AFRL-AFOSR-UK-TR-2017-0022

Initial Orbit Determination based on propagation of admissible regions with Differential Algebra

Pierluigi Di Lizia
POLITECNICO DI MILANO

01/19/2017
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ IOE
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
21-03-2017	Final	15 Jun 2015 to 14 Jun 2018

4. TITLE AND SUBTITLE
Initial Orbit Determination based on propagation of admissible regions with Differential Algebra

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-15-1-0244

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
Pierluigi Di Lizia, Roberto Armellin

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
POLITECNICO DI MILANO
PIAZZA LEONARDO DA VINCI 32
MILANO, 20133 IT

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
EOARD
Unit 4515
APO AE 09421-4515

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/AFOSR IOE

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-AFOSR-UK-TR-2017-0022

12. DISTRIBUTION/AVAILABILITY STATEMENT
A DISTRIBUTION UNLIMITED: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This work contains the definition of the initial orbit determination (IOD) method with some first results, a literature review about the Manifold, the creation of the algorithms and architecture for the ADS and the initial merging of the ADS into the IOD technique. The IOD algorithm is explained, implemented and comparisons with results from literature are also outlined. In specific it takes as input optical observations and gives as output the state of the object at the central time of observation expanded with respect to perturbations on the observations. Since one expansion is not accurate enough to describe the full initial domain, the ADS tool is introduced. This tool is constructed on the mathematical definition of the manifold. The initial state vector is then defined as a set of Charts and the propagator integrates in time this initial manifold while the ADS manages the convergence radius of manifold, or rather the convergence radius of every single Chart. Thus, when a single truncated power series (TPS) is not sufficient to represent the whole manifold, that is when the estimation error of a chart is bigger than the threshold, the ADS splits the manifold and goes ahead with the propagation.

15. SUBJECT TERMS
EOARD, space-debris uncertainty, orbit determination, orbit propagation, differential algebra, space situational awareness

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			MILLER, KENT
Unclassified	Unclassified	Unclassified	SAR	55	**19b. TELEPHONE NUMBER** *(Include area code)* 011-44-1895-616022

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Initial Orbit Determination Based on Propogation of Admissible Region With Differential Algebra

FA9550-15-1-0244

DANIELE ANTONIO SANTERAMO, PIERLUIGI DI LIZIA
Politecnico di Milano

LAURA PIROVANO, ROBERTO ARMELLIN, JUAN FELIX SAN JUAN
Universidad de la Rioja

ALEXANDER WITTIG
Advanced concepts team, ESA

January 18, 2017

Contents

List of Figures

List of Tables

Summary

Following the previous report, where a literature study on current sensor capacities has been carried out together with the mathematical definition of automatic domain splitting (ADS) and virtual observatory (VO), this work contains the definition of the initial orbit determination (IOD) method with some first results, a literature review about the Manifold, the creation of the algorithms and architecture for the ADS and the initial merging of the ADS into the IOD technique. The IOD algorithm is explained, implemented and comparisons with results from literature are also outlined. In specific it takes as input optical observations and gives as output the state of the object at the central time of observation expanded with respect to perturbations on the observations. Since one expansion is not accurate enough to describe the full initial domain, the ADS tool is introduced. This tool is constructed on the mathematical definition of the manifold. The initial state vector is then defined as a set of Charts and the propagator integrates in time this initial manifold while the ADS manages the convergence radius of manifold, or rather the convergence radius of every single Chart. Thus, when a single truncated power series (TPS) is not sufficient to represent the whole manifold, that is when the estimation error of a chart is bigger than the threshold, the ADS splits the manifold and goes ahead with the propagation.

Chapter 1

Introduction

Every day thousands of detections of objects orbiting the Earth are retrieved by observatories. However, a single passage above the station is not sufficient to fully determine an orbit. The previous report showed an approach to perform IOD based on the generation of an AR to find a space of solutions rather than a single point. Indeed, due to the observation geometry and the uncertainty related to sensor accuracy, timing accuracy, and observer state knowledge, it is not possible to obtain a single orbital state. This report is going to outline a new method, called the Differential Algebraic Initial Orbit Determination (DAIOD), which aims to describe the solution of the IOD as a TPS that depends on variations of the observations.

Section 1.1 explains the output of a typical real optical observation. Although for the present work the observations are simulated with the VO (Section 2.2), real data used in previous works are exploited to create realistic outputs. Section 1.2 presents the relevant nomenclature.

1.1 The observations

A typical optical observation is made of four important values:

- t The time at which the observation is made. In a geostationary Earth orbit (GEO), the time between two subsequent observation is typically around $2-3$ min;
- α The longitude of the observed object with respect to the observatory;
- δ The latitude of the observed object with respect to the observatory;
- σ_P The precision of the observation (usually in arcseconds). Typical values for the precision vary from $1''$, being the most accurate observation, to $5-10''$, being the least accurate.

For the work at hand, an optical observatory with a very large field of view (FOV) has been assumed. As outlined by Milani et al. (2004), the closer the observations are in time, the more difficult it is to perform IOD. For this reason, given a full passage over the station, the first, middle and last observations are considered in this work, which results in about 8 min separation. The observations used to perform IOD are simulated with the VO, presented in Section 2.2.

1.2 Nomenclature

In this Section a quick overview of the nomenclature used for the work at hand and in previous papers regarding IOD of space debris is given.

Observation. As outlined in Section 1.1, 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 Milani et al. (2004). Although introduced with such a name, in Milani and Gronchi (2009) and Worthy III and Holzinger the name has been changed to **too short arc (TSA)**, while other authors simply call it **short arc (SA)** (Fujimoto and Scheeres, 2012). However, all the papers agree on its definition: 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. Milani et al. (2004) defined an attributable the useful information that could be extracted from a TSA. It is made of two angles and two angular rates:

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

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 used by all authors to describe the $4\,D$ vector containing partial information about the object observed.

$(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain. Considering the variations of the observed angles as gaussians with zero mean and $\sigma^2_{P,i}$ variance

$$\begin{cases} \delta\alpha_i \sim \mathcal{N}\left(0, \sigma^2_{P,i}\right) \\ \delta\delta_i \sim \mathcal{N}\left(0, \sigma^2_{P,i}\right) \end{cases} \qquad \text{for } i = 1, 2, 3 \tag{1.1}$$

the $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain is the $6D$ region containing the $3\boldsymbol{\sigma}_P$ variation of the angles.

Orbit set (OS). Given the state of the object as a function of $\delta\boldsymbol{\alpha}$ and $\delta\boldsymbol{\delta}$, the orbit set (OS) is the set of the orbital states enclosed by the range 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 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: $\rho(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta})$ and $\dot{\rho}(\delta\boldsymbol{\alpha}, \delta\boldsymbol{\delta})$.

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.

Chapter 2

SPICE and the Virtual observatory (VO)

This chapter introduces two important tools used in the work at hand: Spacecraft Planet Instrument C-matrix Events (SPICE) and the VO. The first one is a powerful tool developed by the Navigation and Ancillary Information Facility (NAIF) that is able to assist scientists in planning and interpreting scientific observations from space-borne instruments, and to assist engineers involved in modeling, planning and executing activities needed to conduct planetary exploration missions. The VO is the tool developed to simulate the observations and is based on SPICE functions.

2.1 SPICE

The Navigation and Ancillary Information Facility (NAIF), acting under the directions of National Aeronautics and Space Administration (NASA)'s Planetary Science Division, has built an information system named SPICE, which use extends from mission concept development through the post-mission data analysis phase, including help with correlation of individual instrument data sets with those from other instruments on the same or on other spacecraft (NAIF, 2016).

As the name reveals, SPICE contains a big variety of information:

- **S** Spacecraft ephemeris. When this file is available, the satellite state can be retrieved at any time within the interval of definition of the file.
- **P** Planet, satellite, comet, or asteroid ephemerides, or more generally, location of any target body, given as a function of time. It also includes physical, dynamical and cartographic constants for target bodies, such as size and shape specifications, and orientation of the spin axis and prime meridian.
- **I** Instrument description kernel, containing descriptive data peculiar to a particular scientific instrument, such as field-of-view size, shape and orientation parameters.
- **C** Pointing kernel, containing a transformation, traditionally called the "C-matrix", which provides time-tagged pointing (orientation) angles for a spacecraft bus or a spacecraft structure upon which science instruments are mounted.
- **E** Events kernel, summarizing mission activities - both planned and unanticipated.

The Events kernel is rarely used. For the DAIOD algorithm, only the S and P kernels will be used. In particular, they will be necessary to create the observation (as explained in Section 2.2), retrieve the observatory and Earth states at certain times, easily recover body constants and make conversions between units of measurements. While the Earth ephemerides, the body

constants and the conversion files are already available in the SPICE download package, the ephemerides of the observatory have to be built with some SPICE toolkits: after loading the Earth ephemerides file and the file containing the conversion between the Earth inertial and Earth-fixed reference frames, the ephemerides of the observatory on the Earth can be computed with the PINPOINT and MKSPK toolkits.

2.2 The Virtual observatory

The VO, referred to as IRIS from now on, is a tool that can simulate optical and radar survey scenarios. It is based on a high fidelity numerical propagator (including Earth geopotential, third body perturbations, solar radiation pressure and atmospheric drag) capable of accurately predicting the motion of object in different orbital regimes: low-Earth orbit (LEO),GEO,medium-Earth orbit (MEO) and high-Earth orbit (HEO). The object ephemerides are automatically converted into SPICE kernels (NAIF, 2016) to accurately simulate observations from any ground location. The observations are simulated implementing different visibility constraints and producing outputs compatible with real sensors.

The IRIS tool is based on the automatic generation of SPICE kernels for both space object trajectories and observatories. The inputs set by the user are (see yellow boxes in Figure 2.1):

- object initial conditions;
- object parameters;
- simulation window;
- observer location;
- instrument type and accuracy statistics.

The initial conditions of the space object can be given in different formats, i.e., two-line elements (TLE), Cartesian coordinates, or osculating classical orbital elements. Once the user has set the initial space object state and the model parameters, the tool computes the object trajectory through a high-fidelity orbital propagator, referred to as Accurate Integrator for Debris Analysis (AIDA), and generates its corresponding kernel through SPICE functions. Figure 2.1 shows the IRIS tool together with SPICE functionalities and AIDA.

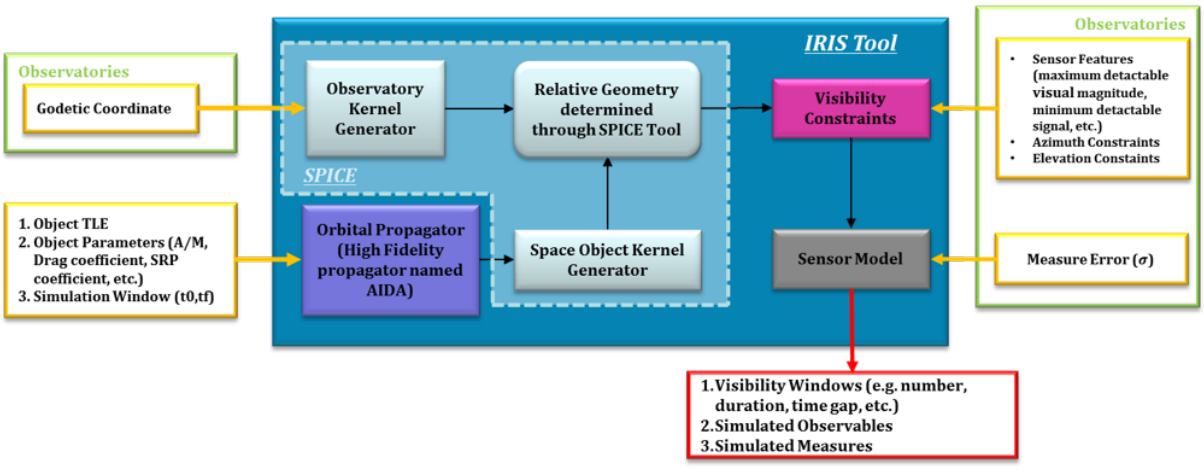


Figure 2.1: IRIS software tool description.

Once the object and observatories kernels are built with the necessary toolkits, all the powerful features of SPICE can be exploited to study the relative geometry between the object and observer. This includes the possibility of generating observables in a large number of inertial, local, and user defined reference frames, avoiding all the nuisances associated to time conversions, reference frame conversion, and observation details such light time and stellar aberration. The input deriving from the sensor type is used to quickly compute the visibility windows and to simulate the associated observables. More specifically, all constraints deriving from the sensor type are evaluated by exploiting the functionalities offered by SPICE. For instance, whenever one considers an optical sensor, some aspects such as the sky background luminosity, the object illumination, and the object elevation can be taken into account to define the observability windows. Finally, the relative geometry between the observatories and observable objects can be used in combination with user-defined measurement noises or as input for sensor simulators to generate simulated observations.
The output of the IRIS tool is the *observations set* defined in Section 1.2.

Chapter 3

Initial orbit determination (IOD)

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 need to be solved. This problem is known as initial orbit determination (IOD). When many more observations are taken over an orbit arc of adequate length, accurate orbit determination (AOD) can be performed (Armellin et al., 2015). As outlined in the previous report, there are currently two different types of sensors able to survey the sky: radar and optical. So far, the latter has been taken into account to build the algorithm for IOD and the type of observation was explained in Section 1.1. Then, Section 2.1 outlined some SPICE functions necessary to retrieve constants and fixed objects states. Now, other tools necessary to complete the IOD algorithm are explaine: in Section 3.1 some differential algebra (DA) tools are introduced, Section 3.2 shows two well-known algorithms and, lastly, the mathematical definition of the IOD algorithm will be performed in Section 3.3. Results and comparisons with literature are then outlined in Section 5.1.

3.1 Introduction to differential algebra (DA) tools

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. 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 (Berz, 1986, 1987). In the following subsections, the expansion of the solution of parametric implicit equations and the nonlinear mapping of the estimate statistics are explained. The descriptions are taken from Armellin et al. (2012) and Armellin et al. (2015).

