INITIAL ORBIT DETERMINATION BASED ON PROPAGATION OF ORBIT SETS WITH DIFFERENTIAL ALGEBRA

L. Pirovano[(1)], D. A. Santeramo[(2)], A. Wittig[(3)], R. Armellin[(4)], and P. Di Lizia[(5)]

[(1)]*Dept. de Matemáticas y Computación, Universidad de La Rioja, Spain, laura.pirovano@unirioja.es*
[(2)]*Dept. of Aerospace Science and Technology, Politecnico di Milano, Italy, danieleantonio.santeramo@polimi.it*
[(3)]*European Space Agency (ESA/ESTEC), Netherlands , alexander.wittig@esa.int*
[(4)]*Surrey Space Center - University of Surrey, United Kingdom, r.armellin@surrey.ac.uk*
[(5)]*Dept. of Aerospace Science and Technology, Politecnico di Milano, Italy, pierluigi.dilizia@polimi.it*

ABSTRACT

The object of this paper is the initial orbit determination of an orbit set compatible with an observational arc by means of differential algebra. The initial estimate is retrieved as a truncated power series expanded with respect to the uncertainties in the measurements. The analysis of the region is performed with the automatic domain splitting, that splits the orbit set into two or more regions defined by just as many Taylor expansions when the estimated truncation error introduced by the truncated power series exceeds a certain tolerance. A comparison between the proposed initial orbit determination approach and alternative methods from the literature is included to show the improvements achieved by exploiting accuracy information using differential algebra. The goal of the description of the initial orbit determination solution as orbit set is to propagate several independent orbit set to a common epoch and analyze them to decide whether they're correlated or not. Initial results for the linkage of two independent observations are also included.

Key words: Initial orbit determination, differential algebra, admissible region.

1. INTRODUCTION

Every day thousands of detections of objects orbiting the Earth are retrieved by observatories. However, a single observational arc is in general not sufficient to accurately determine the orbit of an unknown object. Due to the observation geometry and the uncertainty related to sensor accuracy, timing accuracy, and observer state knowledge, we obtain a relatively large orbit set (OS) that is compatible with one observational arc rather than one single orbital state. This paper outlines a new method, called the Differential Algebraic Initial Orbit Determination (DAIOD), which aims to describe the solution of the initial orbit determination (IOD) as a truncated power series (TPS) that depends on variations of the observations,the so called OS. The reason to do so is to have the solution in a suitable form to then propagate it. Indeed, by propagating different OSs at the same epoch, it is possible to analyze the region they define: by analyzing the overlapping of two OSs, one can decide if they're correlated and if so, the orbit estimate can be refined.

Since for the present work the observations are simulated, the output of a typical real optical observation is first explained here after to then describe the simulating tool, called the virtual observatory (VO). The relevant nomenclature from literature is then outlined to introduce the DAIOD method.

A typical optical observation is characterized of four important parameter:

- t The time at which the observation is made. In a geostationary Earth orbit (GEO), the time between two subsequent observations is typically around $2{-}3\,\mathrm{min}$;
- $\hat{\alpha}$ The longitude of the observed object with respect to the observatory;
- $\hat{\delta}$ The latitude of the observed object with respect to the observatory;
- σ_{P} The precision of the observation (usually in arcseconds), defined as the 3σ variation of the observed angle. Typical values for the precision vary from $1''$, being the most accurate observation, to $5{-}10''$, being the least accurate.

The observation noise is assumed to be normally distributed, such that the observed angles can be represented as Gaussian variables:

$$\begin{cases} \alpha_{\mathrm{i}} \sim \mathcal{N}(\hat{\alpha}_{\mathrm{i}}, \sigma^2) \\ \delta_{\mathrm{i}} \sim \mathcal{N}(\hat{\delta}_{\mathrm{i}}, \sigma^2) \end{cases} \tag{1}$$

where

$$\sigma_i = \frac{\sigma_{\mathrm{P,i}}}{3} \tag{2}$$

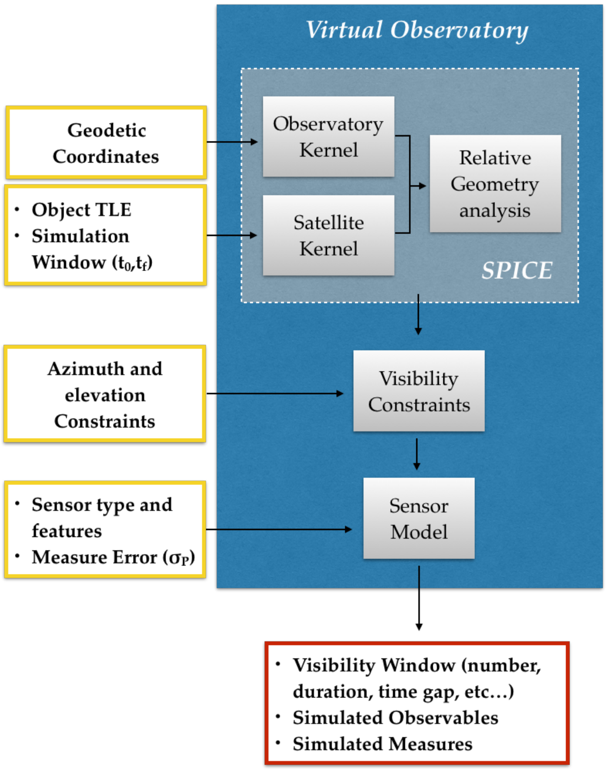


Figure 1. VO software tool description.

During a full passage above a station, typically $5-8$ observations are made for the same object, when the sensor is fixed in a certain position. For the work at hand, the optical observations are simulated through the VO. This software is able to recreate optical and radar survey scenarios from any point on the Earth, thanks to the use of Spacecraft Planet Instrument C-matrix Events (SPICE), a powerful tool developed by the Navigation and Ancillary Information Facility (NAIF) group at NASA. Figure 1 shows the inputs for the VO in the yellow boxes, while the output is given in the red box. After receiving the two-line elements (TLEs) of the satellite and the observatory coordinates, the SPK kernels of these two objects are created: these files contain data that represent the ephemerides of the objects for a certain time interval, which means that the states of the satellite and of the observatory can be retrieved at any time within the user defined simulation window. From here, the relative geometry between the satellite and the observatory can be analyzed, by taking into account the sky background luminosity, the object illumination and the object elevation to define the observability windows. Lastly, the observations are simulated by defining the sensor type and by adding user-defined measurement noises, that is the σ_{P}. A file containing the observations made during the visibility window is then created.

Now, the relevant nomenclature that will be used throughout the paper is outlined.

Observation. An observation is made of a time t, two angles α and δ and the associated precision σ_P.

Very short arc (VSA). It is a sequence of N observations where an object is found to move [9, 10]. Due to the short interval between each detection, these observations do not allow for the definition of a track, but still contain useful information about the object.

Observations set. It is the set of three observations used to perform IOD, generally the first, middle and last of a VSA.

Attributable. Reference [10] defines an attributable as the useful information that could be extracted from a VSA. It is made of two angles and two angular rates for an optical observation:

$$\mathcal{A} = \left(\alpha, \delta, \dot{\alpha}, \dot{\delta}\right) \tag{3}$$

where generally the two angles coincide with the middle observation and the angular rates are calculated with the remaining data from the observation set. This definition is now widely used to describe the $4\,D$ vector containing partial information about the object observed, when considering an error-free observation.

$(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain. It is the $6D$ region containing the $\boldsymbol{\sigma}_{\mathrm{P,i}}$ uncertainties in the angles.

Orbit set (OS). Given the state of the object as a function of $\delta\boldsymbol{\alpha}$ and $\delta\boldsymbol{\delta}$, the OS is the set of the orbital states $(\boldsymbol{r}_2, \boldsymbol{v}_2)$ enclosed by the range of the state function over a given $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain. Once the OS is defined, it is possible to retrieve other values as a function of the state, thus also having them expanded with respect to the same $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain. Functions that will be used later are, for example, the range and the range-rate:

$$\begin{aligned}
\rho(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta}) &= \|\boldsymbol{r}_2(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta}) - \boldsymbol{R}_2\| \\
\dot{\rho}(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta}) &= \frac{(\boldsymbol{r}_2(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta}) - \boldsymbol{R}_2) \cdot (\boldsymbol{v}_2(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta}) - \boldsymbol{V}_2)}{\rho(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta})}
\end{aligned} \tag{4}$$

where $\mathbf{R}_2$ and $\mathbf{V}_2$ are respectively the position and velocity of the observer.

Admissible region (AR). It is the 2D plane generated by the two degrees of freedom of the *Attributable*, that is the $(\rho, \dot{\rho})$ plane, where ρ is the range and $\dot{\rho}$ is the range-rate. It is the region where the attributable places the information, respecting physical constraints such as the energy law and minimum/maximum distance from the Earth. Alternatively, it can be described as the range of the $\rho(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta})$ and $\dot{\rho}(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta})$ functions, given the initial $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain, thus being the projection of the OS on the $(\rho, \dot{\rho})$ plane. It is, however, important to underline the difference of these two definitions: the first one is purely deterministic, and every point of the region has equal probability of being the real solution; the second definition takes into account the uncertainties $\sigma_{\mathrm{P,i}}$ in the observations and thus defines the region as a point solution together with its neighboring area defined by the variations in the observations.

