seen that the higher the number of revolutions is the more convenient it is to add information with a second-order polynomial. An exception to this scenario is the GEO family where the order 1 is always convenient. This is due to the low non-linearity of the perturbation affecting the orbital motion.

	$0.9\,T_{\text{orb}}$	$0.98\,T_{\text{orb}}$	$2.9\,T_{\text{orb}}$	$2.98\,T_{\text{orb}}$	$10.9\,T_{\text{orb}}$	$10.98\,T_{\text{orb}}$
E-Grasp	0,374	0,149	0,215	0,062	0,065	– [1,2]
Proba 3	0,081	0,094	0,053	0,075	0,040	0,047
Molniya	0,260	0,064	0,154	– [2,3]	0,022	0,105
Tundra	0,972	0,026	0,275	– [1]	– [1,2]	0,315
XMM-Newton	1,836	0,189	1,029	0,039	0,115	– [2]
Integral	2,188	0,204	1,192	0,601	0,115	– [2]
SimbolX	2,357	0,511	1,548	0,277	2,075	0,227
Standard GTO	0,846	0,112	0,311	– [1,2]	0,040	0,361
Super GTO	0,894	– [1]	0,019	0,177	0,283	0,075
SSTO	0,932	0,024	0,007	– [4]	– [2]	0,113
ATV	0,350	0,119	0,232	0,052	0,131	– [4]
CryoSat	1,052	0,645	0,473	0,296	0,201	0,552
EyeSat	0,603	0,627	0,357	0,589	0,208	– [3]
Jason 1	0,452	0,129	0,196	0,096	0,099	0,123
LEO	1,120	0,198	0,559	0,140	0,223	0,161
Prisma FFiord	1,245	0,458	0,469	0,339	0,205	0,364
Proba 2	0,238	0,272	0,135	0,198	0,187	0,148
SPOT Like	0,587	0,344	0,294	0,373	0,123	0,898
Galileo	2,286	1,674	1,889	1,124	1,168	0,728
T2C	2,271	1,550	1,787	1,097	1,372	0,585
Standard GEO	2,240	2,385	1,789	1,917	1,299	1,586

[1] Order 1 not converging
[2] Order 2 not converging
[3] Order 1 converging to a different solution
[4] Order 2 converging to a different solution

Order 2 is more convenient | Order 1 is more convenient | τ_{PLP} cannot be computed

Table 3 Time gain τ_{PLP} between order 2 and order 1.

VII. Conclusions and Future Perspectives

In this paper a shooting method based on DA thrust-region optimization has been developed, implemented and tested. Results are promising even though further studies are necessary.

Several remarks must be made:

- The first point to remark upon is the importance of the initial condition for obtaining convergence. Two types of optimization initial condition $\boldsymbol{v}_{in}^{0}$ have been proposed. From a convergence point of view, the first Keplerian solution $\boldsymbol{v}_{in_k}$ ensures convergence in most of the cases as the step-size is smaller at the beginning of the optimization and the algorithm does not diverge. Nevertheless, the second Keplerian solution $\boldsymbol{v}'_{in_k}$ assures a faster convergence. Therefore, an accurate choice of the optimization initial condition is advised for highly non-linear orbit families, such as HEO and GTO.
- The preferred polynomial order is the order 2 as it is a satisfying compromise between accuracy of the polynomial expansion and the optimization time. A more detailed study can be performed by looking at each orbit family. On the one hand, an order 1 is generally faster for MEO and GEO orbits as the non-linearity is weakly affecting the orbit motion. On the other hand, an order 2 assures a concrete and non-negligible decrease of computing time in all the other cases.
- By looking at previously-expounded results, it seems clear that the present method is useful for solving the PLP for the LEO family as the computational time is lower than 10 minutes with all propagation time Δt. HEO orbits require more time as, when R and γ increase, each iteration needs a considerable time to compute the final position by performing the complete propagation. Moreover, as the absolute tolerance is low, convergence is slower than the one expected in operational cases.

Future studies are needed to improve the PLP solver performances. In order to enlarge the radius of convergence of the Taylor polynomial expansion automatic domain splitting [17] could be used even though the error thresholds and the number of

splitting would be user-fixed. Moreover, in order to avoid long term propagation, a DA evaluation map could be performed as in Reference 18. With this approach, long propagation can be reported to a single revolution problem even if it must be understood when the validity of the Taylor approximation ends. Furthermore, a regularization parameter λ should be added to ensure convergence to an initial velocity closer to the solver initial condition. An example could be:

$$J = \left|\Delta\tilde{\boldsymbol{r}}^n_{fin}\right| + \lambda\left|\tilde{\boldsymbol{v}}^n_{in} - \tilde{\boldsymbol{v}}^0_{in}\right| \tag{18}$$

where J is the criteria to be optimized. Finally, the third body perturbation should be implemented in order to have a complete overview of the possible perturbing forces.

Acknowledgment

I would like to convey special gratitude to my supervisor Vincent Morand for giving me the rewarding opportunity to carry out this internship at CNES as well as for his expert guidance and support.