3.1.1 Expansion of the solution of parametric implicit equations

Well-established numerical techniques (e.g., Newton's method) exist, which can effectively identify the solution of a classical implicit equation

$$f(x) = 0. \tag{3.1}$$

Suppose an explicit dependence on a parameter p can be highlighted in the previous function f, which leads to the parametric implicit equation

$$f(x, p) = 0. \tag{3.2}$$

Suppose the previous equation is to be solved, whose solution is represented by the function $x(p)$ returning the value of x solving (3.2) for any value of the parameter p. Thus, the dependence of the solution of the implicit equation on the parameter p is of interest. DA techniques can effectively handle the previous problem by identifying the function $x(p)$ in terms of its Taylor expansion with respect to the parameter p. The DA-based algorithm is presented in the following for the solution of the scalar parametric implicit Eq. (3.2); the generalization to a system of parametric implicit equations is straightforward.

The solution of (3.2) is sought, where sufficient regularity is assumed to characterize the function f; i.e., $f \in \mathcal{C}^{k+1}$. This means that $x(p)$ satisfying

$$f(x(p), p) = 0 \tag{3.3}$$

is to be identified. The first step is to consider a reference value of the parameter p and to compute the solution x by means of a classical numerical method; e.g., Newton's method. The variable x and the parameter p are then initialized as k-th order DA variables, i.e.,

$$\begin{array}{rcl} [x] & = & x + \delta x \\ [p] & = & p + \delta p. \end{array} \tag{3.4}$$

A DA-based evaluation of the function f in (3.2) delivers the k-th order expansion of f with respect to x and p:

$$\delta f = \mathcal{M}_f(\delta x, \delta p), \tag{3.5}$$

where $\mathcal{M}_f$ denotes the Taylor map for f. Note that the map (3.5) is origin-preserving as x is the solution of the implicit equation for the nominal value of the parameter p; thus, δf represents the deviation of f from its reference value. The map (3.5) is then augmented by introducing the map corresponding to the identity function on p (i.e., $\delta p = \mathcal{I}_p(\delta p)$) ending up with

$$\begin{bmatrix} \delta f \\ \delta p \end{bmatrix} = \begin{bmatrix} \mathcal{M}_f \\ \mathcal{I}_p \end{bmatrix} \begin{bmatrix} \delta x \\ \delta p \end{bmatrix}. \tag{3.6}$$

The k-th order map (3.6) is inverted, obtaining

$$\begin{bmatrix} \delta x \\ \delta p \end{bmatrix} = \begin{bmatrix} \mathcal{M}_f \\ \mathcal{I}_p \end{bmatrix}^{-1} \begin{bmatrix} \delta f \\ \delta p \end{bmatrix}. \tag{3.7}$$

The inversion is a built-in tool in the C++ implementation of DA, called DACE. As the goal is to compute the k-th order Taylor expansion of the solution manifold $x(p)$ of (3.2), the map (3.7) is evaluated for $\delta f = 0$

$$\begin{bmatrix} \delta x \\ \delta p \end{bmatrix} = \begin{bmatrix} \mathcal{M}_f \\ \mathcal{I}_p \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ \delta p \end{bmatrix}. \tag{3.8}$$

The first row of map (3.8)

$$\delta x = \mathcal{M}_x(\delta p), \tag{3.9}$$

expresses how a variation of the parameter δp affects the solution of the implicit equation as a k-th order Taylor polynomial. In particular, by plugging map (3.8) in first of (3.4) we obtain

$$[x] = x + \mathcal{M}_x(\delta p), \tag{3.10}$$

which is the k-th order Taylor expansion of the solution of the implicit equation. For every value of δp, the approximate solution of $f(x, p) = 0$ can be easily computed by evaluating the Taylor polynomial (3.10). The solution obtained by means of map (3.10) is a Taylor approximation of the exact solution of Eq. (3.2). The accuracy of the approximation depends on both the order of the Taylor expansion and the displacement δp from the reference value of the parameter. Thus, a careful analysis is always mandatory to tune the expansion order to assure that Eq. (3.10) is sufficiently accurate for the entire range of p we are interested in.

3.1.2 Nonlinear mapping of the estimate statistics

Consider a random variable $\boldsymbol{x} \in \Re^n$ with probability density function (PDF) $p(\boldsymbol{x})$ and a second random variable $\boldsymbol{y} \in \Re^m$ related to $\boldsymbol{x}$ through the nonlinear transformation

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}). \tag{3.11}$$

The problem is to calculate a consistent estimate of the main cumulants of the transformed pdf $p(\boldsymbol{y})$.

The Taylor expansion of $\boldsymbol{y}$ with respect to deviations $\delta\boldsymbol{x}$ can be obtained automatically by initializing the independent variable as a DA variable and evaluating (3.11) in DA framework. For the i-th component of $\boldsymbol{y}$, this procedure delivers

$$[y_i] = f_i([\boldsymbol{x}]) = y_i + \mathcal{M}_{y_i}(\delta\boldsymbol{x}) = \sum_{p_1+\cdots+p_n \leqslant k} c_{i,p_1\ldots p_n} \cdot \delta x_1^{p_1} \cdots \delta x_n^{p_n}, \tag{3.12}$$

where in this expression y_i is the zero-th order term of the expansion map, and $c_{i,p_1\ldots p_n}$ are the Taylor coefficients of the resulting Taylor polynomial

$$c_{i,p_1\ldots p_n} = \frac{1}{p_1! \cdots p_n!} \cdot \frac{\partial^{p_1+\cdots+p_n} f_i}{\partial x_1^{p_1} \cdots \partial x_n^{p_n}}. \tag{3.13}$$

The evaluation of (3.12) for a selected value of $\delta\boldsymbol{x}$ supplies the k-th order Taylor approximation of y_i corresponding to the displaced independent variable. The Taylor series in the form (3.12) can be used to efficiently compute the propagated statistics (Valli et al., 2012; Park and Scheeres, 2006). The method consists of analytically describing the statistics of the solution by computing the l-th moment of the transformed pdf using a proper form of the l-th power of the solution map (3.12). The result for the first two moments is

$$\begin{cases} \mu_{y_i} = E\{[y_i]\} = \displaystyle\sum_{p_1+\cdots+p_n \leqslant k} c_{i,p_1\ldots p_n} E\{\delta x_1^{p_1} \cdots \delta x_n^{p_n}\} \\ \boldsymbol{P}_{y_i y_j} = E\{([y_i]-\mu_i)([y_j]-\mu_j)\} = \displaystyle\sum_{\substack{p_1+\cdots+p_n \leqslant k, \\ q_1+\cdots+q_n \leqslant k}} c_{i,p_1\ldots p_n} c_{j,q_1\ldots q_n} E\{\delta x_1^{p_1+q_1} \cdots \delta x_n^{p_n+q_n}\}, \end{cases} \tag{3.14}$$

where $c_{i,p_1\ldots p_n}$ are the Taylor coefficients of the Taylor polynomial describing the i-th component of $[\boldsymbol{y}]$ (in the covariance matrix formula, the coefficients $c_{i,p_1\ldots p_n}$ and $c_{j,q_1\ldots q_n}$ are updated to include the subtraction of the mean). Note that the expectation values on the right side of Eq. (3.14) are function of the known $p(\boldsymbol{x})$.

When $\boldsymbol{x}$ is a Gaussian random variable, its statistics are completely described by the first two moments, i.e. the mean $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{P}$. The expectation value terms of Eq. (3.14) are thus functions of the initial mean and covariance only and they can be computed applying Isserlis's formula (Isserlis, 1918). The resulting moments are then used to describe the transformed PDF.

3.2 Gauss's and Lambert's Algorithms

3.2.1 Gauss's Algorithm

Gauss's 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 (ρ_1, ρ_2, ρ_3) in order to obtain the object positions

$$\boldsymbol{r}_i = \boldsymbol{R}_i + \rho_i \hat{\boldsymbol{\rho}}_i, \qquad \text{where } i = 1,2,3 \tag{3.15}$$

in two-body dynamics. Figure 3.1 shows the geometry of the problem. The description of this algorithm refers to Curtis (2015).

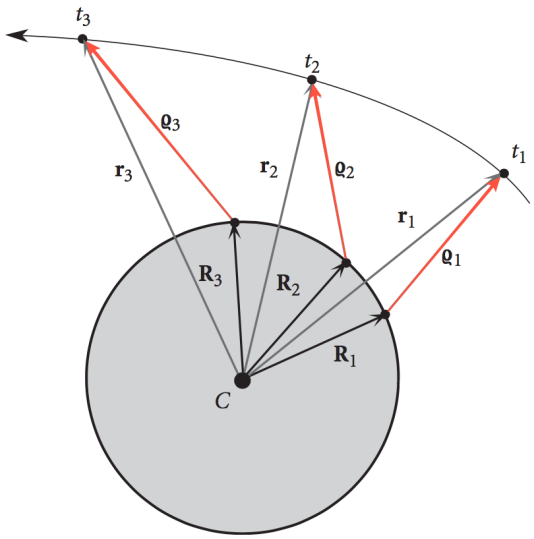


Figure 3.1: Geometry of input and output values for Gauss's algorithm. Modified from Curtis (2015).

Vector equation 3.15 is made of 9 scalar equations and 12 unknowns. Three additional scalar equations can be added by remembering that the three position vectors must lie in the same plane, due to the angular momentum conservation. Thus:

$$\boldsymbol{r}_2 = c_1 \boldsymbol{r}_1 + c_3 \boldsymbol{r}_3. \tag{3.16}$$

However Equation 3.16 introduces two new unknowns, bringing the total of equations and unknowns to respectively 12 and 14. Another consequence of the two-body dynamics is that $\boldsymbol{r}$ and $\boldsymbol{v}$ can be expressed in terms of the state vector at any given time by means of the Lagrange coefficients. In this case the two extremes $\boldsymbol{r}_1$ and $\boldsymbol{r}_3$ can be defined in terms of the state at central time t_2:

$$\begin{cases} \boldsymbol{r}_1 = f_1 \boldsymbol{r}_2 + g_1 \boldsymbol{v}_2 \\ \boldsymbol{r}_3 = f_3 \boldsymbol{r}_2 + g_3 \boldsymbol{v}_2 \end{cases} \tag{3.17}$$

where f_i, g_i depend on t_2, r_2 with reasonable precision. This means that 6 equations and 4 unknowns are added, having at the end 18 equations and 18 unknowns, to be solved to obtain the values $\boldsymbol{r}_1, \boldsymbol{r}_2$ and $\boldsymbol{r}_3$.

The algorithm starts by calculating quantities that are going to be useful in the process:

$$\boldsymbol{p}_i = \hat{\boldsymbol{\rho}}_j \times \hat{\boldsymbol{\rho}}_k, \quad \text{where } \{i,j,k\} = \{1,2,3\}, \{2,3,1\}, \{3,1,2\} \tag{3.18}$$

$$\tau_1 = t_1 - t_2, \quad \tau_3 = t_3 - t_2, \quad \tau = t_3 - t_1 \tag{3.19}$$

$$D_{ij} = \boldsymbol{R}_i \cdot \boldsymbol{p}_j, \quad \text{where } i = 1,2,3 \tag{3.20}$$

$$D_0 = \hat{\boldsymbol{\rho}}_1 \cdot \boldsymbol{p}_1 \tag{3.21}$$

Once these quantities are defined, one can find A, B, and E:

$$A = \frac{1}{D_0}\left(-D_{12}\frac{\tau_3}{\tau} + D_{22} + D_{32}\frac{\tau_1}{\tau}\right) \tag{3.22}$$

$$B = \frac{1}{6\,D_0}\left[D_{12}\left(\tau_3^2 - \tau^2\right)\frac{\tau_3}{\tau} + D_{32}\left(\tau^2 - \tau_1^2\right)\frac{\tau_1}{\tau}\right] \tag{3.23}$$

$$E = \boldsymbol{R}_2 \cdot \hat{\boldsymbol{\rho}}_2 \tag{3.24}$$

After a lengthy calculation well shown in Curtis (2015), we are left with an 8^{th}-degree polynomial that solves for the position vector at central time:

$$r_2^8 + a\,r_2^6 + b\,r_2^3 + c = 0 \tag{3.25}$$

where

$$a = -\left(A^2 + 2AE + R_2^2\right) \tag{3.26}$$

$$b = -2\mu B(A + E) \tag{3.27}$$

$$c = -\mu^2 B^2 \tag{3.28}$$

In order to avoid numerical issues, the polynomial is scaled by $d = \sqrt{|a|}$, so that the new coefficients are $\frac{a}{d^2}$, $\frac{b}{d^5}$, $\frac{c}{d^8}$. Due to the nature of our variable, only positive real solutions are accepted. The polynomial is solved in the computer environment by calculating the eigenvalues of its relative companion matrix. Since the polynomial has been scaled,

$$r_{2,i} = \lambda_i \cdot d \tag{3.29}$$

where i depends on the number of acceptable solutions and λ_i are the positive real eigenvalues. If more than one solution is found, they need to be analyzed: there may be a degenerate solution, a solution out of the observatory range of view or a single observation could create two or more possible solutions for the first guess of the IOD. So far, around 1000 observations have been analyzed with Gauss's algorithm by the team and they always produced one solution. However, if more than one acceptable solution is found, Equations from 3.30 to 3.34 have to be carried out for each acceptable $r_{2,i}$ so that the three position vectors are found for each solution. This would, at the end of the algorithm, give as output i solutions for a single observation.

$$f_1 = 1 - \frac{1}{2\,r_2^3}\mu\tau_1^2 \qquad g_1 = \tau_1 - \frac{1}{6}\mu\left(\frac{\tau_1}{r_2}\right)^3 \tag{3.30}$$

$$f_3 = 1 - \frac{1}{2\,r_2^3}\mu\tau_3^2 \qquad g_3 = \tau_3 - \frac{1}{6}\mu\left(\frac{\tau_3}{r_2}\right)^3 \tag{3.31}$$

Now, only ρ_1, $\rho_2\,\rho_3$ are missing to compute the position vectors:

$$\rho_1 = \frac{r_2^3\left(6D_{10}\tau - 6D_{00}\tau_3 + 6D_{20}\tau_1\right) + D_{00}\mu\tau_3^3 - D_{20}\mu\tau_1^3 - D_{00}\mu\tau^2\tau_3 + D_{20}\mu\tau^2\tau_1}{\left(D_0\mu\tau^2\tau_3 - D_0\mu\tau_3^3 + 6\,D_0\tau_3 r_2^3\right)} \tag{3.32}$$

$$\rho_2 = A + \frac{\mu B}{r_2^3} \tag{3.33}$$

$$\rho_3 = -\frac{r_2^3\left(6D_{12}\tau - 6D_{02}\tau_3 + 6D_{22}\tau_1\right) + D_{02}\mu\tau_3^3 - D_{22}\mu\tau_1^3 - D_{02}\mu\tau^2\tau_3 + D_{22}\mu\tau^2\tau_1}{D_0\tau_1\left(\mu\tau^2 - \mu\tau_1^2 + 6r_2^3\right)} \tag{3.34}$$

Having the position vectors of the observatory from SPICE, the cosine directions from the observations and the slant ranges from this algorithm, the position vectors for the object observed can be found.