Linkage problem. It is the problem of identifying two or more VSAs belonging to the same physical object: the VSA alone cannot define an orbit, but if two or more are found to be correlated, the orbit can be computed.

Having defined the keywords, the paper will now describe the IOD algorithm in Section 2, where the OS is generated starting from an observation set. Then, Section 3 introduces the automatic domain splitting (ADS) algorithm necessary to sample the OS in order to keep the error introduced by the TPS below a certain threshold for the entire (α, δ)-domain. Results are shown in Section 4, where comparison with literature and validation are shown, while Section 5 concludes the paper.

2. INITIAL ORBIT DETERMINATION

Orbit determination (OD) refers to the use of a set of techniques for estimating the orbits of objects and is typically divided into two phases. When the number of observations is equal to the number of unknowns, a nonlinear system of equations needs to be solved. This problem is known as IOD. When many more observations are taken over an orbit arc of adequate length, accurate orbit determination (AOD) can be performed [3]. This paper deals with an IOD algorithm that exploits differential algebra (DA). Since the closer the observations are in time, the more difficult it is to perform IOD [10], given a full passage over the station, the first, middle and last observations are considered. Before explaining the IOD algorithm, the DA framework and some useful algorithms are introduced.

2.1. The DA framework

DA supplies the tools to compute the derivatives of functions within a computer environment. More specifically, by substituting the classical implementation of real algebra with the implementation of a new algebra of Taylor polynomials, any function f of v variables is expanded into its Taylor polynomial up to an arbitrary order n with limited computational effort. This means that every time a generic function is evaluated in a point x_0, the output will be the TPS around x_0. In addition to basic algebraic operations, operations for differentiation and integration can be easily introduced in the algebra, thus finalizing the definition of the differential algebra structure of DA [4, 5]. One important function enabled by DA that will be exploited in the algorithm is here described: the expansion of the solution of parametric implicit equations [1, 2]. Well-established numerical techniques (e.g., Newton's method) exist to compute numerically the solution of an implicit equation

$$\boldsymbol{f}(\boldsymbol{x}) = 0, \tag{5}$$

with $\boldsymbol{f} : \Re^n \to \Re^n$. Suppose an explicit dependence on a vector of parameters $\boldsymbol{p}$ can be highlighted in the vector function $\boldsymbol{f}$, which leads to the parametric implicit equation

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{p}) = 0. \tag{6}$$

We look for the function $\boldsymbol{x}(\boldsymbol{p})$ that solves (6) for any value of $\boldsymbol{p}$.

DA techniques can effectively handle the previous problem by representing $\boldsymbol{x}(\boldsymbol{p})$ in terms of its Taylor expansion with respect to $\boldsymbol{p}$. This result is achieved by applying partial inversion techniques as detailed in [7]. The final result is

$$[\boldsymbol{x}] = \boldsymbol{x} + \mathcal{T}_{\boldsymbol{x}|f(\boldsymbol{x})=0}\,(\delta \boldsymbol{p}), \tag{7}$$

which is the k-th order Taylor expansion of the solution of the implicit equation. For every value of $\delta \boldsymbol{p}$, the approximate solution of $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{p}) = 0$ can be easily computed by evaluating the Taylor polynomial (7). The solution obtained by means of the polynomial map (7) is a Taylor approximation of the exact solution of Eq. (6). The accuracy of the approximation depends on both the order of the Taylor expansion and the displacement $\delta \boldsymbol{p}$ from the reference value of the parameter.

In C++ the DA framework is implemented in the library DACE [12].

2.2. Gauss' and Lambert's algortihms

Gauss' Algorithm
Gauss' algorithm works in double precision and takes as input the times of observations $(t_1,\, t_2,\, t_3)$, the positions of the observatory at these times $(\boldsymbol{R}_1,\, \boldsymbol{R}_2,\, \boldsymbol{R}_3)$ and the direction cosine vectors $(\hat{\boldsymbol{\rho}}_1\, \hat{\boldsymbol{\rho}}_2\, \hat{\boldsymbol{\rho}}_3)$. The algorithm estimates the slant ranges $(\rho_1,\, \rho_2,\, \rho_3)$ in order to obtain the object positions

$$\boldsymbol{r}_{\mathrm{i}} = \boldsymbol{R}_{\mathrm{i}} + \rho_{\mathrm{i}}\, \hat{\boldsymbol{\rho}}_{\mathrm{i}}, \qquad \text{where } i = 1, 2, 3 \tag{8}$$

in two-body dynamics. Figure 2 shows the geometry of the problem. The description of this algorithm can be found in [6].

Lambert's Algorithm
Lambert's algorithm takes as input two position vectors, the Δt between them and the gravitational parameter and gives as output the velocity vectors. The geometry of the problem is shown in Figure 3. The algorithm can work with both double precision and DA variables. Since it is not possible to solve the problem analytically, as outlined

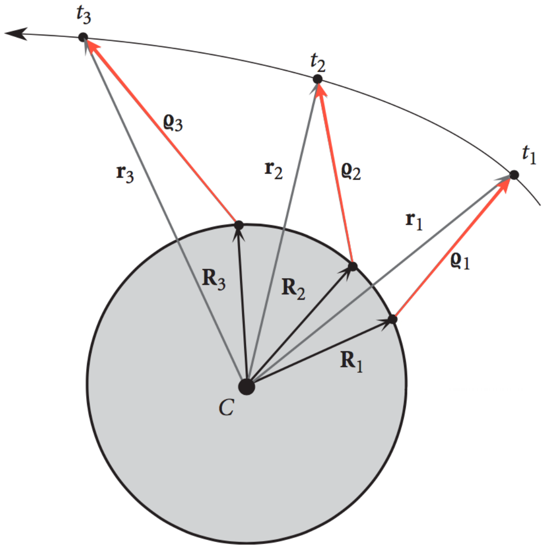


Figure 2. Geometry of input and output values for Gauss's algorithm. Modified from [6].

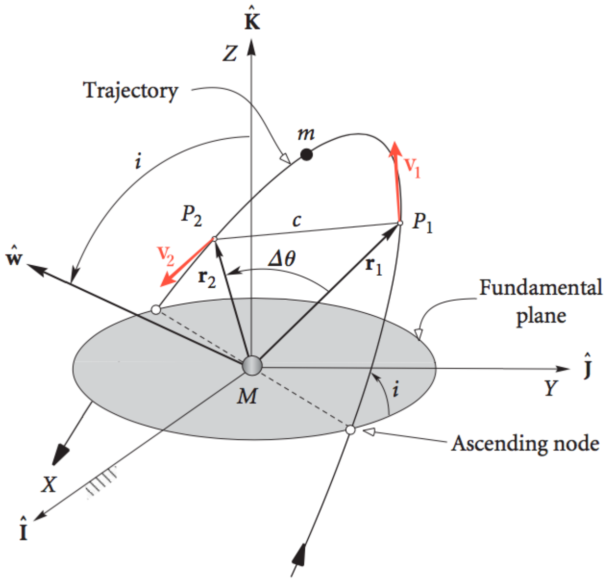


Figure 3. Geometry of input and output values for Lambert's algorithm. Modified from [6].

in [14], several methods were created in the past. For the work at hand, the C++ implementation in [8] has been used, after updating it to be able to accept both double precision and DA variables.

2.3. Mathematical definition of the DAIOD algorithm

The DAIOD algorithm takes as input the observation of an object as defined in Section 1 and gives as output the TPS of the object state at the central time of the observation. To do so, an initial estimate of the object position at t_1, t_2 and t_3 is obtained through Gauss' algorithm in double precision. The observatory state is retrieved through the kernel obtained with the VO, while the direction cosines are found through the observation angles:

$$\hat{\boldsymbol{\rho}}_{\mathrm{i}} = \begin{bmatrix} \cos\delta_{\mathrm{i}}\cos\alpha_{\mathrm{i}} \\ \cos\delta_{\mathrm{i}}\sin\alpha_{\mathrm{i}} \\ \sin\delta_{\mathrm{i}} \end{bmatrix} \tag{9}$$

where $i = 1,\, 2,\, 3$ refers to the observation instances.

At this point an estimate for the position vectors $\boldsymbol{r}_1$, $\boldsymbol{r}_2$ and $\boldsymbol{r}_3$ is available in double precision. Now, the velocities have to be computed. Lambert's algorithm takes as input two position vectors and the Δt between them and gives as output the velocity vectors. This means that by computing Lambert's algorithm twice (from t_1 to t_2 and from t_2 to t_3) one should be able to retrieve the three state vectors. However, Gauss' algorithm does not ensure that the three estimated vectors define one unique orbit, thus it does not ensure that the two velocity vectors found at t_2 ($\boldsymbol{v}_2^-$ and $\boldsymbol{v}_2^+$) coincide, as can be seen in Figure 4. To fix this problem and obtain the expansion of the state, Lambert's algorithm will be used twice: the first time to find the $\delta\boldsymbol{\rho} = (\delta\rho_1,\, \delta\rho_2,\, \delta\rho_3)$ necessary to ensure that $\boldsymbol{v}_2^- = \boldsymbol{v}_2^+$ and the second time to expand the corrected solution $(\boldsymbol{r}_2,\, \boldsymbol{v}_2)$ with respect to the observation angle variations $\delta\boldsymbol{\alpha}$, $\delta\boldsymbol{\delta}$. The first step allows us to improve the estimation of the ranges made in Gauss's algorithm by forcing the three estimates to be part of the same orbit, while the last step allows us to analyze the variations in the state vectors due to variations in the observations just by means of function evaluations. Indeed, although the observations are our known values, they are not free of errors, hence it is important to analyze the neighborhood of the solution.