References

[1] Bate, R. R., Mueller, D. D., and White, J. E., *Fundamentals of astrodynamics*, Courier Corporation, 1971.

[2] Battin, R. H., *An introduction to the mathematics and methods of astrodynamics*, AIAA, 1999.

[3] Vallado, D. A., *Fundamentals of astrodynamics and applications*, Vol. 12, Springer Science & Business Media, 2001.

[4] Pirovano, L., Santeramo, D. A., Wittig, A., Armellin, R., and Di Lizia, P., "Initial orbit determination based on propagation of orbit set with Differential Algebra," 2017.

[5] Izzo, D., "Revisiting Lambert's problem," *Celestial Mechanics and Dynamical Astronomy*, Vol. 121, No. 1, January 2015, pp. 1–15.

[6] Berz, M., "Modern map methods for charged particle optics," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, Vol. 363, No. 1-2, 1995, pp. 100–104.

[7] Wittig, A., Armellin, R., Colombo, C., and Di Lizia, P., "Long-term orbital propagation through differential algebra transfer maps and averaging semi-analytical approaches," *Advances in the Astronautical Sciences Series, Volume 152: SpaceFlight Mechanics 2014*, Vol. 152, No. 4, 2014, pp. 339–357.

[8] Valli, M., Armellin, R., Di Lizia, P., and Lavagna, M., "Nonlinear mapping of uncertainty in celestial mechanics," *Journal of Guidance, Control and Dynamics*, Vol. 36, No. 1, 2012, pp. 48–63.

[9] Rasotto, M., Morselli, A., Wittig, A., Massari, M., Di Lizia, P., Armellin, R., Valles, C., and Ortega, G., "Differential algebra space toolbox for nonlinear uncertainty propagation in space dynamics," 2016.

[10] Armellin, R., Di Lizia, P., Bernelli-Zazzera, F., and Berz, M., "Asteroid close encounters characterization using differential algebra: the case of Apophis," *Celestial Mechanics and Dynamical Astronomy*, Vol. 107, No. 4, 2010, pp. 451–470.

[11] Di Lizia, P., Armellin, R., and Lavagna, M., "Application of high order expansions of two-point boundary value problems to astrodynamics," *Celestial Mechanics and Dynamical Astronomy*, Vol. 102, No. 4, 2008, pp. 355–375.

[12] Di Lizia, P., Massari, M., and Cavenago, F., "Assessment of onboard DA state estimation for spacecraft relative navigation," 2017.

[13] Curtis, H. D., *Orbital mechanics for engineering students*, Butterworth-Heinemann, 2013.

[14] Hofmann-Wellenhof, B., Lichtenegger, H., and Wasle, E., *GNSS–global navigation satellite systems: GPS, GLONASS, Galileo and more*, Springer Science & Business Media, 2007.

[15] Drozyner, A., "An algorithm for recurrent calculation of gravitational acceleration," *Artificial Satellites*, Vol. 12, 1977, pp. 33–39.

[16] Jorba, À. and Zou, M., "A software package for the numerical integration of ODEs by means of high-order Taylor methods," *Experimental Mathematics*, Vol. 14, No. 1, 2005, pp. 99–117.

[17] Wittig, A., Di Lizia, P., Armellin, R., Makino, K., Bernelli-Zazzera, F., and Berz, M., "Propagation of large uncertainty sets in orbital dynamics by Automatic Domain Splitting," *Celestial Mechanics and Dynamical Astronomy*, Vol. 122, No. 3, 2015, pp. 239–261.

[18] Wittig, A. and Armellin, R., "High order transfer maps for perturbed Keplerian motion," *Celestial Mechanics and Dynamical Astronomy*, Vol. 122, No. 4, 2015, pp. 333–358.