3.2.2 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 3.4. The algorithm can work with both double precision and DA variables.

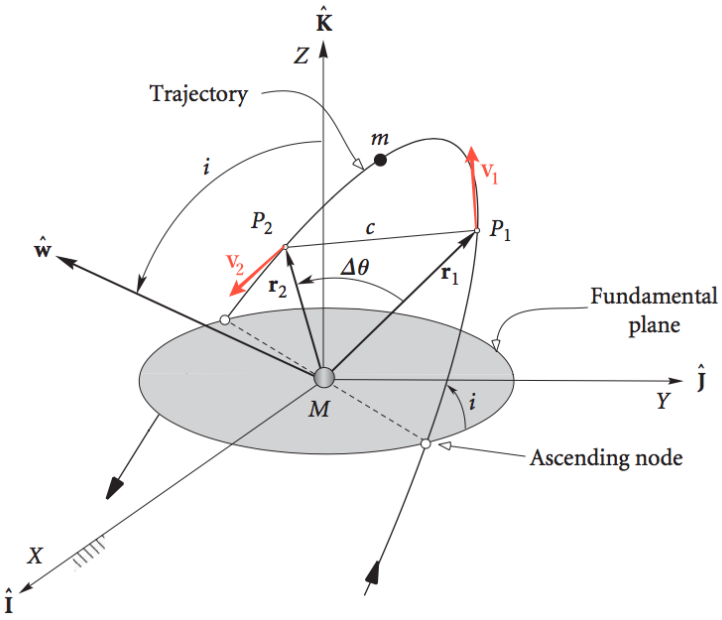


Figure 3.2: Geometry of input and output values for Lambert's algorithm. Modified from Curtis (2015).

The algorithm starts by fixing the sign of $\Delta\theta$ depending on the type of orbit: prograde or retrograde. The two equations that uniquely define $\Delta\theta$ are:

$$\begin{cases} \Delta\theta = \text{acos}\left(\dfrac{\boldsymbol{r}_1 \cdot \boldsymbol{r}_2}{r_1 r_2}\right) \\ h_Z = (\boldsymbol{r}_1 \times \boldsymbol{r}_2)_Z \end{cases} \tag{3.35}$$

Then, depending on the sign of h_Z and on the type of orbit: Figure 3.3 shows the geometry of

Table 3.1: Definition of $\Delta\theta$ depending on type of orbit

	Prograde	Retrograde
$h_Z > 0$	$\Delta\theta = \text{acos}\left(\dfrac{\boldsymbol{r}_1 \cdot \boldsymbol{r}_2}{r_1 r_2}\right)$	$\Delta\theta = 2\pi - \text{acos}\left(\dfrac{\boldsymbol{r}_1 \cdot \boldsymbol{r}_2}{r_1 r_2}\right)$
$h_Z < 0$	$\Delta\theta = 2\pi - \text{acos}\left(\dfrac{\boldsymbol{r}_1 \cdot \boldsymbol{r}_2}{r_1 r_2}\right)$	$\Delta\theta = \text{acos}\left(\dfrac{\boldsymbol{r}_1 \cdot \boldsymbol{r}_2}{r_1 r_2}\right)$

the four possible transfers. In the picture t_m stands for h_Z. Recalling the description made in Subsection 3.2.1, once $(\boldsymbol{r}_1, \boldsymbol{v}_1)$ are defined, the state of the object can be retrieved at any time through the Lagrange coefficients. This means that finding the Lagrange coefficients solves Lambert's problem (Curtis, 2015). However, it is not possible to solve the problem analytically. In the past, several methods were created, as outlined by Vallado and McClain (2001), here the C++ implementation by Izzo (2014) has been used, after updating it to be able to accept both double precision and DA variables.

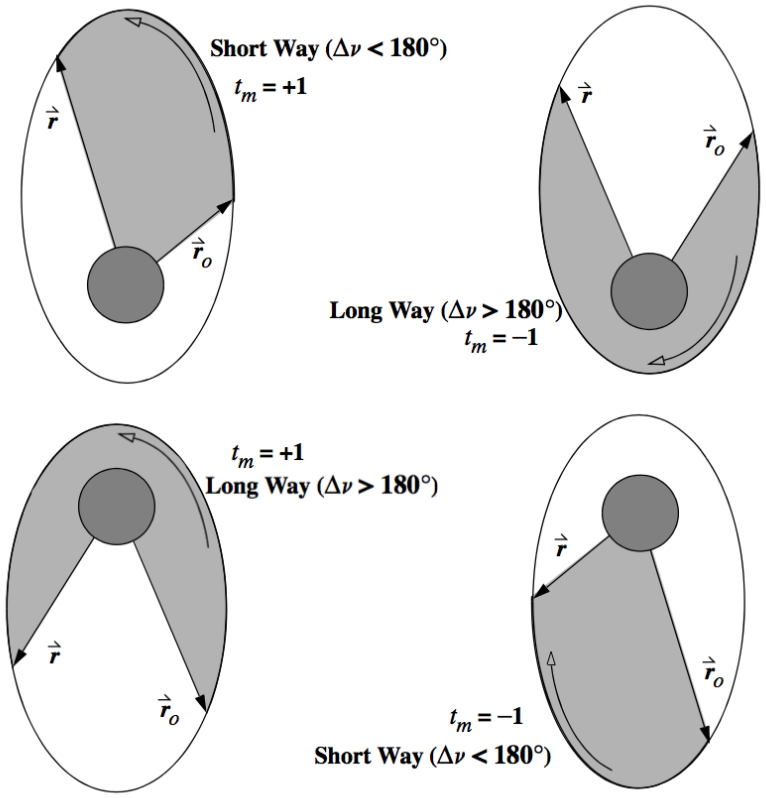


Figure 3.3: Geometry of the four possible transfers depending on type of orbit. Modified from Vallado and McClain (2001).

3.3 Mathematical definition of the initial orbit determination (IOD) algorithm

The IOD algorithm takes as input the observation of an object as defined in Section 1.1 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's algorithm (Subsection 3.2.1) in double precision. The observatory state is retrieved with SPICE (Section 2.1) and the direction cosines are found though the observation angles:

$$\hat{\boldsymbol{\rho}}_i = \begin{bmatrix} \cos\delta_i \cos\alpha_i \\ \cos\delta_i \sin\alpha_i \\ \sin\delta_i \end{bmatrix} \tag{3.36}$$

where $i = 1, 2, 3$ refer to the observation times.

At this point an estimate for the position vectors relative to the observation times 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's 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 3.4. To fix this problem

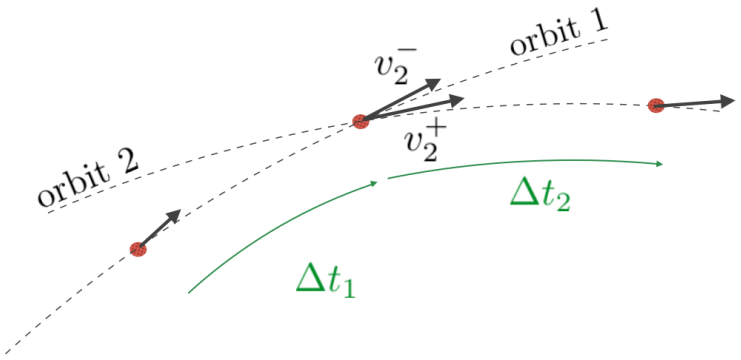


Figure 3.4: Output of Lambert algorithm taking as input the output from Gauss's algorithm

and obtain the expansion of the state, Lambert's algorithm will be used twice: the first time to modify $\boldsymbol{\rho}$ to ensure that $\boldsymbol{v}_2^- = \boldsymbol{v}_2^+$ and the second one to expand the solution $(\boldsymbol{r}_2, \boldsymbol{v}_2)$ with respect to the observation angles variations. The first step allows us to improve the estimation of the ranges made in Gauss's algorithm, while the last step is very important, because it 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 knwon values, they not free of errors: sensor accuracy, timing accuracy and observer state knowledge all influence the observation.

For the first usage of Lambert's algorithm, the values ρ_1, ρ_2, ρ_3 are initialized as DA variables. Equation 3.37 shows the mathematical and computer adapted definition:

$$\begin{cases} \rho_{1,DA} = \rho_{1,Gauss} + \delta\rho_1 \\ \rho_{2,DA} = \rho_{2,Gauss} + \delta\rho_2 \\ \rho_{3,DA} = \rho_{3,Gauss} + \delta\rho_3 \end{cases} \qquad \begin{cases} \rho_{1,DA} = \rho_{1,Gauss} + \mathrm{DA}(1) \\ \rho_{2,DA} = \rho_{2,Gauss} + \mathrm{DA}(2) \\ \rho_{3,DA} = \rho_{3,Gauss} + \mathrm{DA}(3) \end{cases} \tag{3.37}$$

The importance of counting the number of DA variables will become clear later. In this way, the output of Lambert's algorithm are the velocity functions depending on variations of the slant ranges. In particular:

$$\boldsymbol{v}_2^- = \boldsymbol{v}_2^- (\delta\rho_1, \delta\rho_2) \tag{3.38}$$

$$\boldsymbol{v}_2^+ = \boldsymbol{v}_2^+ (\delta\rho_2, \delta\rho_3) \tag{3.39}$$

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

$$\Delta\boldsymbol{v} = \boldsymbol{v}_2^+ - \boldsymbol{v}_2^- = \boldsymbol{v}\,(\delta\rho_1, \delta\rho_2, \delta\rho_3) \tag{3.40}$$

By forcing $\Delta\boldsymbol{v} = \boldsymbol{0}$ one wants to find the $\delta\boldsymbol{\rho}$ necessary to obtain it. Newton method for DA, well explained in Section 3.1.1, 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

$$\Delta\boldsymbol{v}\,(\rho_1, \rho_2, \rho_3; \delta\rho_1, \delta\rho_2, \delta\rho_3) = \boldsymbol{0} \tag{3.41}$$

one obtains:

$$\begin{cases} \rho_{1,L1} = \rho_{1,Gauss} + \Delta\rho_1 \\ \rho_{2,L1} = \rho_{2,Gauss} + \Delta\rho_2 \\ \rho_{3,L1} = \rho_{3,Gauss} + \Delta\rho_3 \end{cases} \quad \text{such that} \quad \Delta\boldsymbol{v}\,(\rho_{1,L1}, \rho_{2,L1}, \rho_{3,L1}; 0, 0, 0) = \boldsymbol{0} \tag{3.42}$$

At the end of this step, one has obtained the states of the object that satisfy the constraint of pertaining to a unique orbit. However, the solution is expanded with respect of 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. Depending on the total number of DA variables allowed, two methods are possible: the 9-variables and 6-variables method. The first one can exploit again Newton's method, while the second one does not add 3 redundant variables.

- **9-variables method.**
  This method adds 6 DA variables to the three already existing, which are re-initialized. The non-constant parts of the angle polynomials are scaled by the precision of the observation $\sigma_{P,i}$, so that the variation of the final solution will act according to the reliability of the observations. Equation 3.43 shows the mathematical and computer adapted definition:

$$\begin{cases} \rho_{1,DA} = \rho_{1,L1} + \delta\rho_1 \\ \rho_{2,DA} = \rho_{2,L1} + \delta\rho_2 \\ \rho_{3,DA} = \rho_{3,L1} + \delta\rho_3 \\ \alpha_{1,DA} = \alpha_{1,obs} + \sigma_{P,1}\,\delta\alpha_1 \\ \alpha_{2,DA} = \alpha_{2,obs} + \sigma_{P,2}\,\delta\alpha_2 \\ \alpha_{3,DA} = \alpha_{3,obs} + \sigma_{P,3}\,\delta\alpha_3 \\ \delta_{1,DA} = \delta_{1,obs} + \sigma_{P,1}\,\delta\delta_1 \\ \delta_{2,DA} = \delta_{2,obs} + \sigma_{P,2}\,\delta\delta_2 \\ \delta_{3,DA} = \delta_{3,obs} + \sigma_{P,3}\,\delta\delta_3 \end{cases} \qquad \begin{cases} \rho_{1,DA} = \rho_{1,L1} + \mathrm{DA}(1) \\ \rho_{2,DA} = \rho_{2,L1} + \mathrm{DA}(2) \\ \rho_{3,DA} = \rho_{3,L1} + \mathrm{DA}(3) \\ \alpha_{1,DA} = \alpha_{1,obs} + \sigma_{P,1}\,\mathrm{DA}(4) \\ \alpha_{2,DA} = \alpha_{2,obs} + \sigma_{P,2}\,\mathrm{DA}(5) \\ \alpha_{3,DA} = \alpha_{3,obs} + \sigma_{P,3}\,\mathrm{DA}(6) \\ \delta_{1,DA} = \delta_{1,obs} + \sigma_{P,1}\,\mathrm{DA}(7) \\ \delta_{2,DA} = \delta_{2,obs} + \sigma_{P,2}\,\mathrm{DA}(8) \\ \delta_{3,DA} = \delta_{3,obs} + \sigma_{P,3}\,\mathrm{DA}(9) \end{cases} \tag{3.43}$$