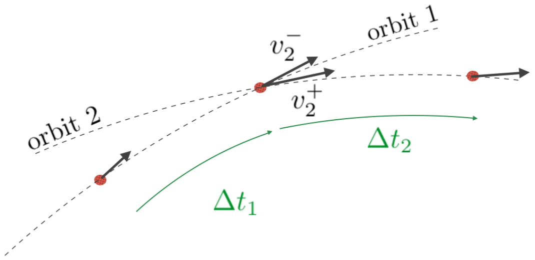


Figure 4. Output of Lambert algorithm taking as input the output from Gauss's algorithm

For the first usage of Lambert's algorithm, the values ρ_1, ρ_2, ρ_3 are initialized as DA variables. Equation 10 shows the mathematical definition. The curly brackets show the computer adapted definition of the variation δ.

$$\begin{cases} \rho_{1,DA} = \rho_{1,Gauss} + \delta\rho_1\ \{\mathrm{DA}(1)\} \\ \rho_{2,DA} = \rho_{2,Gauss} + \delta\rho_2\ \{\mathrm{DA}(2)\} \\ \rho_{3,DA} = \rho_{3,Gauss} + \delta\rho_3\ \{\mathrm{DA}(3)\} \end{cases} \tag{10}$$

In this way, the output of Lambert's algorithm are the velocity functions depending on variations of the slant ranges. In particular:

$$\boldsymbol{v}_2^- = \boldsymbol{v}_2^- \left(\delta\rho_1,\, \delta\rho_2\right) \tag{11}$$

$$\boldsymbol{v}_2^+ = \boldsymbol{v}_2^+ \left(\delta\rho_2,\, \delta\rho_3\right) \tag{12}$$

With the goal of solving the discontinuity in t_2, the $\Delta\boldsymbol{v}$ between the left and right velocities is calculated:

$$\Delta\boldsymbol{v} = \boldsymbol{v}_2^+ - \boldsymbol{v}_2^- = \Delta\boldsymbol{v}\left(\delta\rho_1,\, \delta\rho_2,\, \delta\rho_3\right) \tag{13}$$

By forcing $\Delta\boldsymbol{v} = \mathbf{0}$ one wants to find the $\delta\boldsymbol{\rho}$ necessary to obtain it. Newton method for DA explained in Subsection 2.1 is used here. Indeed, the function f is $\Delta\boldsymbol{v}$, the parameters $\boldsymbol{x}$ are ρ_i and the variations $\delta\boldsymbol{p}$ are $\delta\rho_i$. Thus, from

$$\Delta\boldsymbol{v}\left(\boldsymbol{\rho};\, \delta\boldsymbol{\rho}\right) = \mathbf{0} \tag{14}$$

one obtains:

$$\begin{cases} \rho_{1,L1} = \rho_{1,Gauss} + \Delta\rho_1 \\ \rho_{2,L1} = \rho_{2,Gauss} + \Delta\rho_2 \\ \rho_{3,L1} = \rho_{3,Gauss} + \Delta\rho_3 \end{cases} \tag{15}$$

such that

$$\Delta\boldsymbol{v}\left(\rho_{1,L1},\, \rho_{2,L1},\, \rho_{3,L1};\, 0,\, 0,\, 0\right) = \mathbf{0} \tag{16}$$

At the end of this step, one has obtained the object states that satisfy the constraint of pertaining to the same orbit. However, the solution is expanded with respect to the slant ranges. This is not useful, since one wants the solution in terms of the observations and not in terms of the output itself. For this reason, Lambert's algorithm is used again. This time, the DA variables initialized are the observations. The non-constant parts of the angle polynomials are scaled by the precision of the observation $\sigma_{P,i}$, so that by evaluating the final solution within the interval $[-1, 1]$, one can find the 3σ interval solution depending on the accuracy. Equation 17 shows the definition:

$$\begin{cases} \alpha_{1,DA} = \hat{\alpha}_1 + \sigma_{\text{P},1}\, \delta\alpha_1 & \{\sigma_{\text{P},1}\text{DA}(1)\} \\ \alpha_{2,DA} = \hat{\alpha}_2 + \sigma_{\text{P},2}\, \delta\alpha_2 & \{\sigma_{\text{P},2}\text{DA}(2)\} \\ \alpha_{3,DA} = \hat{\alpha}_3 + \sigma_{\text{P},3}\, \delta\alpha_3 & \{\sigma_{\text{P},3}\text{DA}(3)\} \\ \delta_{1,DA} = \hat{\delta}_1 + \sigma_{\text{P},1}\, \delta\delta_1 & \{\sigma_{\text{P},1}\text{DA}(4)\} \\ \delta_{2,DA} = \hat{\delta}_2 + \sigma_{\text{P},2}\, \delta\delta_2 & \{\sigma_{\text{P},2}\text{DA}(5)\} \\ \delta_{3,DA} = \hat{\delta}_3 + \sigma_{\text{P},3}\, \delta\delta_3 & \{\sigma_{\text{P},3}\text{DA}(6)\} \end{cases} \tag{17}$$

At this point $\boldsymbol{\rho}$ is the output of the first Lambert's algorithm in double precision: $\boldsymbol{\rho} = \boldsymbol{\rho}_{L1}$. The mathematical ground of this method is the first order Newton:

$$\Delta\boldsymbol{v}\left(\boldsymbol{\rho}\right) = \mathbf{0} \;\Rightarrow\; \boldsymbol{\rho}_{i+1} = \boldsymbol{\rho}_i - J^{-1}_{\Delta\mathbf{v}(\boldsymbol{\rho}_0)}\, \Delta\boldsymbol{v}(\boldsymbol{\rho}_i) \tag{18}$$

Here, the assumption is made that the Jacobian does not change in the loop. The iteration is carried out until i = MaxOrder, thus until the highest order of the DA variable is reached. Once the ranges are found depending on the observations, the position vectors are obtained with Equation 8, while the velocities can be calculated with one last Lambert's procedure.

An important outcome of this method is that one not only obtains the point solution, but can also easily calculate its 3σ variation by means of functions evaluations. Recalling the definition of the nomenclature in Section 1, the original domain of the variations is the $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain, while the range of this domain through the DAIOD algorithm is called the OS. The solution, however, is not valid for any variation of the input. While considering a power series one has to respect the radius of convergence, when dealing with TPSs one has to carefully estimate the truncation error: if this exceeds a certain tolerance, the approximation is not good enough. This implies that one single TPS may not able to describe the entire OS and the initial $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain may need to be split in sub-regions. With this division, there would be as many TPSs as sub-regions, keeping the error of the approximation below a certain threshold for the whole range of solutions. The tool that is able to estimate the error, divide the domain and find the new TPS for the OS is called the ADS tool and is introduced in Section 3.

3. AUTOMATIC DOMAIN SPLITTING

When using DA tools it is important to analyse the region of validity of a TPS since it defines the validity of a DA map. To solve this problem the idea is to create a procedure able to estimate the limits of validity of a TPS and split the domain linked to this TPS in two or more subdomains when these limits are exceeded. This means that when a single Taylor expansion is not enough to represent the DA map, the tool splits the domain on which the map is defined and evaluates a new map for each of the new domains.

The tool which automatically performs the estimation of the validity of the map is the automatic domain splitting (ADS). ADS generates a list of domains and respective TPSs, whose union corresponds to the initial DA set. In this way all the TPSs obtained lie on a region within their convergence disk and the approximation error can be controlled and kept below fixed threshold. This can be done when the DA map is created as well as when it is propagated in time. During the creation of the map the truncation error is estimated for the TPS that represents the transformation from an initial domain to a codomain at the same epoch, instead for the propagation the TPS that approximates the propagated state is taken into account. The efficiency of the tool for uncertainty propagation is highlighted in [17].

The mathematical concept that lies behind the representation of the obtained list of subsets is the *Manifold* as outlined in [15]. To better understand it, three main definitions are here outlined:

- A topological space $\mathcal{M}$ is locally Euclidean of dimension n if every point $p \in \mathcal{M}$ has a neighborhood $\mathcal{U}$ such that there is a homeomorphism φ from $\mathcal{U}$ onto an open subset of $\Re^n$.

- A *Chart* is the couple $(\mathcal{U}, \varphi)$, where $\mathcal{U} \subset \mathcal{M}^n$ is an open subset of the manifold and $\varphi : \mathcal{U} \mapsto \Re^n$ is a homomorphism of $\mathcal{U}$ into Cartesian space.
- An *Atlas* $\mathcal{A}$ is a collection of charts $(\mathcal{U}_\alpha, \varphi_\alpha)$ such that the union of $\mathcal{U}_\alpha$ form a open cover of the manifold, i.e. $\bigcup_{\alpha \in A} \mathcal{U}_\alpha = \mathcal{M}^n$.

Every manifold admits such an atlas and if all the φ_α are *r*-times differentiable we can refer to the atlas as a $\mathcal{C}^r$atlas. It is to be noted that the atlas is not unique, since different atlases can represent the same manifold and an atlas is in general composed of an infinite number of charts. However, manifolds of practical relevance are usually described by a finite atlas, thus becoming closed manifolds. The previous definitions can be associated to the process of mapping the globe: when projecting a part of the globe on a 2-D map, the part of the globe to be mapped can be identified with a subset of the manifold and once the projection function is defined, the globe section can be mapped on a 2-D Cartesian space. Thus, the section of the globe and its projection function constitute a Chart for the globe. Furthermore, a global projection function able to map the whole globe onto a single Cartesian space chart does not exist if the mapping is to be homeomorphic. More detailed properties and characteristics are presented in many math text book as [11] and [13].