	$0.9\,T_{\text{orb}}$		$0.98\,T_{\text{orb}}$		$2.9\,T_{\text{orb}}$		$2.98\,T_{\text{orb}}$		$10.9\,T_{\text{orb}}$		$10.98\,T_{\text{orb}}$	
	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$
E-Grasp	9,837	42	46,481	197	97,192	138	742,558	1151	1999,776	749	– [3]	– [3]
Proba 3	303,381	216	741,5	489	2553,436	566	3882,731	938	26880,885	1712	31053,409	1957
Molniya	52,044	59	743,785	792	666,878	242	33551,874 [2]	12429 [2]	20449,479 [1]	2047 [1]	10352,277 [1]	1074 [1]
Tundra	56,458	33	5493,018	3063	1694,393	315	– [3]	– [3]	– [3]	– [3]	4238,418	222
XMM-Newton	30,819	9	968,207	291	303,421	30	15979,405	1580	27379,839	756	62526,606	1723
Integral	46,934	9	1641,489	311	459,499	30	21784,91	1424	55789,639	1020	58101,108	1047
SimbolX	29,411	4	179,635	26	129,144	6	1625,049	80	1240,723	18	29384,94	399
STD GTO	31,736	42	663,425	855	590,342	252	– [3]	– [3]	29749,384 [1]	3330 [1]	7593,837	883
Super GTO	126,51	56	– [3]	– [3]	2108,548 [1]	286 [1]	8264,914	1123	16629,764	663	46826,47	1853
SSTO	131,825	51	7231,724 [1]	2519 [1]	39706,251	5212	8950,043	1064	15958,925	553	33131,218	1144
ATV	4,355	39	25,859	221	38,437	121	1509,539	4384	1289,493	984	5452,162	4544
CryoSat	0,999	8	2,167	16	7,791	22	20,675	55	110,021	76	445,539 [1]	334 [1]
EyeSat	2,124	17	3,275	25	17,488	50	33,207	89	298,709	207	417,074 [2]	313 [2]
Jason 1	3,076	24	16,393	123	24,038	66	107,935	279	423,2	282	672,303	489
LEO	1,093	9	8,405	68	7,791	23	61,646	171	141,735	102	458,969	363
Prisma FFiord	2,654	8	7,741	23	20,275	22	53,721	55	316,844	84	326,908	95
Proba 2	7,948	66	9,931	77	67,16	192	94,9	254	794,941	542	1282,097	971
SPOT Like	1,609	13	5,063	39	12,413	35	51,301	138	200,782	140	997,72 [1]	749 [1]
Galileo	3,527	3	7,055	6	16,62	5	43,255	13	126,65	10	444,15	39
T2C	5,994	3	10,091	5	22,78	4	51,073	9	146,01	7	429,686	22
STD GEO	5,604	3	6,024	3	21,361	4	27,465	5	174,122	9	205,233	11

[1] Converging only with the first optimization initial condition $\boldsymbol{v}_{in_k}$
[2] Converging to a different solution to the one it is expected
[3] Not converging with both optimization initial conditions

Table 4 Computation times and iterations in a order 1 case for the previously-exposed test cases.

	$0.9\,T_{\text{orb}}$		$0.98\,T_{\text{orb}}$		$2.9\,T_{\text{orb}}$		$2.98\,T_{\text{orb}}$		$10.9\,T_{\text{orb}}$		$10.98\,T_{\text{orb}}$	
	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$	Time $[s]$	Iterations $[-]$
E-Grasp	3,683	6	6,911	10	20,893	12	46,242	25	130,092	21	– [3]	– [3]
Proba 3	24,534	7	69,86	18	134,955	12	290,7	26	1086,997	29	1446,577	38
Molniya	13,52	6	47,529	19	102,947	15	– [3]	– [3]	444,716	19	1086,037	46
Tundra	54,861	13	140,546	27	466,746	36	– [3]	– [3]	15375,942	337	1336,033	29
XMM-Newton	56,597	7	183,359	21	312,155	13	620,297	24	3138,816	36	– [3]	– [3]
Integral	102,68	8	334,825	26	547,547	15	13086,253	346	6427,815	48	– [3]	– [3]
SimbolX	69,331	4	91,872	5	199,89	4	449,973	9	2574,034	7	6660,268	19
Standard GTO	26,846	15	74,421	36	183,885	33	– [3]	– [3]	1179,132	57	2743,875	130
Super GTO	113,154	20	72,358	12	322,312	19	1459,454	83	4703,298 [1]	75 [1]	3502,926	57
SSTO	122,919	19	175,263	25	271,758	14	59964,938 [2]	2960 [2]	– [3]	– [3]	3735,476	52
ATV	1,523	5	3,075	10	8,928	11	78,859	88	168,288	51	1108,452 [2]	330 [2]
CryoSat	1,051	3	1,397	4	3,684	4	6,118	6	22,127	6	246,148	67
EyeSat	1,281	4	2,053	6	6,243	7	19,56	19	62,028	17	356,149 [1]	72 [1]
Jason 1	1,39	4	2,113	6	4,713	5	10,383	10	42,038	11	82,983	21
LEO	1,224	4	1,663	5	4,355	5	8,631	9	31,614	9	73,944	21
Prisma FFiord	3,303	4	3,549	4	9,519	4	18,211	7	64,828	7	119,013	13
Proba 2	1,889	6	2,701	8	9,068	10	18,824	19	148,982	43	190,015	53
SPOT Like	0,944	3	1,744	5	3,653	4	19,151	20	24,765	7	896,321	248
Galileo	8,062	3	11,808	4	31,389	4	48,601	6	147,911	5	323,313	11
T2C	13,613	3	15,646	3	40,718	3	56,025	4	200,345	4	251,281	5
Standard GEO	12,552	3	14,365	3	38,222	3	52,655	4	226,194	5	325,556	7

[1] Converging only with the first optimization initial condition $\boldsymbol{v}_{in_k}$
[2] Converging to a different solution to the one it is expected
[3] Not converging with both optimization initial conditions

Table 5 Computation times and iterations in a order 2 case for the previously-exposed test cases.