  Hence, 9 DA variables are used. This means that at the beginning of the algorithm the three states have the following dependencies:

$$\boldsymbol{x}_1 = \boldsymbol{x}_1\left(\delta\rho_1,\, \delta\alpha_1,\, \delta\delta_1\right) \tag{3.44}$$
$$\boldsymbol{x}_2 = \boldsymbol{x}_2\left(\delta\rho_2,\, \delta\alpha_2,\, \delta\delta_2\right) \tag{3.45}$$
$$\boldsymbol{x}_3 = \boldsymbol{x}_3\left(\delta\rho_3,\, \delta\alpha_3,\, \delta\delta_3\right) \tag{3.46}$$
$$\tag{3.47}$$

  After applying Lambert's method to the couples $(\boldsymbol{x}_1,\, \boldsymbol{x}_2)$ and $(\boldsymbol{x}_2,\, \boldsymbol{x}_3)$, the constant part of $\Delta\boldsymbol{v}$ is null, thanks to the previous modification of $\boldsymbol{\rho}$ and depends on all 9 variables. The task is now to find the variation in the slant ranges as a function of the variation in the observations. This is actually fairly easy, since Newton's method can be used again. Indeed, the following partial polynomial inversion is possible

$$\Delta\boldsymbol{v}\left(\delta\boldsymbol{\rho};\, \delta\boldsymbol{\alpha},\, \delta\boldsymbol{\delta}\right) = \boldsymbol{0} \quad \Rightarrow \quad \delta\boldsymbol{\rho} = \delta\boldsymbol{\rho}\left(\delta\boldsymbol{\alpha},\, \delta\boldsymbol{\delta}\right) \tag{3.48}$$

  thus obtaining:

$$\begin{cases} [\rho_{1,L2}] = \rho_{1,L1} + \delta\rho_{1,DA}\left(\delta\boldsymbol{\alpha},\, \delta\boldsymbol{\delta}\right) \\ [\rho_{2,L2}] = \rho_{2,L1} + \delta\rho_{2,DA}\left(\delta\boldsymbol{\alpha},\, \delta\boldsymbol{\delta}\right) \\ [\rho_{3,L2}] = \rho_{3,L1} + \delta\rho_{3,DA}\left(\delta\boldsymbol{\alpha},\, \delta\boldsymbol{\delta}\right) \end{cases} \tag{3.49}$$

  Although the method is straightforward and fast, at the end of the algorithm the first three DA variables are left unused. This is not a problem per se, but it wastes memory and may create instabilities further on. For this reason, the 6-variables method is introduced.

- **6-variables method.**
  This method deletes the first three DA variables and overwrites them with the 6 DA angles defined in the 9-variables method. Equation 3.50 shows the mathematical and computer adapted definition:

$$\begin{cases} \alpha_{1,DA} = \alpha_{1,obs} + \sigma_{P,1}\,\delta\alpha_1 \\ \alpha_{2,DA} = \alpha_{2,obs} + \sigma_{P,2}\,\delta\alpha_2 \\ \alpha_{3,DA} = \alpha_{3,obs} + \sigma_{P,3}\,\delta\alpha_3 \\ \delta_{1,DA} = \delta_{1,obs} + \sigma_{P,1}\,\delta\delta_1 \\ \delta_{2,DA} = \delta_{2,obs} + \sigma_{P,2}\,\delta\delta_2 \\ \delta_{3,DA} = \delta_{3,obs} + \sigma_{P,3}\,\delta\delta_3 \end{cases} \qquad \begin{cases} \alpha_{1,DA} = \alpha_{1,obs} + \sigma_{P,1}\,\mathrm{DA}(1) \\ \alpha_{2,DA} = \alpha_{2,obs} + \sigma_{P,2}\,\mathrm{DA}(2) \\ \alpha_{3,DA} = \alpha_{3,obs} + \sigma_{P,3}\,\mathrm{DA}(3) \\ \delta_{1,DA} = \delta_{1,obs} + \sigma_{P,1}\,\mathrm{DA}(4) \\ \delta_{2,DA} = \delta_{2,obs} + \sigma_{P,2}\,\mathrm{DA}(5) \\ \delta_{3,DA} = \delta_{3,obs} + \sigma_{P,3}\,\mathrm{DA}(6) \end{cases} \tag{3.50}$$

  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:

$$\Delta\boldsymbol{v}\,(\boldsymbol{\rho}) = \mathbf{0} \quad \Rightarrow \quad \boldsymbol{\rho}_{i+1} = \boldsymbol{\rho}_i - J^{-1}_{\Delta\boldsymbol{v}(\boldsymbol{\rho}_0)}\,\Delta\boldsymbol{v}(\boldsymbol{\rho}_i) \tag{3.51}$$

  Here, the assumption is made that the Jacobian does not change in the loop. The iteration is carried out until $i = \mathrm{MaxOrder}$, thus until the highest order of the DA variable is reached.

Both methods mathematically deliver the same output: the slant ranges at all three times of observation in terms of the variations of the observations. The position vectors are then obtainable with Equation 3.15, while the velocities can be calculated with one last Lambert's procedure. The only difference in the output is that with the 9-variable method, the output depends on 9 variables, of which the first three are always zero.

An important outcome of this method is that one not only obtains the point solution, but can also easily calculate variations by means of functions evaluations. The original domain of the variations is the $(\boldsymbol{\alpha},\boldsymbol{\delta})$-domain, while the range of this domain through the state function $(\boldsymbol{r}_2,\boldsymbol{v}_2)$ is called the orbit set (OS), both described in Section 1.2.

The solution, however, is not valid for any variation of the input. Indeed a TPS is only valid within a certain region, defined by the *radius of convergence*. Outside this convergence disk, the error of the polynomial approximation is bigger than a certain threshold, which means that the approximation is not good enough. This implies that one single TPS is not able to describe the entire OS and the initial $(\boldsymbol{\alpha},\boldsymbol{\delta})$-domain needs 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 Chapter 4.

Chapter 4

Automatic Domain Splitting

Section 3.3, ended by outlining the importance of analyzing 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 tool able to calculate the limits of validity of a TPS and able to split the initial domain in two or more subsets, when a single Taylor expansion is not enough to represent the DA map. The tool without much extra computation evaluates a new Taylor expansion for each of the new domains. This approach 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, since the error of the Taylor expansion grows along the trajectory due to, for example, nonlinearities in the dynamic. The efficiency of DA for uncertainties propagation is highlighted in Armellin et al. (2010).

The mathematical basis on which the ADS operates for sub-domains generation and structure is described by the concept of *manifold* introduced in Section 4.1 and 4.2. Then, the implementation of the ADS routine is described in Section 4.3. Sections 4.4, 4.5 and 4.6 describe the implementation of the ADS algorithm on domains belonging to different Euclidean spaces. The latter describes the space where the IOD algorithm is defined. Lastly, Section 4.7 introduces the ADS algorithm for propagation.

4.1 Mathematical Background and Definition

Following these considerations and looking for a new advanced method to represent a DA set, the mathematical concept of *Manifold* is used to find new tools for the work at hand. To better understand the concept, some mathematical definitions are here outlined (Wittig, 2016) .

Definition 1. The n-dimensional manifold M^n is a topological space. Each point $p \in M^n$ is contained in an open set U_p, which is a homeomorphism of an open set of the Euclidean space R^n.

Using this definition it is possible to define, in case of astrodynamics, a phase space as a 6-dimensional manifold. More detailed properties of the topological manifold are presented in many available Math text books, such as Jeffrey M.Lee (2009).

Definition 2. A *Chart* is the couple (U, φ), where $U \subset M^n$ is an open subset of the manifold and

$\varphi : U \mapsto R^k$ is a homeomorphism of U into Cartesian space.
Definition 3. An *Atlas* $\mathscr{A}$ is a collection of charts $(U_\alpha, \varphi_\alpha)$ that the U_α form a open cover of the manifold, i.e. $\bigcup_{\alpha \in A} U_\alpha = M$.

Every topological manifold admits such an atlas, and if all the φ_α are r-times continuously differentiable, we refer to the atlas as a C^r-atlas. The atlas is not unique, since many different atlases can describe the same manifold.
To summarize, a topological n-dimensional manifold can be represented by an atlas, which is composed of charts. While infinite in general, many manifolds of relevance in practice (e.g. closed manifolds) can be described by a finite atlas.
The previous definitions are similar to the process of mapping the globe: keeping in mind the projection of a part of the globe on a 2-D map, the part of the globe can be identified with a subset of the manifold and once it is defined the projection function, the globe section can be mapped on a 2-D Cartesian space. Thus, a section of the globe and its projection function constitute a *Chart* for the globe. Furthermore does not exist a projection function able to map the whole globe onto a single one Cartesian space chart if the mapping is to be homeomorphic.

4.2 Differential algebra (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 $\left(V_\alpha, {\varphi_\alpha}^{-1}\right)_{\alpha \in A}$ where ${\varphi_\alpha}^{-1} : V_\alpha \subset R^k \mapsto {\varphi_\alpha}^{-1}(V_\alpha) \subset M$.
Applying the ideas behind to the DA set representation, some analogies can be drawn. A differential algebraic vector (DAvector) associated with a fixed domain of $(-1,1)^v$ can be seen as representing an inverse chart. However, just like it is impossible to map the whole globe with only one single chart, it is usually not possible to represent a DA manifold or set with one single DAvector. Instead, an atlas of several DA vectors is used to cover the entire DA manifold
The choice for inverse charts mainly comes from the fact that it is much easier to specify a subset of Euclidean space for the inverse chart to act on than to specify a subset of an arbitrary manifold. Practically, this means that the DA manifold representation of in our case the TPS is made from DAvector mapping from Euclidean space to phase space. Thus, ${\varphi_\alpha}^{-1}$ is a w-dimensional DAvector with v variables, while the Euclidean domain for simplicity is chosen to be $(-1,1)^v$. That definition of DA Manifold concides with that given by Wittig (2016).

Definition 4. A v-dimensional DA manifold M embedded in a w-dimensional space is defined by its finite DA atlas $\mathscr{A}$ of DA charts $\left((-1,1)^v, {\varphi_\alpha}^{-1}\right)$ with ${\varphi_\alpha}^{-1} \in {}_wD_v$.

Moreover, the employment of inverse maps and the scaled Cartesian 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 to save memory.

4.2.1 The automatic domain splitting (ADS) and differential algebra (DA) manifold

As described in Chapter 3, the initial state is generated by mapping a Cartesian domain and depends on the nominal values and their uncertainties. The determination of a Cartesian domain and its maps is not unique, thus, referring to the manifold analogy, it is possible to obtain different atlas structures which represent the initial state.
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. This mapped region, the orbit set (OS), can be propagated to a chosen epoch. For the Initial state determination, the map is a function that goes from the Cartesian space to phase space, while for the state propagation the map goes from the phase space to phase space. For both the maps defined above, the convergence radius needs to be respected. Indeed, as already pointed out, the estimation of the error between the true value and the map evaluation is strictly linked to the convergence radius: only when the mapped point lies within the convergence disk it well approximates the real state. This means that the estimate and control of the error is a crucial point in our work. Since the ADS provides the estimate and control tools for the mapping error, it is briefly introduced here.Firstly, the maps are mathematically defined. For the initial state determination the maps goes from the Cartesian domain $U \in R^v$ to the whole DA manifold, which represent the initial state. Thus, the atlas representing the manifold and defined by one chart is: $\mathscr{A} = \{(U, \gamma)\}$ where $\gamma : (-1,1)^v \in R^v \mapsto \gamma((-1,1)^v) \in R^w$. The propagation map goes from the phase space R^v to R^v itself. Indeed, when propagating from x_0 to x_n, the function associated can be written as: $x_n : (-1,1)^v \in R^v \mapsto x_n((-1,1)^v) \in R^v$. For both cases, it is used the convergence estimation order by order analysis of TPS to check the estimation error of the map (for the propagation it is checked at every time-step t_n) and if this error is bigger than a fixed threshold, the ADS tool splits the initial domain U into two or more sub-domains and re-evaluates the maps over each of the new domains. In this way a new atlas that represents the initial manifold is obtained and can be written as: $\mathscr{A} = \{(U_\alpha, \gamma_\alpha)\}_{\alpha \in A} = M$ where $\bigcup_{\alpha \in A} U_\alpha = U$ and $\gamma_\alpha : U_\alpha \in R^v \mapsto \gamma_\alpha(U_\alpha) \in R^w$.

4.3 Structure and Architecture of Algorithm

The mathematical concepts of manifold and charts need an implementation for computer representation. Three main classes are here introduced to represent each component of the DA manifold:

Patch. The Patch assumes the role of the chart, and thus the scaled Cartesian domain fixed with $(-1,1)^v$ and the map onto the the phase space are stored here. To optimally store the charts, the Taylor expansion of the map and SplittingHistory are stored here, since the latter implicitly corresponds to the domain. Synthetically, this class embeds the propriety of a DAvector to store the map and the ID SplittingHistory to identify the respective domain. Lastly, some additional member functions are created within this class to evaluate the estimation error of the expansion map, to establish the direction of the split and to obtain the new Patch during the split.

SplittingHistory. As outlined before, the scaled Cartesian domain and inverse maps will be stored during the split. However, when the domain is split, one has to know where the new domain is with respect to the initial one. For this reason the SpittingHistory class is introduced: it keeps track of all the operations performed on the domain through a vector that stores integer values representing the variable direction being split. To complete the class, it is equipped with a member function that extrapolates information about the domain and can replay it after or during the splitting.

Manifold. The Manifold assumes the role of the mathematical manifold: it is described by an atlas that is the union of several Patches. When the manifold is performed by the ADS algorithm, 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 to decide when the manifold convergence analysis is acceptable or not acceptable during

the propagation. In short, the Manifold is defined as a list of Patches and embeds some member functions to analyse it, to extrapolate additional information and to *propagate* it.