3.1. DA Manifold Representation

Since the φ_α functions of the charts are bijective functions, it is also possible to define an atlas using the inverse maps, therefore the inverse charts can be defined as

$$\begin{cases} (V_\alpha, {\varphi_\alpha}^{-1})_{\alpha \in A} \\ {\varphi_\alpha}^{-1} : V_\alpha \subset \Re^k \mapsto {\varphi_\alpha}^{-1}(V_\alpha) \subset \mathcal{M}^n. \end{cases} \tag{19}$$

By exploiting the representation of an atlas of inverse charts, the concept of manifold can be easily applied to the DA set description: it is indeed much easier to specify a subset of the Euclidean space than to specify a subset of an arbitrary manifold.
Practically, this means that the DA manifold is composed of an atlas of charts made of DA vector that represent the Euclidean domain, while φ_α^{-1} is a TPS that maps the Euclidean space onto the manifold. Moreover, by choosing the Euclidean domain to be $(-1, 1)^v$ and following the exposition in [15], we introduced the DA manifold as follows:

> A *v*-dimensional DA manifold $\mathcal{M}$ embedded in a *w*-dimensional space is defined by its finite DA atlas $\mathcal{A}$ of DA charts $((-1, 1)^v, {\varphi_\alpha}^{-1})$ with ${\varphi_\alpha}^{-1} \in {}_wD_v$.

The employment of inverse maps and the scaled initial domain lead to another simplification: once the domain is fixed to $(-1, 1)^v$ it does not need to be stored in the DA chart thus saving memory.

3.2. The ADS within the DA Manifold

Thanks to previous definitions, the concept of DA manifold can be straight used for the OS determination and propagation. For the DA representation of the DA chart and the DA manifold, Taylor theory is used: the maps are Taylor expansions around a nominal point, that is a map representation of the initial state with its uncertainties, the orbit set (OS). For any map used for the representation, the truncation error has to be smaller than a fixed threshold to ensure that it well approximates the real state. The estimation and control of the error is provided by the ADS. To build the basis of the error control process, that will be exploited in Section 3.3, the maps will now be mathematically defined.

For the initial OS determination the map goes from the initial domain $U \in \Re^v$ to the DA manifold. At the beginning of the procedure, the atlas representing the manifold is defined by one chart:

$$\begin{cases} \mathcal{A} = \{(\mathcal{U}, \gamma)\} \\ \gamma : (-1, 1)^v \in \Re^v \mapsto \gamma((-1, 1)^v) \in \Re^n \end{cases} \tag{20}$$

The error of the map is then estimated. If this error is bigger than a fixed threshold, the ADS tool splits the initial domain $\mathcal{U}$ into two sub-domains along the direction of biggest error and re-evaluates a map over each of the new domains. The dual splitting is carried out until the criterion is met. In this way a new atlas that represents the initial manifold is obtained and can be written as:

$$\begin{cases} \mathcal{A} = \{(\mathcal{U}_\alpha, \gamma_\alpha)\}_{\alpha \in A} = \mathcal{M}^n \\ \bigcup_{\alpha \in A} \mathcal{U}_\alpha = \mathcal{U} \\ \gamma_\alpha : \mathcal{U}_\alpha \in \Re^v \mapsto \gamma_\alpha(\mathcal{U}_\alpha) \in \Re^n \end{cases} \tag{21}$$

3.3. Map Error Estimation Process

The most crucial problem with the DA representation and the Taylor expansion is the calculation of the error between the real function and the TPS that approximates it. The theory behind the error estimation function, given in [17] and developed within the ADS algorithm is now explained.

Let f be a $n + 1$ differentiable function $f \in C^{n+1}$ and P_f be its n-th order Taylor expansion with radius $r > 0$. For some $C > 0$ with e_r the maximum error of the expansion, if we halve the radius of the domain, then the error will decrease by a factor $\frac{1}{2^{n+1}}$

$$|f(\delta x) - P_f| \leq C \cdot \delta x^{n+1} \leq C \cdot \left(\frac{r}{2}\right)^{n+1} = \frac{e_r}{2^{n+1}} \tag{22}$$

This justifies why reducing the size of the domain improves the convergence radius of the expansion to the $n + 1$ power.
Continuing to consider the exposition in [17], to estimate the error for any expansion order, the coefficients of the

same order *i* are used and the sum of their magnitude S_i is considered:

$$S_i = \sum_{|\alpha|=i} |a_\alpha| \tag{23}$$

An exponential is performed to compute the parameters A and B , to match the function S_i in terms of least squares

$$S_i = f(i) = A \cdot exp(B\,n) \tag{24}$$

Once A and B are found, the fit function $f(i)$ is used to compute the value of $f(n{+}1)$ and thus the size S_{n+1}. The mathematical background that supports this technique is the fact that on a sufficiently small domain, the terms of any convergent power series converge at least exponentially [15]. If S_{n+1} is too big with respect to the error threshold, the domain is split and new expansions are evaluated over the two new domains. In accordance with Equation 22, the terms of any order n in the two new expansions will be smaller by a factor of 2^n than the corresponding terms in the initial expansion [16].

In the case that the Taylor expansion is a multi variable map the previous method is used to check the direction truncation error. Indeed, the process for the computation of the exponential fit $f(i)$ is carried out for each variable v. The value of the fit function magnitude $f_v(n+1)$ is then extracted for each variable, where n is the maximum expansion order. Now it is possible to identify the variable with higher error that corresponds to the higher magnitude $f_v(n+1)$. How will be explained in Section 3.4 the knowledge of direction with higher truncation error is crucial for the domain splitting.

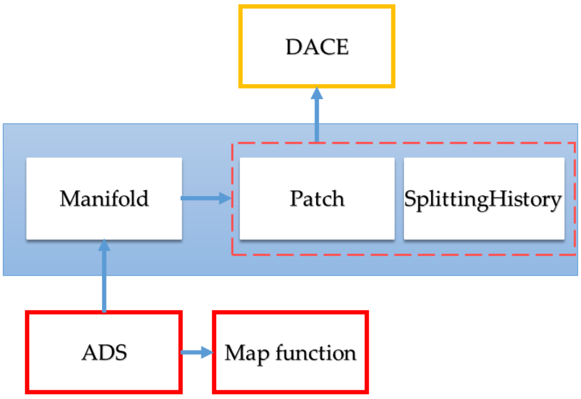


Figure 5. ADS Framework Overview.

3.4. Implementation and Application of ADS

The mathematical concept of manifold is implemented for computer representation into three main class: Manifold, Patch and SplittingHistory.

The respective of the mathematical manifold is the class Manifold. As the mathematical manifold is described by an atlas that is the union of several charts, the class Manifold is described by the union of Patches. The ADS algorithm checks that all the Patches lie within the convergence disk, thanks to the construction of an error controlling function. In this way, the ADS algorithm becomes a member function that decides when the manifold convergence analysis is acceptable.

The Patch is where the Taylor expansion of the map and the scaled initial domain $(-1,1)^v$ are stored, respectively by means of a differential algebraic vector (DAvector) and of the class SplittingHistory. It is also responsible for estimating the error of the expansion map, establishing the direction of the split and obtaining new Patches after performing a split. The SplittingHistory class, on the other hand, is responsible for keeping track of where the new domain is located with respect to the initial one. And it does so by storing integer values representing the variable direction being split in a vector. This class is also equipped with a member function that extrapolates information about the domain and can replay it after or during the splitting. In Section 3.3 the criterion of truncation error estimation and the direction of splitting is established. The common procedure for different types of map functions and domains is now described. The first step is the definition of the initial domain, thus an initial Patch, which at the beginning of the procedure coincides with the entire manifold. The ADS routine needs the definition of a map function and of a domain as a vector of DA elements to be able to store them into the DA Patch and in the DA Manifold. Then the truncation error of the TPS is computed and if it is higher than threshold, the variable with maximum error is identified and then the domain is split along it. The ADS algorithm thus automatically analyses the manifold and then constructs the atlas structure that represents it.

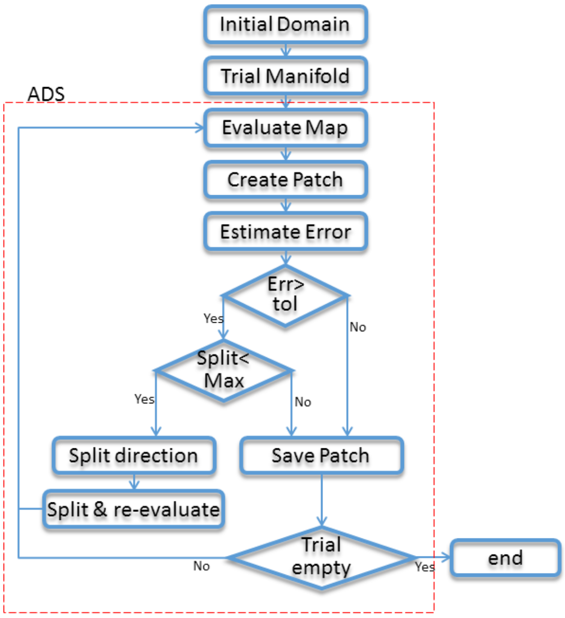


Figure 6. ADS routine to evaluate the scalar and vector function with single and multi variable domain

Thanks to its flexibility by just changing the definition of the initial DAmanifold, ADS can handle: Single Variable Scalar Functions, Multi-Variate Scalar Functions and Multi-Variate Vector Functions. The routine is shown in Figure 6. Furthermore the ADS automatically interact with the different classes for the computer environment implementation. An overview of the ADS routine with the connection between the different structures are represented in Figure 5 . The arrows indicate that the class pointing takes the properties of the class/object being pointed to, the red boxes indicate the functions/operators, while the dashed box represents the close link between the two enclosed classes, each class has no sense without the other.

4. RESULTS

4.1. DAIOD: AR and comparison with literature

In Section 1, the concepts of *attributable* and *admissible region (AR)* were introduced as part of the literature available on IOD of space debris. Here, they are represented in order to make comparisons between the method from literature [9] and the current approach. Remembering the definitions given in Section 1 we want to stress the difference between the two types of admissible regions (ARs): the AR from literature is the result of a deterministic approach where the observation is error-free and the area within the $(\rho, \dot{\rho})$ plane is created by posing physical constraints, while the AR for this paper is the projection of the OS on the $(\rho, \dot{\rho})$ plane, thus being a solution point and its neighboring area created by the uncertainties in the observations. The concept of AR is used here to have a visual understanding of the work carried out, however the results for the DAIOD algorithm are always found in terms of the OS, thus in 6 dimensions.
The TLEs of object with NORAD ID 14786, a satellite in a GEO orbit, are used to create the simulation with the VO. Figure 7 shows the AR for the simulated attributable. The energy law is used to constrain the region, by substituting in it respectively a minimum/maximum semimajor axis (a_{min}, a_{max}) and a maximum eccentricity (e_{max}). In this way three constraining inequalities are obtained: $\mathcal{E}_{a_{max}} < 0$, $\mathcal{E}_{a_{min}} < 0$ and $\mathcal{E}_{e_{max}} < 0$. The output is a plot with a Matlab implementation of the algorithm defined in [9]. The light blue region shows all the possible combinations of ρ and $\dot{\rho}$ that would satisfy the physical constraints to complete the attributable and be a possible state for the observed object. As can be seen, it is a fairly big region. It could be shrunk if one had a priori information about the object. For example, if one knew that the satellite was in a GTO, one would put more stringent constraints on the eccentricity and on the semimajor axis, shown as dashed lines in Figure 7, thus obtaining the blue AR, where the region containing circular orbits has been canceled out and only eccentric orbits remain. However, a priori knowledge is in general not available. The AR for the current approach is found by projecting the OS found with the DAIOD algorithm onto the $(\rho, \dot{\rho})$ plane. The resulting box-shaped AR is shown in Figure 8 superimposed on the AR presented in Figure 7. It is easy to check that the real solution, that is the value used to simulate the observation with the VO, is comprised in both regions, but the box-shaped region obtained with the approach here described is much smaller: indeed, it is already possible to say that the observed object is in a high-Earth orbit with small or zero eccentricity. However, as already underlined in Section 2.3, one box may not be accurate enough to describe the entire region and

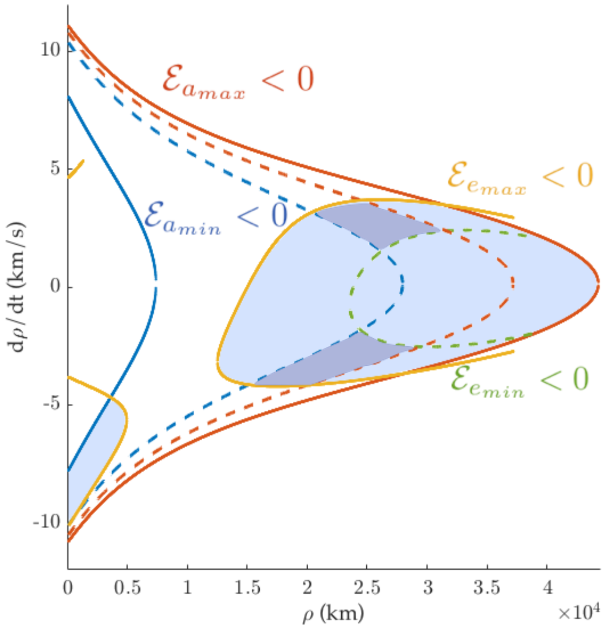


Figure 7. AR from literature. The lightblue area is the general AR, while the blue area is the AR for a GTO object.

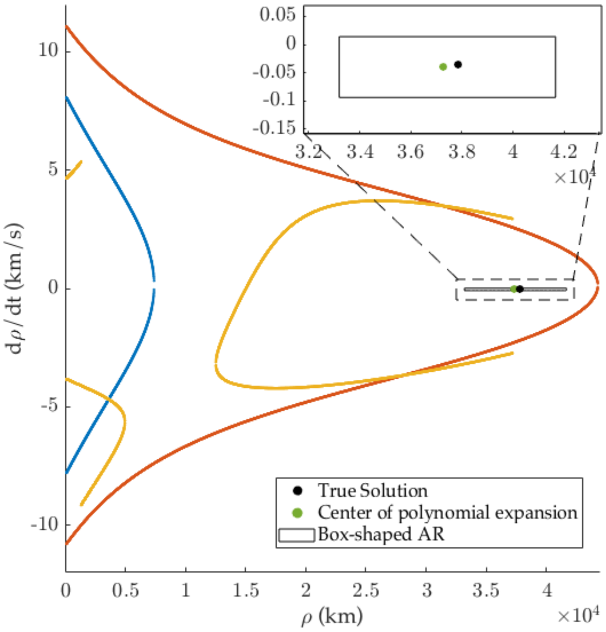


Figure 8. Comparison between ARs: literature vs. presented method

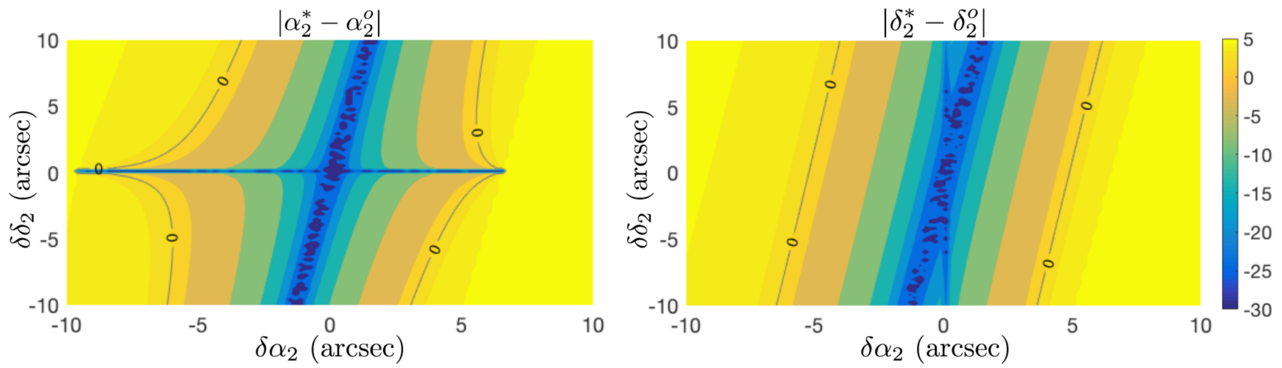


Figure 9. Logarithmic contour plot to check the validity of the map depending on perturbations of α_2 and δ_2. A line is shown for errors equal to $1''$.

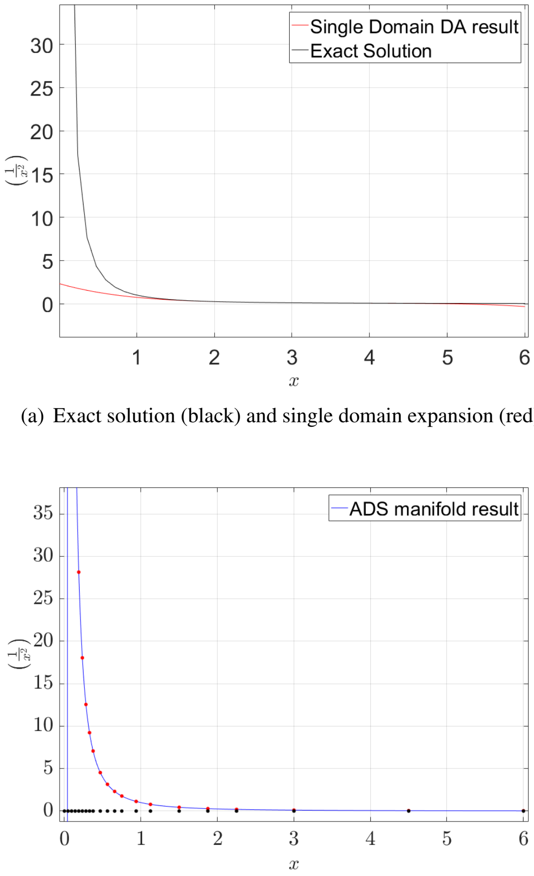
the same simulation will be analyzed and split with the ADS tool in Subsubsection 4.2.3. To prove the necessity of the ADS, a quick check is carried out: if the OS map were to be valid everywhere, the error between the perturbed observation (referred to with a star α^*, δ^*) and the angles retrieved with an evaluation of the perturbed state (referred to with a circle α^o, δ^o) would always be small or zero. Figure 9 shows the contour plot of the differences, where the axes are defined by the $(\delta\alpha_2, \delta\delta_2)$-plane for the central simulated observation. As can be seen, the map is valid in the origin (unperturbed point), but a deviation of few arcseconds would invalidate it. Remembering that observations can have variations $\sigma_P \in [1'', 10'']$, the map needs to be able to handle them without introducing significant numerical errors and for this reason the initial $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain needs to be split.