Hereafter these terms will be used during the explanation of the work and to give results.
Chapter 3 already showed some useful tools provided by DA to solve the IOD problem. Starting from these results, it has been decided to build a new algorithm process of the ADS, based on the SplittingHistory, Patch and Manifold classes. 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, thus, the theory behind the error estimation function, given by Armellin et al. (2010), developed within the ADS algorithm is now explained.
Let f be a $n+1$ differentiable function $f \in C^{n+1}$ and let its Taylor expansion P_f be of n order, then the following holds for the error introduced by the expansion around the origin:

$$|f(\delta x) - P_f| \leqslant C \cdot \delta x^{n+1} \tag{4.1}$$

Now let e_r be the maximum error of the expansion P_f on the domain of radius $r > 0$, for some C that depends on the maximum size of δx. If we reduce the domain to one of radius $\frac{r}{2}$ also the error will decrease by a factor $\frac{1}{2^{n+1}}$

$$|f(\delta x) - P_f| \leqslant C \cdot \delta x^{n+1} \leqslant C \cdot \left(\frac{r}{2}\right)^{n+1} = \frac{e_r}{2^{n+1}} \tag{4.2}$$

This justified why reducing the size of the domain improves the convergence radius of the expansion to the n+1 power.
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:

$$S_i = \sum_{|\alpha|=i} |a_\alpha| \tag{4.3}$$

When variables are scaled by some sufficiently small factor, the terms of any convergent power series converge at least exponentially. When this is the case, the function surely converges for all arguments smaller than 1. We therefore consider this the convergence radius of the function and the exponential decay of the coefficients by order is the feature we use to measure convergence. An exponential fit is performed is performed to compute the parameters A and B , to match S_i to the function

$$S_i = f(i) = A \cdot exp(B\, n) \tag{4.4}$$

The fit function $f(i)$ is used to compute the value $f(n+1)$ and the size S_{n+1}. If the 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, repeating this process sufficiently often ensures that the convergence criterion (i.e. exponential decay) is eventually met.
Once the error estimation is established, the ADS algorithm can work on different domains and maps by just defining the initial DAmanifold.

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 and lastly the Manifold through the DAmanifold structure. However, this first Manifold is a trial manifold because it is represented by a mathematically incorrect atlas structure. Performing the ADS algorithm which automatically analyses the manifold and then the ADS constructs the atlas structure that represents it. An overview of ADS routine and the connection between the different structures are represented in Figure4.1 . The ADS algorithm needs the definition of a function that maps the Cartesian space onto the manifold. This function can be defined by the user but it needs to

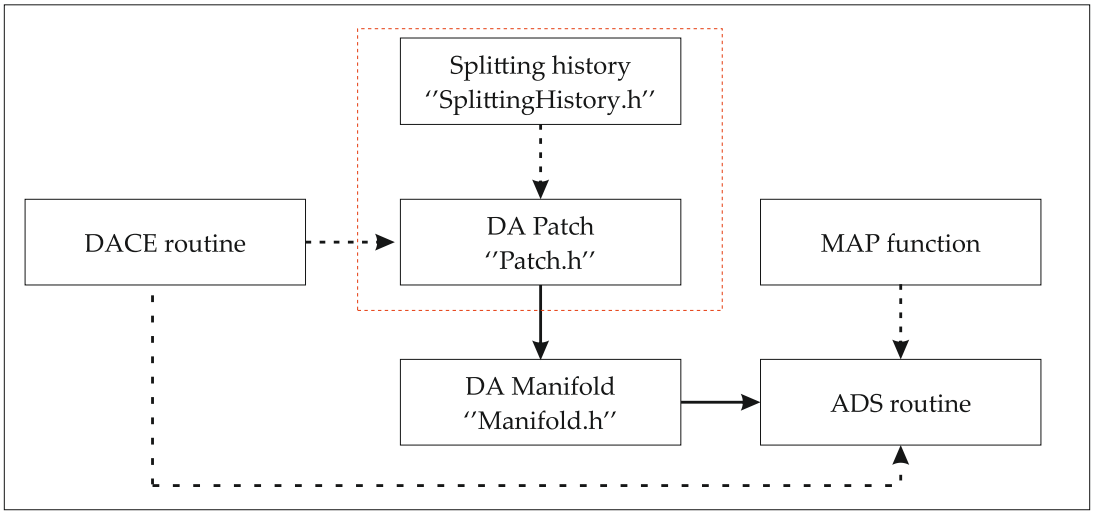


Figure 4.1: Overview Framework ADS.

have a DAvector structure, which means that the input and output must be a DAvector, for the algorithm to be able to store it in a DAPatch and then into a DAManifold.
This ADS algorithm can be used with scalar functions, multi-variable functions and ordinary differential equations (ODEs), which are described in the following sections.

4.4 Single-Variable Scalar Functions

The routine of the ADS algorithm to analyze single-variable scalar functions $f : \Re \rightarrow \Re$ is schematically illustrated in Fig.4.2 on page 28. It is very similar to the process previously described, indeed one just needs to define the initial step of the routine. The algorithm starts from the initial domain, the function to evaluate the scalar value takes as input the 1D Patch of the domain and returns a 1D Patch as output. Once the Patch of scalar function is obtained, the ADS algorithm estimates the error: if the error is below a fixed threshold, the Patch is stored into the result manifold, otherwise the domain is split. The process of the computation of the split direction gives a trivial result, that is the single variable of the domain. The two new Patch sub-domains are inserted into the trivial manifold and the process is resumed. To avoid too small subdomains and limit the runtime, also a maximum number of splits N_{max} is fixed. This means that, it can happen that a sub-domain is stored in the results Manifold when it reaches the minimum allowed size, even though it should be split due to a high error estimation.
The resulting Manifold contains the union of the Patches, which represent the evaluation of the polynomials in each of these sub-domains. This structure provides a more accurate approximation than the one achievable with a single polynomial evaluation.

4.5 Multi-variable Scalar Functions

In the case of multi-variable scalar functions $f : \Re^k \rightarrow \Re$, the ADS routine returns a 1D map and always analyzes the same component. After the computation of the error estimation it evaluates the split direction as shown in Figure 4.2. In this case the result is not trivial because there is more than one variable. The selection of the optimal split direction is crucial in terms of error reduction. This direction is selected through the procedure used in the propagation of the two-body problem by Wittig et al. (2015).

4.6 Multi-variable Vector Functions

Also for the case of multi-variable vector functions $f : \Re^k \to \Re^m$, the routine does not need any adjustments from the structure represented in Fig.4.2 since the maps are Patch elements. In the case of a vector valued functions, the map returned by the routine has more than one component, so the error estimation is carried out for each dimension. Then the component of the map with the maximum error estimation is considered and the polynomial which represents this map component is used to decide the splitting direction. The only caveat in the vector valued case is the definition of the meaning of "largest error". In practice, the quantities in 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 to turn them into comparable, non-dimensional units of error. If the maximum residual value does not exceed the tolerance, the Patch with the vector map is stored in the result manifold.
The multi-variable vector functions are very useful in our work because they allow us, for example, to analyze the conversion between two reference frames and to compute the evaluation of the initial OS determination with the DAIOD algorithm. In both cases, the function maps two algebraic structures of the same dimension $f : \Re^k \to \Re^k$. Furthermore, with a multi-variable vector function, one can also map the AR defined in Section 1.2.

4.7 Flow of ordinary differential equations (ODEs)

The routine for the propagation of ODEs is described in Figure 4.3. The routine starts by considering the state vector at the initial time $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$ and it is stored into a trial manifold as a Patch element. In the propagation case the Patch needs to store additional information about the state vector, that is the integration step t_i which identifies the propagation time as long as the tolerance and state vector size are satisfied. The state vector $\boldsymbol{x}_{(0,i)}$ is propagated until $t_\mathrm{new} = t_\mathrm{i} + t_\mathrm{step}$ using a DA-based integrator to obtain, at each step, the Taylor expansion of the solution $\boldsymbol{x}_{(\mathrm{new},i)}$ with respect to the initial conditions. At each step, the tolerance satisfaction is checked. If the tolerance is satisfied, then the integrator proceeds to the next time step; otherwise, the integrator stops, the domain corresponding to the current state $\boldsymbol{x}_{(0,i)}$ at time t_i is split, and two new state vectors are generated with the information of the last time integration t_i. They are added to the trial manifold, and the procedure restarts from the first state vector in the list. Every time a sub-domain reaches its N_max-th split or the final simulation time t_final, the set is saved in the result manifold with the corresponding truncation epoch. The integration is resumed with a new state vector in the trial manifold until the list is empty, which means that the routine is completed.

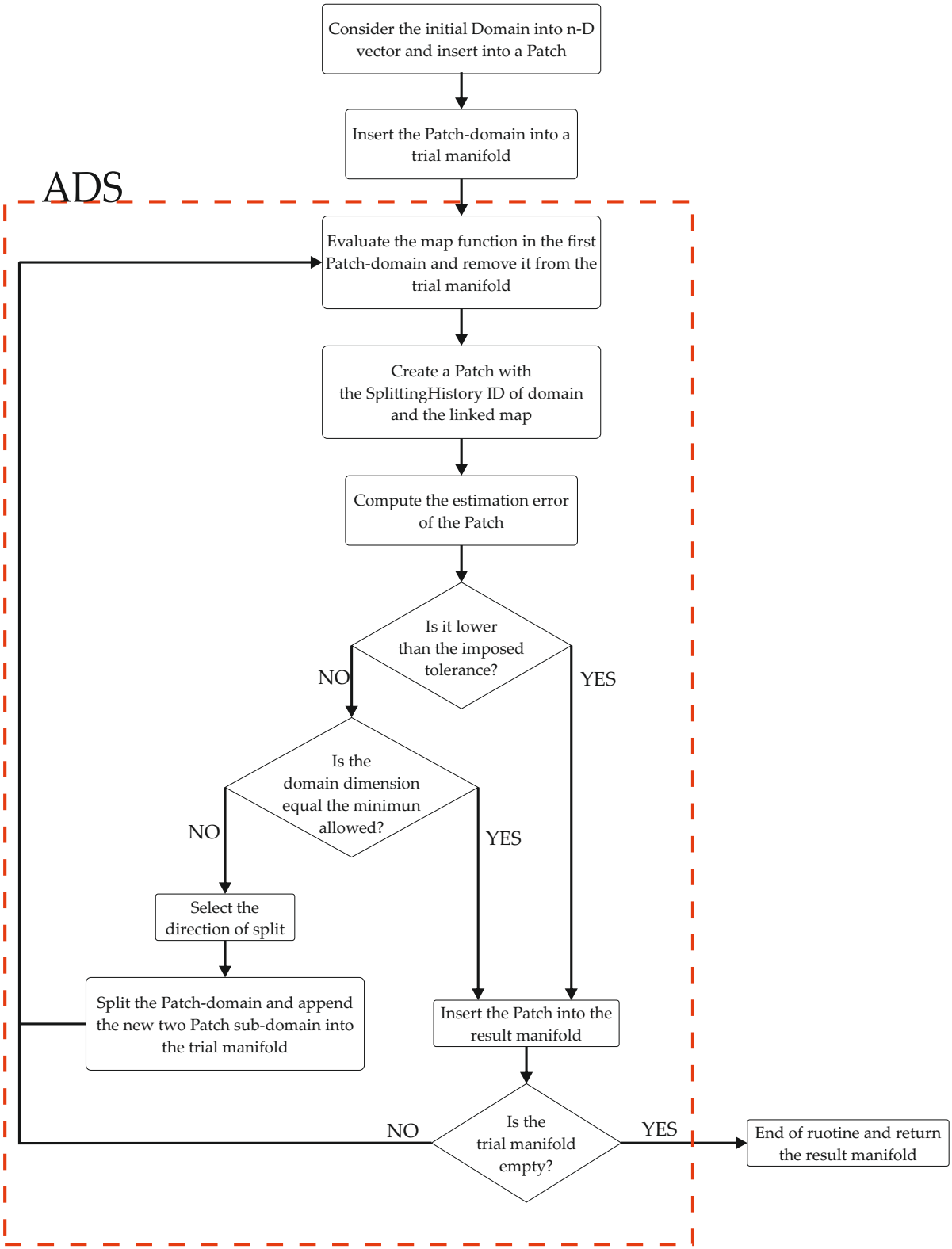


Figure 4.2: ADS routine to sampling the scalar and vector function with single and multi variable domain