4.2. ADS results

As introduced in Section 3.4 the ADS routine can be used on different kinds of maps and for different purposes. The three different cases introduced in Subsection 3.4 will now be used to validate the tool.

4.2.1. Single Variable Scalar Function

A single-variable scalar function is a map $f : \Re \rightarrow \Re$. The DA approach finds a k-th order polynomial that approximates the function in the center of the domain. The ADS algorithm stores the entire domain in an input Patch and the function evaluation of the domain in an output Patch: if the error is below a fixed threshold, the Patch is stored into the result manifold, otherwise the domain is split. The process then computes the optimal split direction. In this example it gives a trivial result, that is the single variable of the domain. The two new Patches are then inserted into the trivial manifold and the process is resumed. To avoid too small sub-domains and limit the runtime, a maximum number of splits N_{max} is also fixed. The resulting manifold contains the union of the Patches which provides a more accurate approximation than the one achievable with a single polynomial evaluation.



(a) Exact solution (black) and single domain expansion (red)

(b) Solution with ADS (blue). The black dots on the x-axis define the extremes of the sub-domains.

Figure 10. ADS on single variable scalar function: the inverse square.

To illustrate the code the inverse-square function $\frac{1}{x^2}$ is considered on the domain of $[0.001, 6]$. The domain is intentionally asymmetrical to avoid the DA looking for an expansion in the singularity point but it is closer as possible to it. The TPS order was defined to be 5, the error estimation tolerance to be 10^{-4} and the maximum number of split for each sub-domain to be 7. Figure 10(a) shows that a single map badly approximates the function at the boundary of the domain and close to the singular point. Figure 10(b), on the other hand, shows an increase in precision thanks to the Manifold of 34 Patches. Further increasing of the number of split allows to obtain a higher precision more and more close to the singularity point. The reason for the presence of so many sub-domains close to the singular point is that the Taylor approximation is poor in the neighborhood of the singularity, thus the ADS routine keeps splitting the domain close to the singularity point until the maximum number of splits is reached.

4.2.2. Multi Variable Scalar Function

Carrying on the illustration of the ADS routine for multi-variable scalar functions, the algorithm works in the same way as for the previous case, with the only difference that the splitting direction is now not trivial anymore.
To illustrate the ADS with the multi-variable scalar function the Gaussian function defined on a squared domain in $[-0.5, 1.5] \times [-0.5, 1.5]$ with parameters $\boldsymbol{\mu} = (0.5\ \ 0.5)$ and $\boldsymbol{\sigma}^2 = \text{diag}(0.1,\, 0.01)$ has been chosen. The non symmetric function is used to understand how the ADS changes the preferred direction according to the shape of function. The expansion order is chosen to be 10, the error tolerance 10^{-5} and the maximum number of splits 10. As can be seen in Figure 11 the ADS routine applied to the Gauss function returns a manifold of 64 patches and the major split are made along the stretched direction. The increase in the accuracy can be achieved by decreasing the error tolerance. Figure 12(a) shows the error for the single expansion on the whole domain, while Figure 12(b) shows the error obtained with the ADS, which is kept below the threshold. The absolute error is computed as the difference between the actual values and the TPS evaluation.

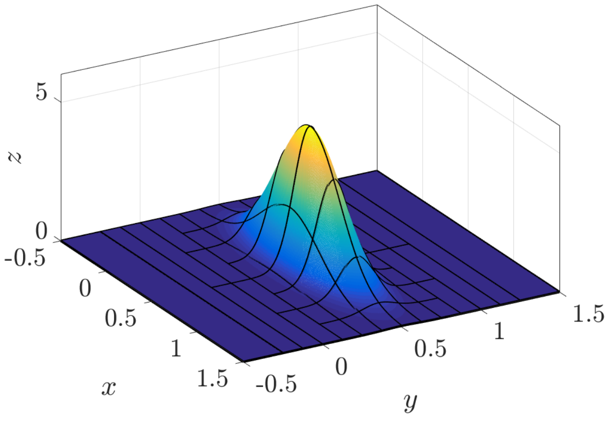
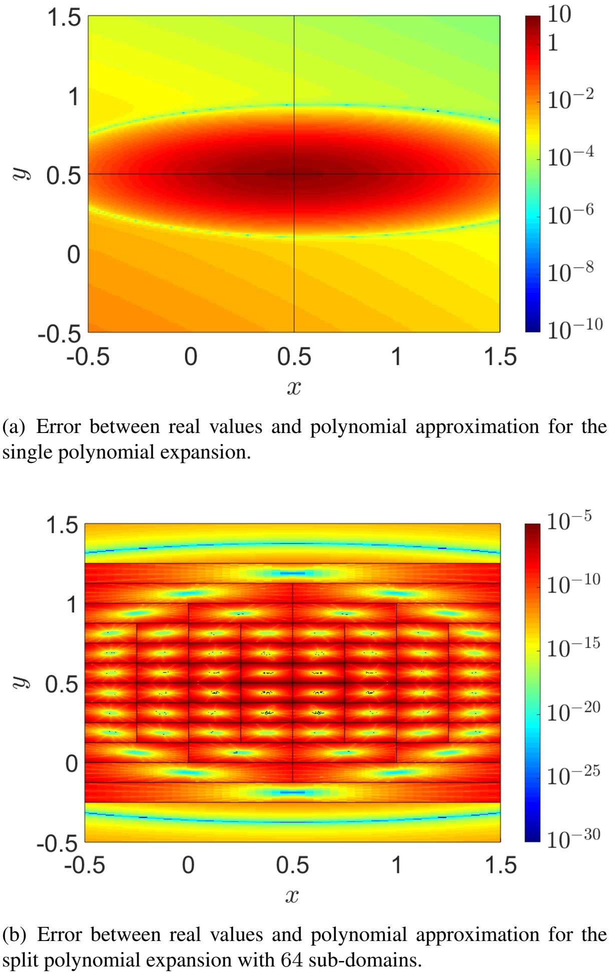


Figure 11. Gauss function exact solution and superimposed manifold representation



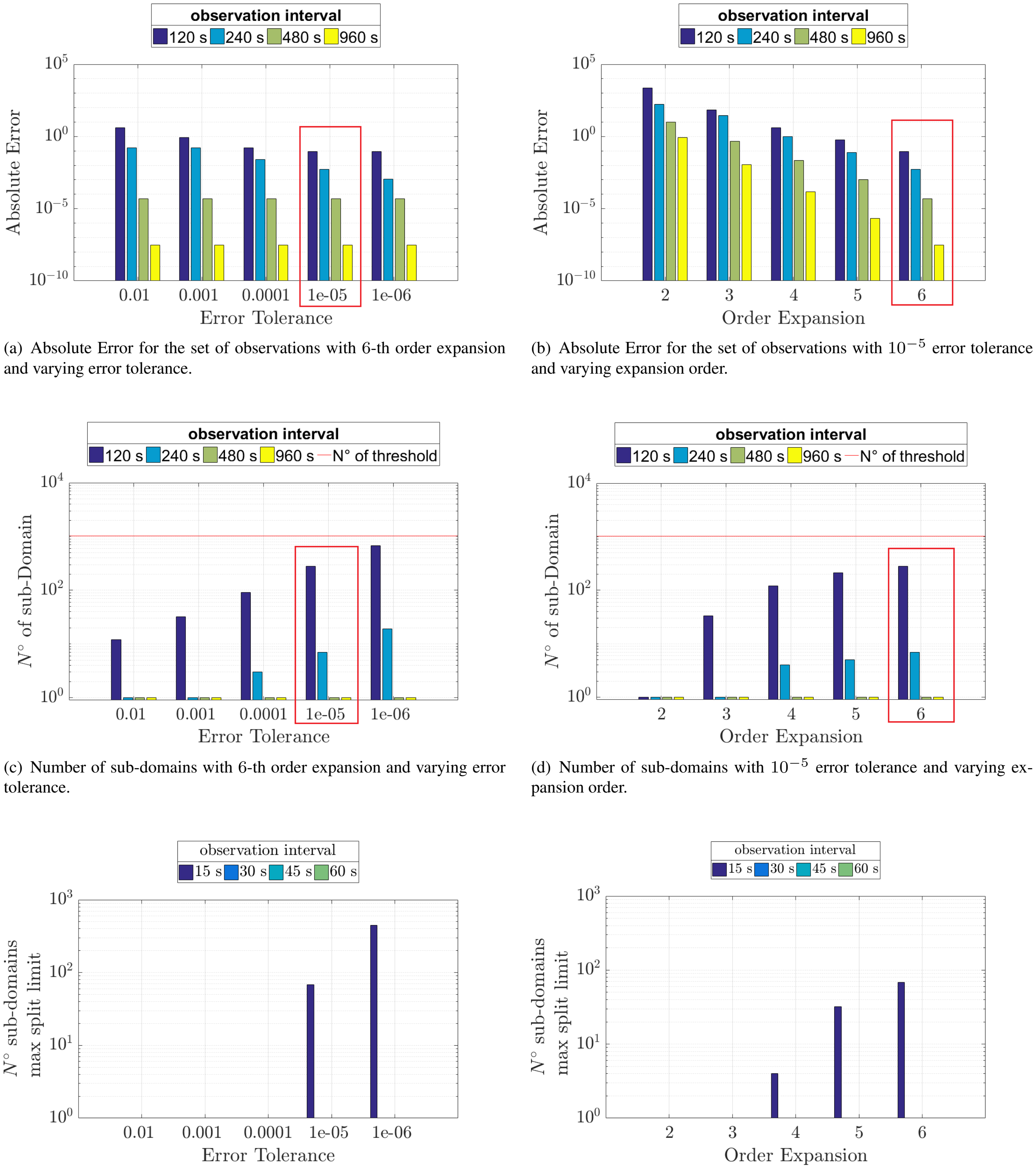
(a) Error between real values and polynomial approximation for the single polynomial expansion.