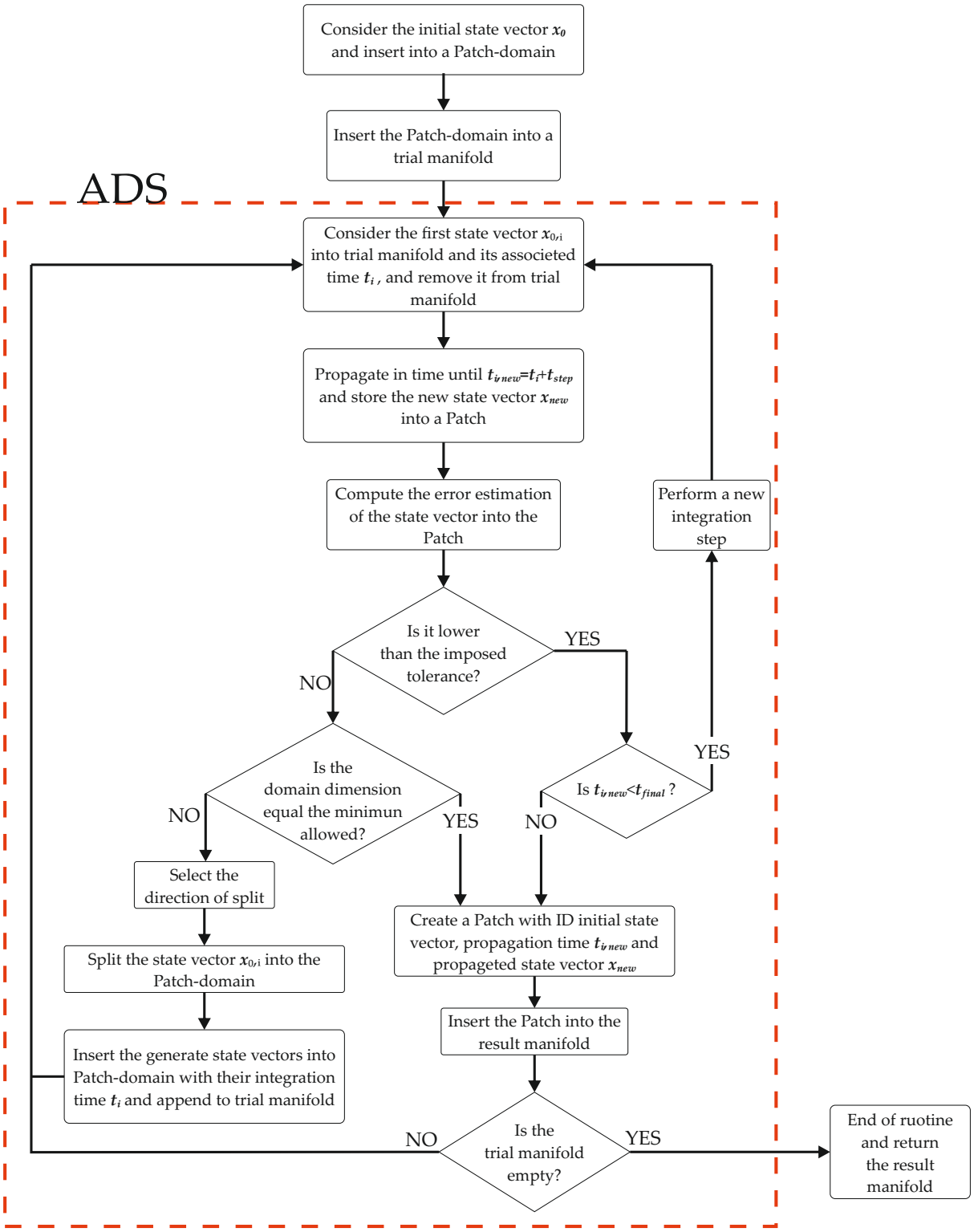


Figure 4.3: ADS routine for the Taylor expansion of the differential ordinary equation

Chapter 5

Results and Discussion

5.1 IOD: Admissible region (AR) and comparison with literature

In Section 1.2, the concepts of *attributable* and *admissible region (AR)* are 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 (Milani and Gronchi, 2009) and the current approach. Figure 5.1 shows the AR for a simulated attributable. The energy law is used to

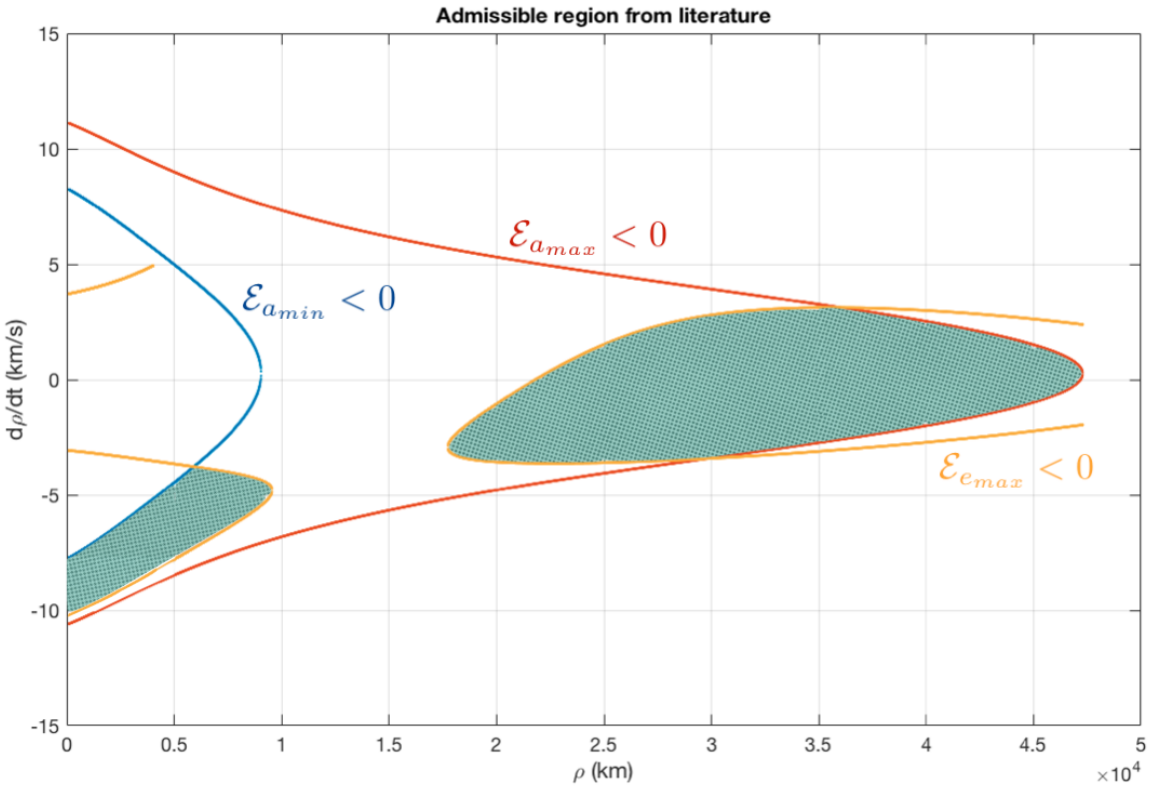


Figure 5.1: AR from literature. The green area is the CAR.

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 an implementation of the algorithm by Milani and Gronchi (2009) in Matlab. The green 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, knowing that the satellite is in a geostationary transfer orbit (GTO) would put more stringent constraints on the eccentricity and semimajor axis, thus obtaining the CAR in Figure 5.2, where the region containing circular orbits has been canceled out and only eccentric orbits remain. However, a priori knowledge is

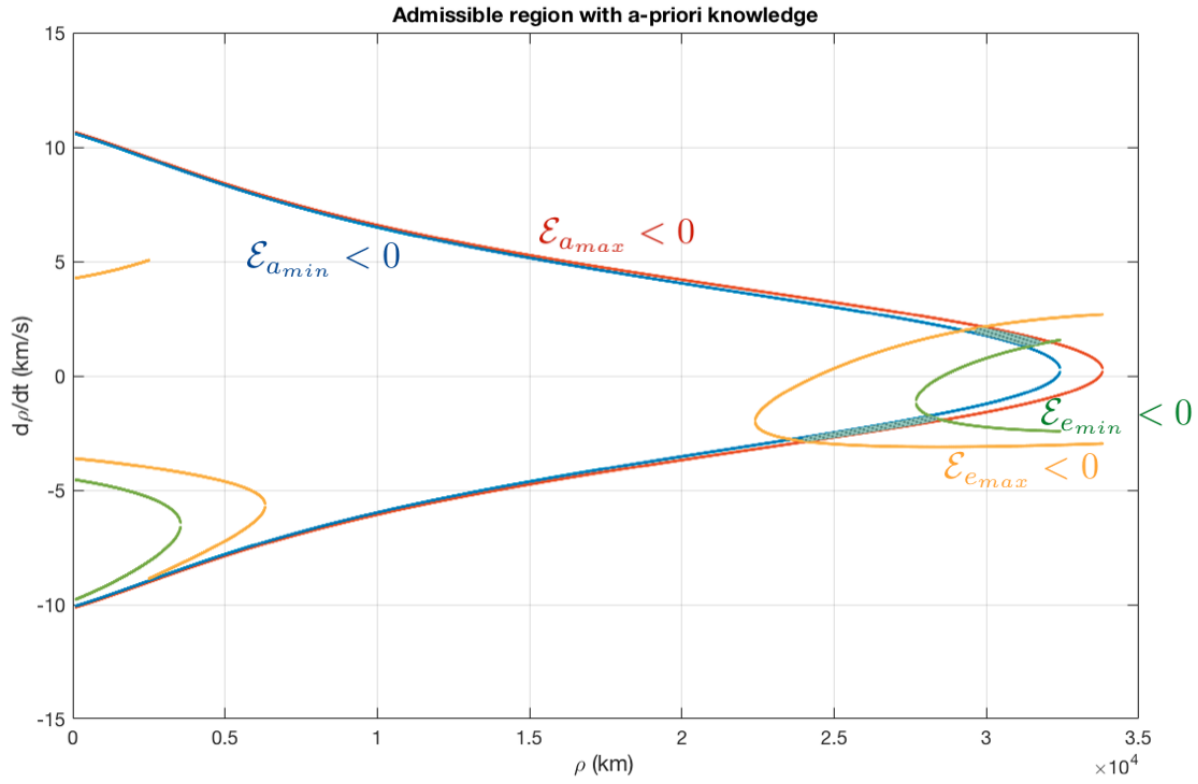


Figure 5.2: AR with a priori knowledge. The green area is the CAR.

in general not available.

The AR for the current approach is defined as the range of the functions ρ and $\dot{\rho}$ starting from the $(\boldsymbol{\alpha}, \boldsymbol{\delta})$-domain. Indeed, being the equations for the range and range-rate a function of the state, ρ and $\dot{\rho}$ also depend on the observations:

$$\begin{cases} \rho = \|\boldsymbol{r}\| = \rho\left(\boldsymbol{\alpha}, \boldsymbol{\delta}\right) \\ \dot{\rho} = \dfrac{\boldsymbol{r} \cdot \boldsymbol{v}}{\|\boldsymbol{r}\|} = \dot{\rho}\left(\boldsymbol{\alpha}, \boldsymbol{\delta}\right) \end{cases} \tag{5.1}$$

Subsection 3.1.2 described how to map the PDF of a variable $\boldsymbol{x}$, which in this case are $\delta\boldsymbol{\alpha}$ and $\delta\boldsymbol{\delta}$, into a transformed PDF through a non-linear transformation, that is Equation 5.1. The resulting box-shaped AR is shown in Figure 5.3 superimposed on the AR presented in Figure 5.1. For this simulation, high precision in the observation was assumed ($\sigma_P = 1$). It is easy to check that the real solution, that is the values used to simulate the observation, 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 HEO with a small or zero eccentricity.

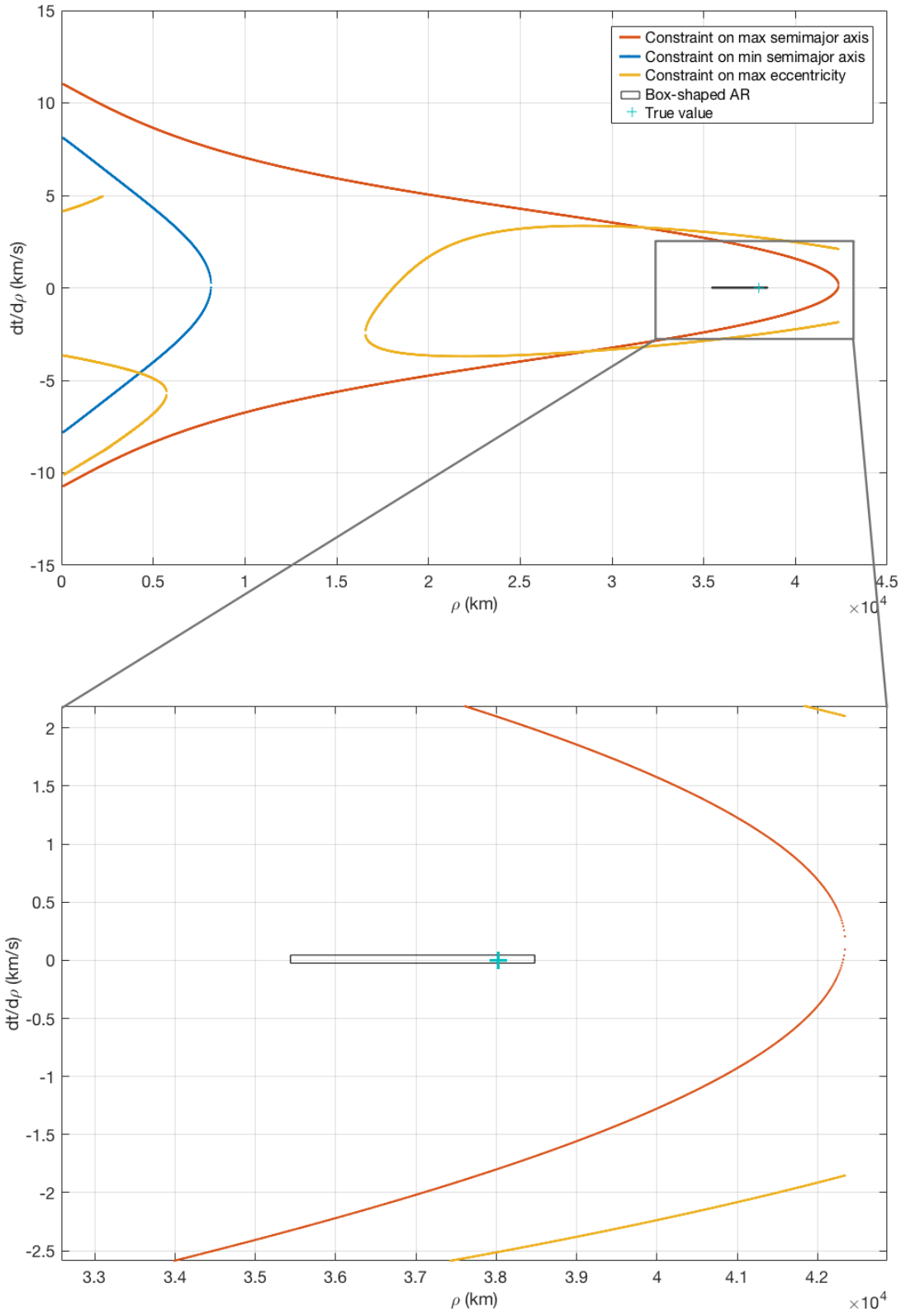


Figure 5.3: Comparison between ARs: literature vs. presented method. Accurate observation.

Figure 5.4 shows the comparison of ARs for a less accurate observation, while Figure 5.5 shows the comparison for an accurate observation of an object in a highly elliptical orbit.