(b) Error between real values and polynomial approximation for the split polynomial expansion with 64 sub-domains.

Figure 12. ADS on multiple variable scalar function: the gaussian function

4.2.3. Multi Variable Vector Function: DAIOD

Multi-variable vector functions $f : \Re^k \to \Re^m$ differ from the structure presented before in the error estimation process. Indeed, because it is a vectorial function, the estimation is carried out for each component of the map. The component with the largest error estimation is then used to decide the splitting direction. The only caveat in the vector case is the definition of the “largest error”. Indeed, the different components of the vector often are not comparable (e.g. angles and distances), and hence a single splitting threshold is not appropriate. In such cases, a weighting factor can be applied to each component of the error estimate. For the the multi-variable vector function case the initial OS determination with the DAIOD algorithm is considered. The function takes as input the Observation Set and returns the TPS of the object state at the central time of the observation t_2, i.e $(\boldsymbol{x}_2, \boldsymbol{v}_2)$. The DAIOD map is a function that transforms the Taylor poly-



(a) Absolute Error for the set of observations with 6-th order expansion and varying error tolerance.

(b) Absolute Error for the set of observations with 10^{-5} error tolerance and varying expansion order.

(c) Number of sub-domains with 6-th order expansion and varying error tolerance.

(d) Number of sub-domains with 10^{-5} error tolerance and varying expansion order.

(e) Number of sub-domains that reach the maximum number of split with 6-th order expansion and varying error tolerance.

(f) Number of sub-domains that reach the maximum number of split with 10^{-5} error tolerance and varying expansion order.

Figure 13. Evaluation of the Absolute Error of the TPS and total number of sub-domains depending on Order Expansion and Error Tolerance. The red boxes highlight the results for the same condition: 6*-th order expansion,* 10 *maximum splits and* 10^{-5} *tolerance*

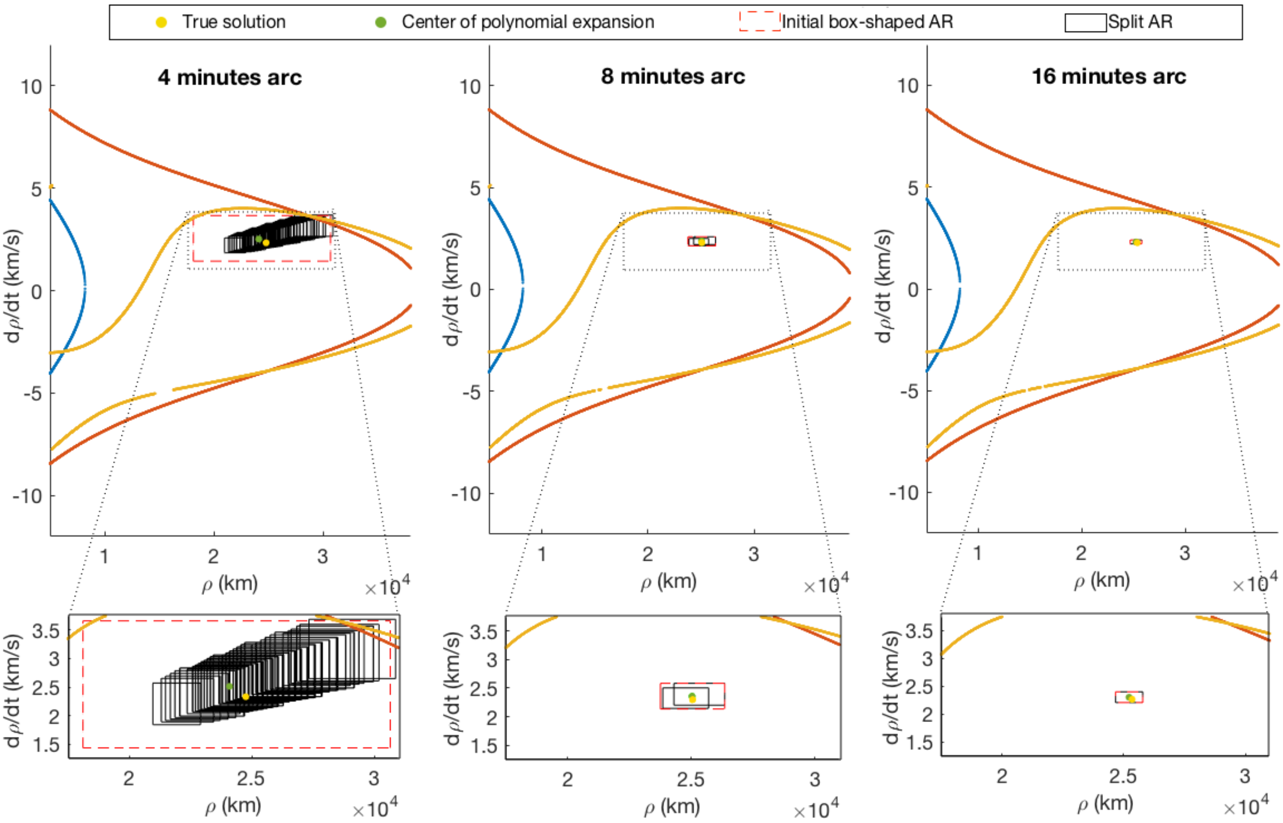


Figure 14. Plot of the AR for IODs performed with observations taken at different Δt: the smaller the time between two observations is, the bigger the uncertainty on the solution is.

nomial of the domain onto the Taylor polynomial of the solution around the nominal value of the phase space with arbitrary expansion order of the TPS and it can be analysed by the ADS to control the error.

Results are shown for the object with NORAD ID 25542, which is in a GTO orbit. The DAIOD is performed for four different time intervals between the observations. There are three main parameters that affect the results: the order of expansion, the maximum number of splits and the error tolerance. These parameters all influence the maximum error of the expansion and their effect can be appreciated in Figure 13. The Absolute error is computed as the maximum of the difference between the value of polynomial evaluation and the actual value performed on the vertices of every sub-domain.

Figure 13(a) shows that the absolute error decreases when a more stringent error tolerance is applied during the ADS routine. In all case, observations that are more separated in time always yield more precise results, as already pointed out in Section 1. For large uncertainties, the error estimation process may produce an inaccurate error estimation which results in the absolute error being bigger than the tolerance. The connection between Figure 13(a) and 13(c) is also clear: the smaller the tolerance, the more the domain is split to keep the absolute error small enough. The effect of the expansion order is analyzed against the absolute error and the number of subdomains, fixing the error tolerance be 10^{-5}. Figure 13(b) shows that by raising the expansion order it is possible to obtain a more accurate estimation of the function, indeed even when the ADS never splits the domain (the 480s and 960s orbits), the absolute error decreases. It is also possible to see in Figure 13(d) that the TPS performed with lower expansion orders are not able to accurately estimate the truncation error, thus the ADS does not split the initial domain, even though the absolute error is beyond the tolerance. From the results it is clear that we need to use observations as much separated in time as possible and we need a high-order computation to obtain accurate estimation. The last notable consideration is that when some sub-domains reach the maximum number of splits, as shown in Figure 13(e) and 13(f), the respective TPSs yield an estimate error larger than the tolerance, hence the maximum absolute error on the whole manifold will be larger than the tolerance itself, see Figure 13(a) and 13(b).

To better understand the behavior of the DAIOD with respect to the time interval between subsequent observations, the OSs obtained for separations of 120, 240 and 480 seconds are projected on the $(\rho,\,\dot{\rho})$ plane and shown in Figure 14. The first notable consideration about the results is that the smaller the time between two observation is, the bigger the uncertainties become. This is visible from the dimension of the AR in the first plot with respect to the other two. Furthermore the ADS produces a higher number of sub-domains when uncertainties are

bigger, resulting in an increase in precision but also in the computational time to produce the output.
Now that the explanation of the DAIOD with ADS is completed, it is possible to apply the ADS to the observation taken into account in Section 4.1 and shown in Figure 8. It was said that a single TPS was not enough to approximate the IOD so that ADS was needed to obtain a better estimate for the OS found. Figure 15 shows the split AR.

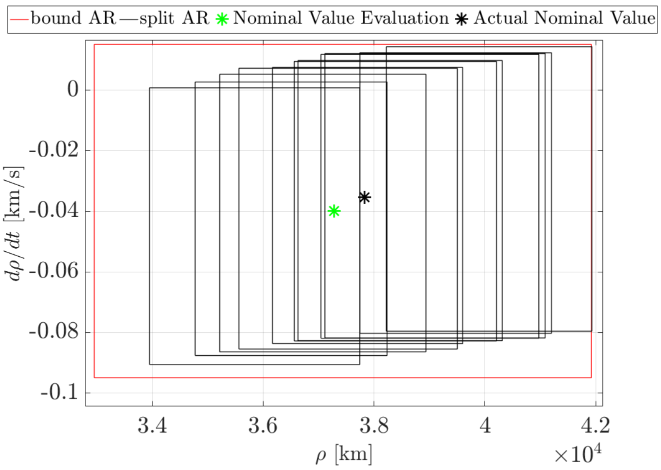
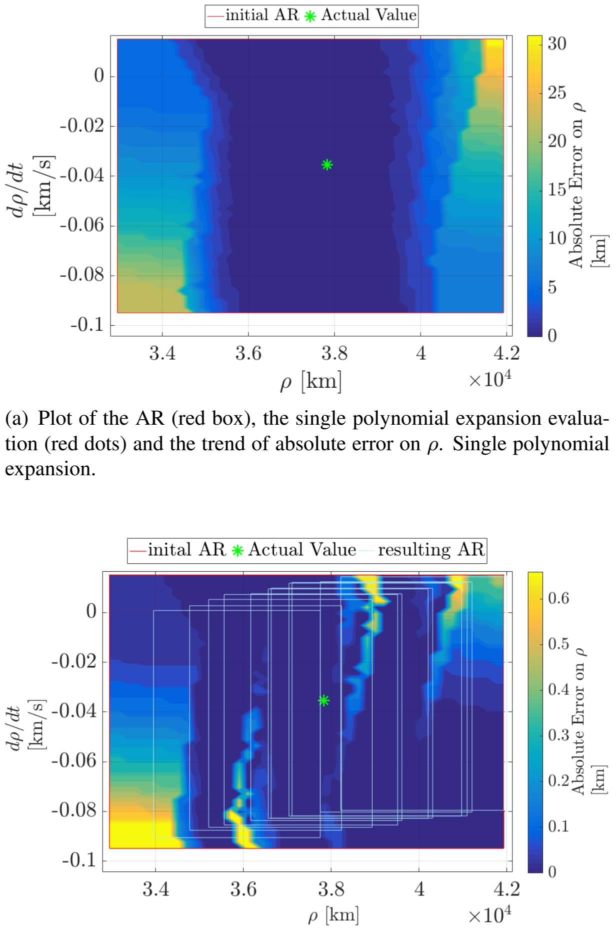


Figure 15. Plot of the AR with ADS for the Observation Set presented in Figure 8.

Other than yielding a known maximum error on the entire domain, the split AR is also smaller than the initial one, thanks to the error estimation process. Figures 16(a) and 16(b) show the absolute error of ρ computed as the difference between the actual values of and the estimated ones for, respectively, a single polynomial expansion and the output with ADS. The parameters for the ADS are: 5th-order expansion, error tolerance of $10^{-3}$and 10^{-6}, respectively for the range and range-rate and maximum number of splits equal to 10. A grid of 729 equally spaced values is created and for each point the absolute error of ρ is calculated. Then with those values a polynomial fit is obtained and projected on the $(\rho,\ \dot{\rho})$ plane to show a continuous trend of the absolute error. Figure 16(a) shows that the single expansion is quite accurate close to the expansion point but is very imprecise close to the borders, while Figure 16(b) show that the manifold evaluation yields a smaller error on the entire split domain.

5. CONCLUSIONS AND FUTURE WORK

This work has achieved three main goals concerning the IOD and propagation of the obtained OS: the developing of the VO, the implementation of the IOD within the DA framework and the analysis of the DAIOD algorithm with the ADS.



(a) Plot of the AR (red box), the single polynomial expansion evaluation (red dots) and the trend of absolute error on ρ. Single polynomial expansion.

(b) Plot of the AR with ADS (black boxes), the manifold polynomial evaluation (red dots) and the trend of absolute error on ρ. Manifold polynomial expansion.

Figure 16. Plots of the error distribution on the domain for single polynomial expansion and manifold of polynomials obtained with ADS.

The VO is a simulating tool that receives as input the TLEs of real known objects, the Δt between two consecutive observations, the position of the observer and the noise to be added to the observations. The output of the VO is the Observation Set, which is used by the DAIOD algorithm to find the OS by exploiting the well-known Gauss' and Lambert's problems in DA environment. With the DA tools it is indeed possible to use an implementation of a new algebra of Taylor polynomial, where a function evaluation is expressed as a Taylor expansion in the neighborhood of the evaluated point. In this way, the OS is represented as a TPS that depends on the uncertainties in the observations and is centered in the solution of the IOD problem. The advantage of finding the solution in terms of a polynomial is that the effect of a perturbation on the initial domain can be computed by means of a function evaluation, instead of repeating the whole procedure.

The results obtained by the DAIOD were shown in terms of the AR, already known in literature, to underline their features visually. The remarkable consideration is the improvement of the solution with respect to the classical method, by exploiting DA and by considering the uncertainties in the observations, instead of using a deterministic approach with physical constraints.

The DA representation of the function as a TPS has a good enough approximation close to the expansion point, but lacks of precision at the boundaries of the domain. The ADS tool has then been implemented to estimate and control the error introduced by the TPS, by halving the domain and re-evaluating new functions as long as the tolerance criterion is met. By merging the DAIOD and the ADS, we obtained a substantial improvement of the description of the OS. First of all, the size of the OS is reduced and secondly the error introduced by the TPSs is known and controled. By analyzing the sensitivity of the OSs with respect to the separation time between subsequent observations, it is possible to appreciate that better solutions are found for Observation Sets defined on longer time spans, in accordance with the literature.

The generation of accurate OSs from independent Observation Sets is built with the goal of propagating them to a common epoch to look for correlations within the OSs. We are currently working on the two main tools necessary for this. The first one is a propagator that includes perturbations and is able to work within the DA framework and with the ADS tool. Indeed, due to the nonlinear nature of the dynamics, the uncertainties in the OSs will increase during the propagation, so that the ADS tool will be applied to the propagation too. The second one is a method to assess the likelihood of two observations pertaining to the same object starting from the analysis of the intersection of their OSs. The trivial outcome is that if the regions do not overlap, the objects are uncorrelated. On the other side, if two or more OSs are linked, the orbit of the object can be refined by performing AOD.

ACKNOWLEDGMENTS

The work has been partially funded by the EOARD contract number FA9550-15-1-0244 “Initial Orbit Determination Based on Propagation of Admissible Region With Differential Algebra”.
R. Armellin acknowledges the support received by the Marie Sklodowska-Curie grant 627111 (HOPT - Merging Lie perturbation theory and Taylor Differential algebra to address space debris challenges).

REFERENCES

1. Armellin, R., Di Lizia, P. & Lavagna, M. High-order expansion of the solution of preliminary orbit determination problem. *Celestial Mechanics and Dynamical Astronomy* **112,** 331–352. ISSN: 09232958 (2012).
2. Armellin, R., Di Lizia, P. & Zanetti, R. Dealing with uncertainties in angles-only initial orbit determination. *Celestial Mechanics and Dynamical Astronomy* **125,** 435–450 (2016).
3. Armellin, R., Di Lizia, P. & Zanetti, R. Dealing With Uncertainties in Initial Orbit Determination. *Aas 15-734,* 1–19 (2015).
4. Berz, M. The method of power series tracking for the mathematical description of beam dynamics. *Nuclear Instruments and Methods A258* (1987).
5. Berz, M. *The new method of TPSA algebra for the description of beam dynamics to high orders* Technical Report AT-6:ATN-86-16. Los Alamos National Laboratory (1986).
6. Curtis, H. *Orbital Mechanics: For Engineering Students* ISBN: 9780080470542 (Elsevier Science, 2015).
7. Di Lizia, P., Armellin, R. & Lavagna, M. Application of high order expansions of two-point boundary value problems to astrodynamics. *Celestial Mechanics and Dynamical Astronomy* **102,** 355–375 (2008).
8. Izzo, D. Revisiting Lambert’s Problem (2014).
9. Milani, A. & Gronchi, G. *The theory of orbit determination* 394 (2009).
10. Milani, A., Gronchi, G., de’ Michieli Vitturi, M. & Knezevic, Z. Orbit determination with very short arcs. I admissible regions. *Celestial Mechanics and Dynamical Astronomy* **90,** 59–87. ISSN: 09232958 (2004).
11. M.Lee, J. *Manifold and Differential Geometry* (American Mathematical Society, Providence, Rhode Island, 2009).
12. Rasotto, M. *et al.* Differential algebra space toolbox for nonlinear uncertainty propagation in space dynamics. *6th International Conference on Astrodynamics Tools and Techniques* (2016).
13. Tu, L. W. *An Introduction to Manifolds* (Springer Science, 2011).
14. Vallado, D. A. & McClain, W. D. *Fundamentals of astrodynamics and applications* (Springer Science & Business Media, 2001).
15. Wittig, A. An Introduction to Differential Algebra and the Differential Algebra Manifold Representation. *Astrodynamics Network AstroNet-II* (2016).
16. Wittig, A. *et al.* An automatic domain splitting technique to propagate uncertainties in highly nonlinear orbital dynamics. *Advances in the Astronautical Sciences* **152,** 1923–1941 (Jan. 2014).
17. Wittig, A. *et al.* Propagation of large uncertainty sets in orbital dynamics by automatic domain splitting. *Celestial Mechanics and Dynamical Astronomy* **122,** 239–261. ISSN: 1572-9478 (2015).