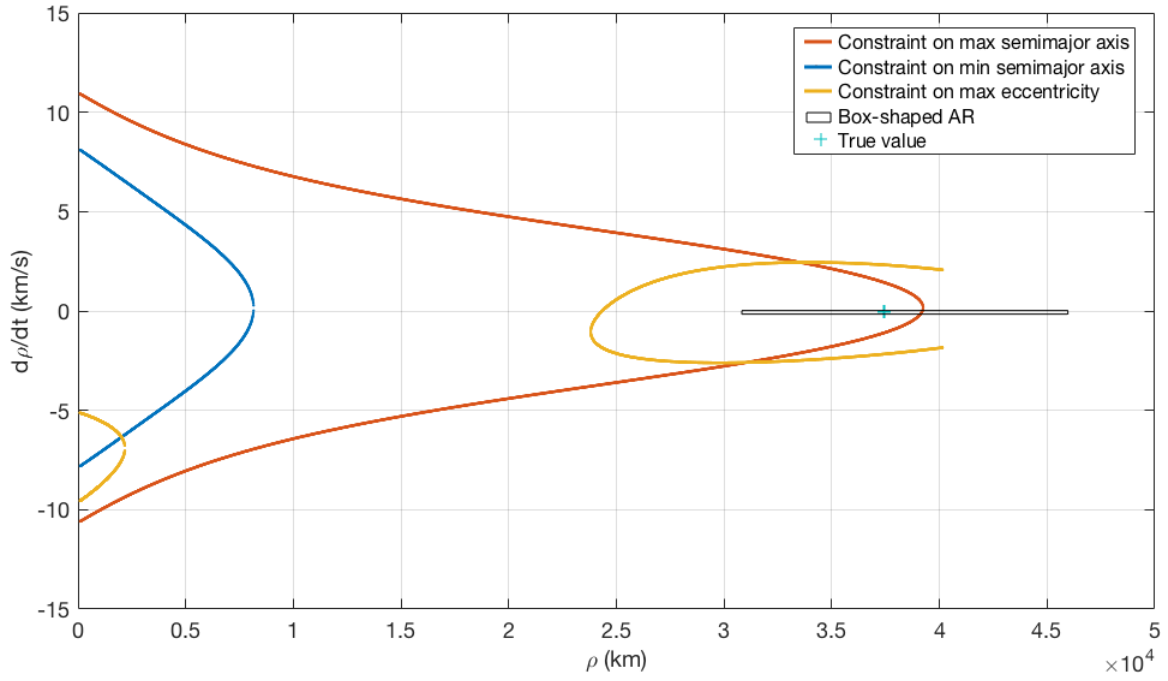


Figure 5.4: Comparison between ARs: literature vs. presented method. Inaccurate observation.

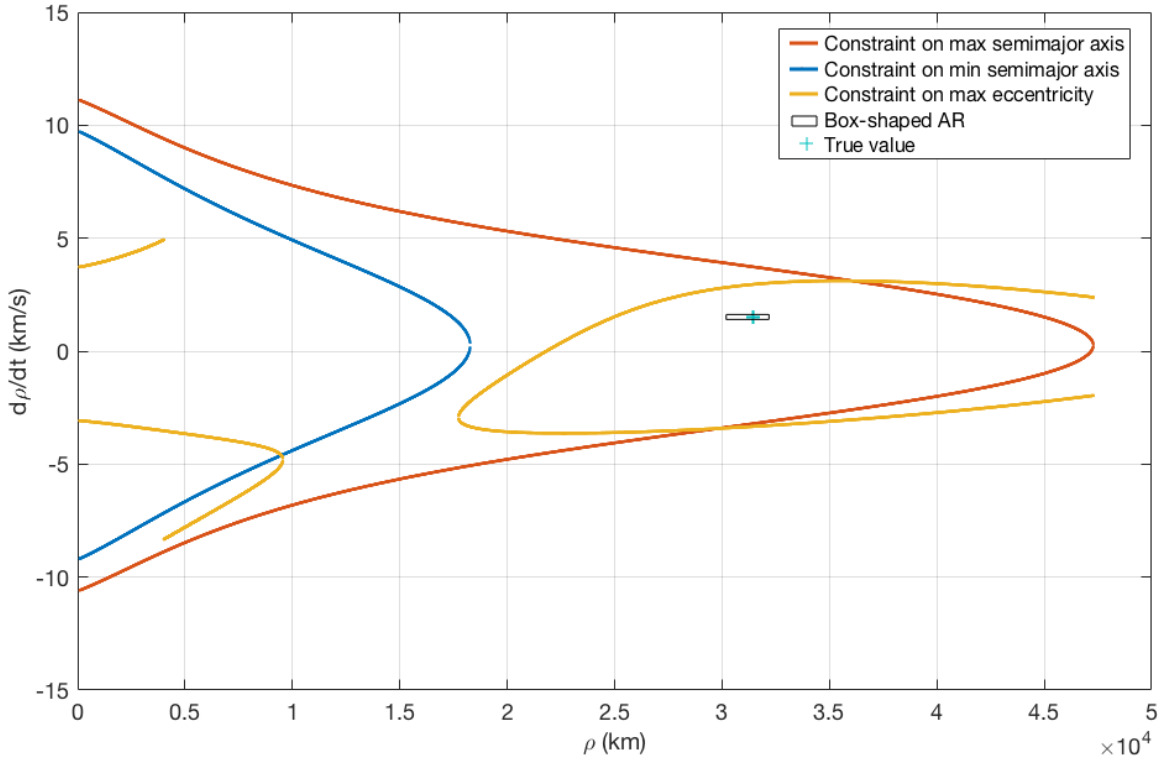


Figure 5.5: Comparison between ARs: literature vs. presented method. Highly elliptical orbit.

As already underlined in Section 3.3, one box is not accurate enough to describe the entire region. Section 5.2.3.2 will show the same ARs presented in this section split with the ADS tool.

Here, a quick check is carried out: if the map were to be valid everywhere, the error between the perturbed observation (referred to with a star α^*) and the angles retrieved with an evaluation of the perturbed state (referred to with a hat $\hat{\alpha}$) would always be small or zero. Figure 5.6 shows the procedure

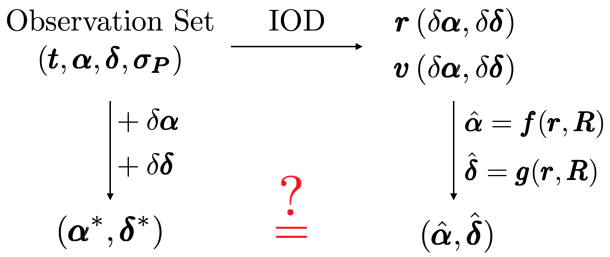


Figure 5.6: Procedure to test the accuracy of the map $(\boldsymbol{r}, \boldsymbol{v})$.

where:

$$\hat{\alpha}_2 = f\left(\boldsymbol{r}, \boldsymbol{R}\right) = \text{atan2}\left(\frac{r_2 - R_2}{\|\boldsymbol{r} - \boldsymbol{R}\|}, \frac{r_1 - R_1}{\|\boldsymbol{r} - \boldsymbol{R}\|}\right) \tag{5.2}$$

$$\hat{\delta}_2 = g\left(\boldsymbol{r}, \boldsymbol{R}\right) = \text{asin}\left(\frac{r_3 - R_3}{\|\boldsymbol{r} - \boldsymbol{R}\|}\right) \tag{5.3}$$

$\boldsymbol{r}$ is the object position, $\boldsymbol{R}$ is the observatory position and the subscripts refer to the coordinate considered. Figure 5.7 shows the contour plot of the procedure. The plots represent the region of the $(\delta\alpha_2, \delta\delta_2)$-plane for the three simulated observations already used before. As can be seen, the map is valid at 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.

5.2 Automatic Domain Splitting

5.2.1 Single-Variable Scalar Function

This section contains the first step to validate the ADS routine. A sine function has been considered to check the accuracy of the ADS routine. Using the guide line explained in Sect. 4.4, a Patch of domain $x \in [-3, 3]$ has been created, the sine function has been defined as a TPS of order 5 and the tolerance for the estimation error threshold has been fixed to 10^{-4}. Figure 5.8 shows the result for the approximation without (left) and with (right) the ADS routine.
It can be seen that 8 sub-domains are sufficient to well describe the function by respecting the maximum error.
If the inverse-square function $\frac{1}{x^2}$ is used as map function, it is possible to appreciate the behavior of ADS routine. In Figure 5.9 the inverse-square function is plot using a 5-th order TPS, 10^{-4} error tolerance and fixing the domain $[-1.88, 4.12]$ (the domain should be such that it does not include 0 but gets close, e.g. $[10^{-5}, 3]$ or so it is chosen to be asymmetric to avoid that the mean value of the domain or any of the subdomains was exactly the singular point which would lead to a failure of the algorithm as the Taylor expansion would be singular). Lastly, the maximum number of split for each sub-domain is fixed to 7, to avoid an excessive number

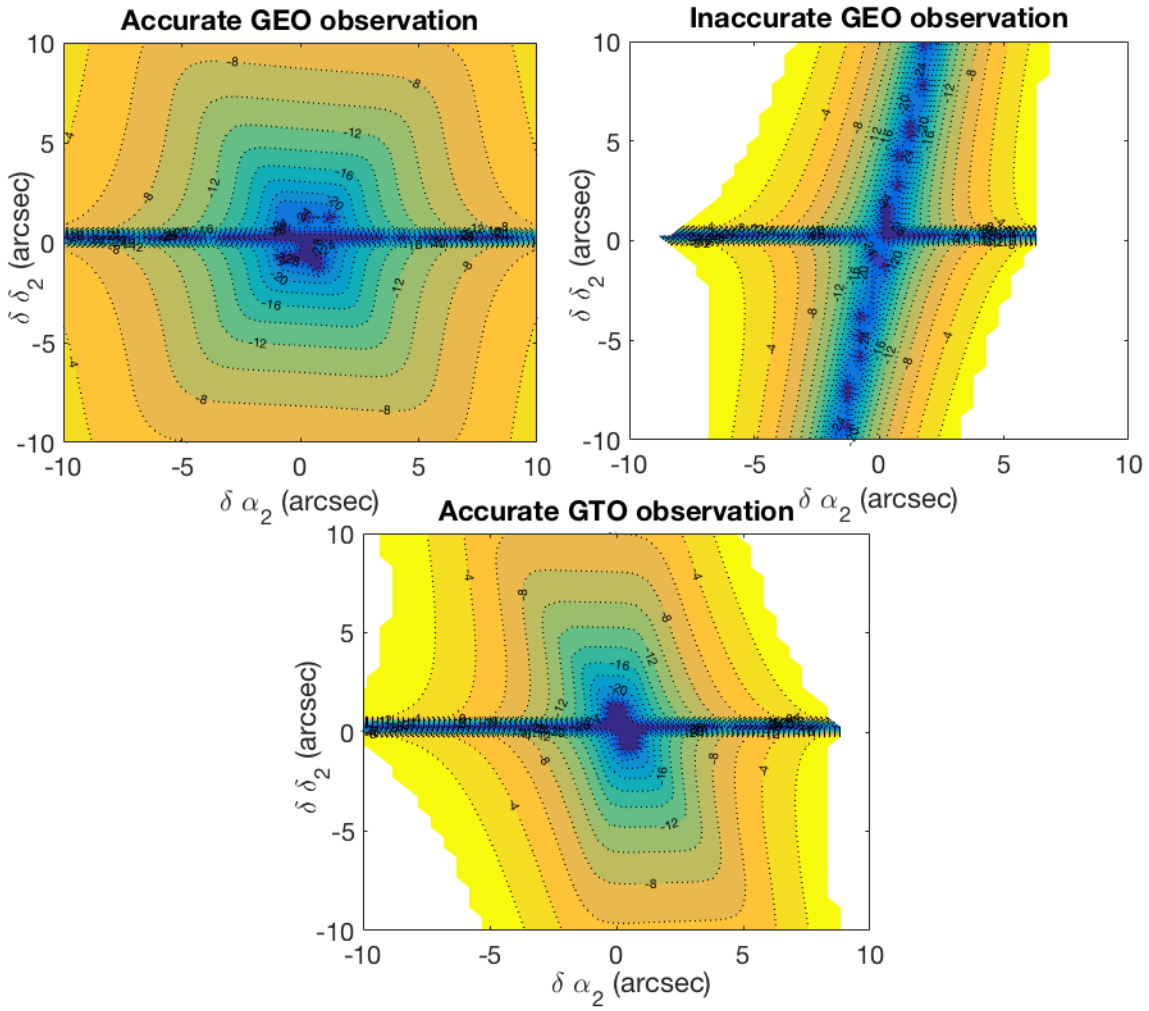


Figure 5.7: Logarithmic contour plot to check the validity of the map depending on perturbations of α_2 and δ_2. The errors are shown up to $1''$, then colors are deleted.

of sub-domains. The left side of Figure 5.9 shows the single domain expansion vs. the exact solution, demonstrating that a single expansion is not enough to map the full domain. The right side shows the inverse-square function with a Manifold of 34 Patches which increase the precision of the map. The same figure displays on the x-axis the bound of the subdomains: and that the subdomains of minimum size are close to the singular point. The reason for the presence of so many sub-domains close to the singular point is that the Taylor approximation is poor because of the singularity, thus the ADS routine keeps splitting the domain close to the singularity point until the maximum number of splits is reached.

5.2.2 Multi-Variable Scalar Function

Carrying on the validation of the ADS routine, this section focuses on its application to the multi-variable scalar function introduced in Section 4.5. The Gaussian function has been chosen to check the output of the ADS routine. The initial domain is square: $[-0.5, 1.5] \times [-0.5, 1.5]$ and the parameters of the Gaussian are $\boldsymbol{\mu} = (0.5 \;\; 0.5)$ and $\boldsymbol{\sigma}^2 = \text{diag}(0.1,\, 0.01)$. 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 5.10 the ADS routine applied to the Gauss function returns a manifold of 64

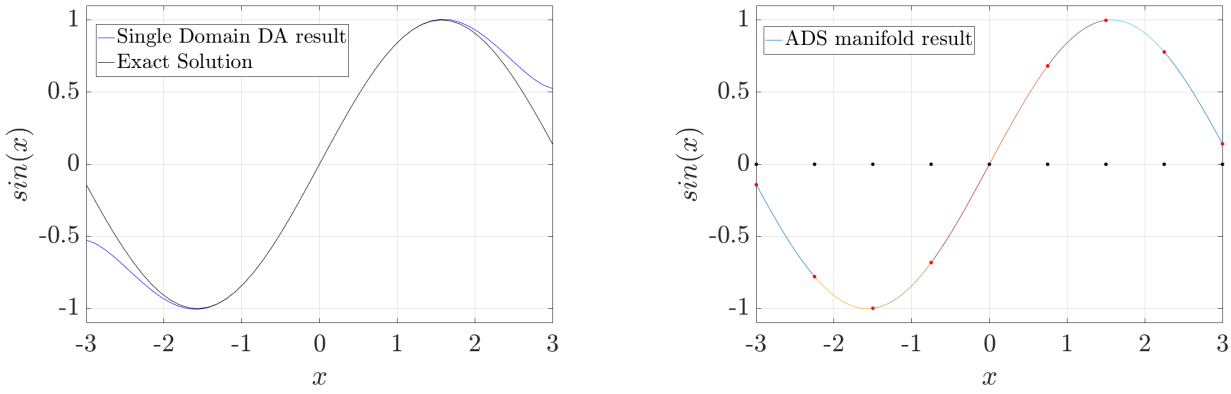


Figure 5.8: Exact solution vs. ADS Manifold results for sine function with an expansion order 5 and tolerance 10^{-4}. On the left side there is the exact solution on the single domain expansion, on the right side the manifold representation. The black dots on the x-axis define the extremes of the sub-domains.

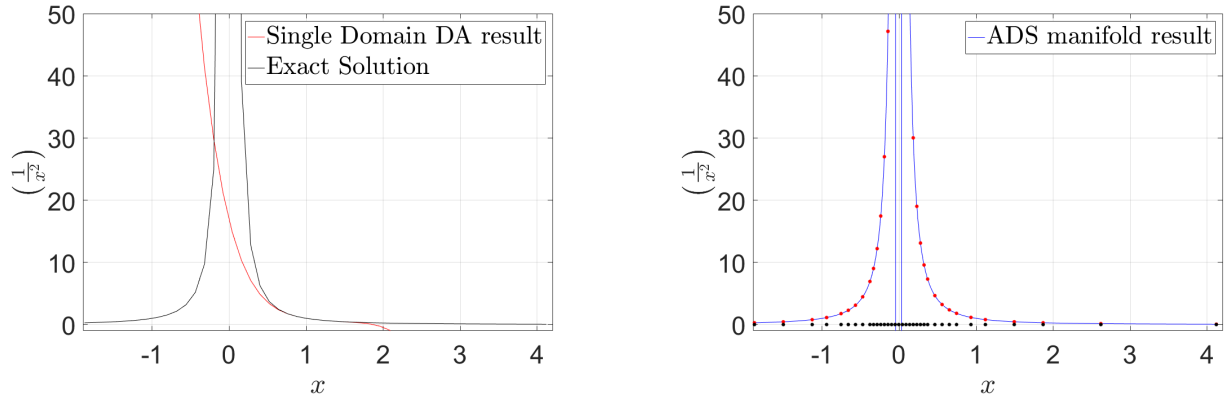


Figure 5.9: Representation of the inverse-square function. On the left, the exact solution and the single domain expansion, on the right, the manifold representation and the extremes of the obtained sub-domains

Patches. To appreciate the accuracy of the expansion, the difference between the actual values and the polynomial evaluation is shown in Figure 5.11 for different maximum number of splits. Figure 5.11(a) is the result of the evaluation with a single domain, Figure 5.11(b) was obtained with a limit of 4 splits for each sub-domain and Figure 5.11(c) is the result for the limit of 10 splits for each sub-domain. To understand the link between the number of sub-domains, the variation of expansion order and the tolerance, Figure 5.12 is shown. The results are acheived by exploiting the Gaussian function. The application of the ADS routine to the Gaussian function can be useful the describe the Patch structure and its similarity with the chart of the manifold. In Figure 5.13 one element of the Patch is considered and is mapped into the DA Manifold. The Patch has a chart structure so it must contain the domain and its map onto the manifold. The domain is defined by the SplittingHistory which stores the *story* of the domain splits and function that maps the sub-domain onto the manifold.
In this way it is possible to have a complete and lightweight structure of the manifold with all the information about the resulting domains and maps after the ADS routine.

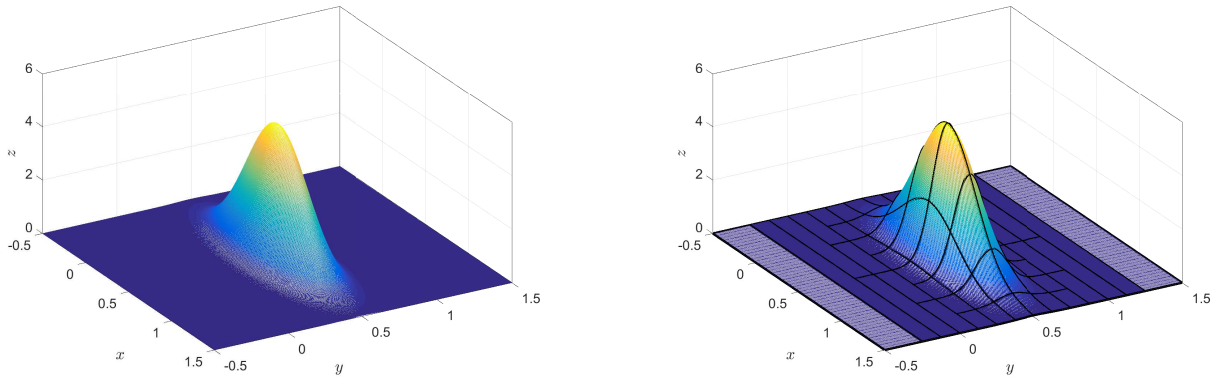
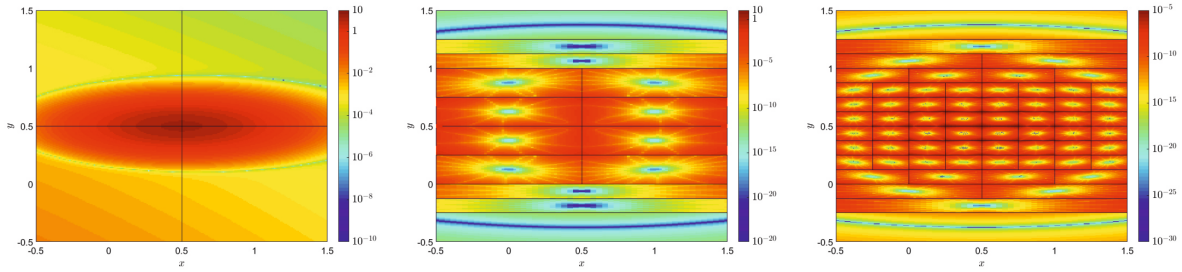


Figure 5.10: Gauss function exact solution(left) vs. manifold representation(right)



(a) Single DA Domain expansion (b) 14 DA sub-Domain expansion (b) 64 DA sub-Domain expansion

Figure 5.11: The error between the real values and the polynomial approximation is computed for the single domain expansion (a), the manifold of 14 sub-domains (b) and a manifold with 64 sub-domains (c)

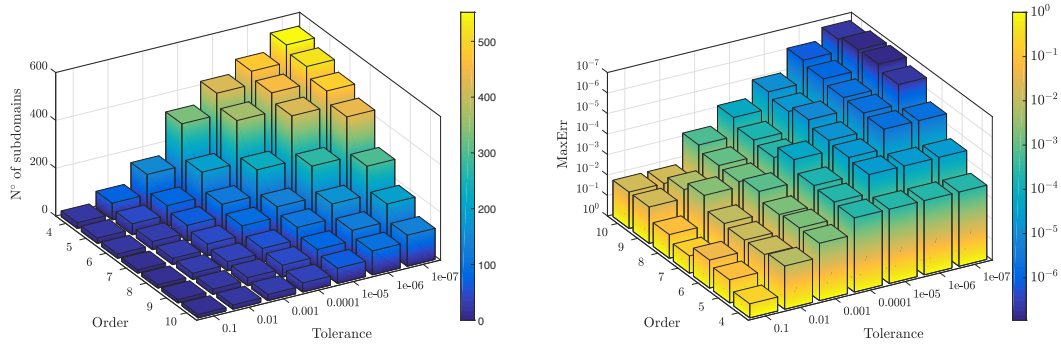


Figure 5.12: Link between number of subdomains, order and tolerance for Gaussian function

5.2.3 Multi Variable Vector Function

Recalling the definition in Section 4.6, these are functions of the type $f : \Re^k \rightarrow \Re^m$. Recalling the goal of this work, two main test cases are here analyzed: the conversion of orbital state representation and the DAIOD function.

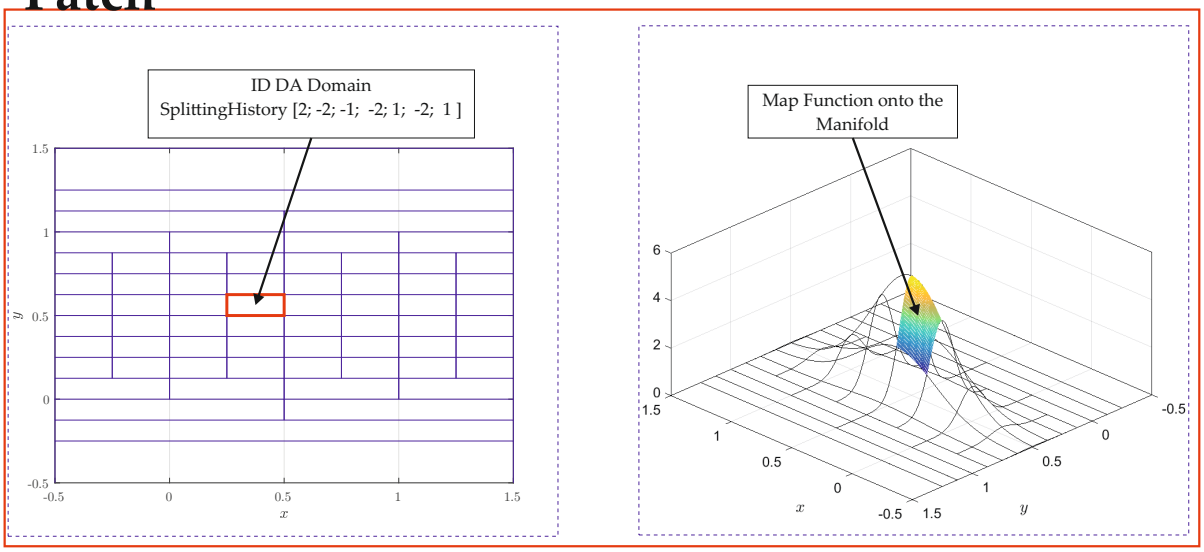


Figure 5.13: Structure of the Patch and its components

5.2.3.1 Conversion of orbital state representation

In Astrodynamics, sometime one state representation better suits some situations than another and thus a conversion function is needed. It is useful to create these functions in the DA environment in order to use them in a DA domain representation to obtain the same DA domain into another representation. The algorithms for these transformations contain mathematical operations and non-linear transformations. The highly non-linear transformations which do not converge well when expanded around a single point could generate a resulting excessive error of the Taylor expansion. If the conversion function is used as map function in the ADS routine, one can convert the domain and set a threshold for the maximum error. In this work the following conversions are implemented:

$$\begin{pmatrix} a \\ e \\ i \\ \Omega \\ \omega \\ \nu \end{pmatrix} \leftrightarrow \begin{pmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \end{pmatrix} \qquad \begin{pmatrix} p \\ f \\ g \\ h \\ k \\ L \end{pmatrix} \leftrightarrow \begin{pmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \end{pmatrix} \qquad \begin{pmatrix} p \\ f \\ g \\ h \\ k \\ L \end{pmatrix} \leftrightarrow \begin{pmatrix} a \\ e \\ i \\ \Omega \\ \omega \\ \nu \end{pmatrix}$$

where $(\, a, e, i, \Omega, \omega, \nu \,)$ are the KEP, $(\, p, f, g, h, f, L \,)$ are the Modified Equinoctial Element (MEE) and $(\, x, y, z, v_x, v_y, v_z \,)$ are the Cartesian coordinates in ECI reference frame. Details about the state representations and the conversions are discussed in more detail by Battin (1999), Curtis (2015) and Bate et al. (1971). They have been adapted to the DA environment such that they map from $\Re^6$ to $\Re^6$.

Furthermore the different state representations could have one or more singularity points where the elements are not defined. Each of state representations can be expressed as a vector of 6 DA variable and mapped onto the desired state representation to obtain the associated vector of 6 DA variables, each one containing the n-th order Taylor expansion of the conversion. To check the conversion error for singularity points, the ADS tools analyses the convergence of the maps and manages the error during the conversion.

As an example, a LEO object is considered and its orbital elements defined. The ECI was found with a double precision transformation of the KEPs. As shown in Table 5.1, the object

Table 5.1: Definition of the phase space vector and orbital elements

ECI	μ	σ	KEP	μ
x	6578 km		a	8038.9 km
y	0 km		e	0.2
z	0 km		i	0.1 deg
v_x	-0.719 km/s		Ω	0 deg
v_y	8.432 km/s	0.2	ω	30 deg
v_z	0.015 km/s	0.2	ν	330 deg

considered lies close to the equatorial plane and its ascending node belongs to the x-axis of ECI reference frame. Without loss of generality, only v_y and v_z are considered as DA variables.After choosing the 2-nd order expansion, 10^{-5} order threshold and maximum 15 splits, the resulting sub-domains obtained during the conversion are shown in Figure 5.14.

The fundamental result is that the ADS executes a big number of splits close to $v_z = 0$, thus

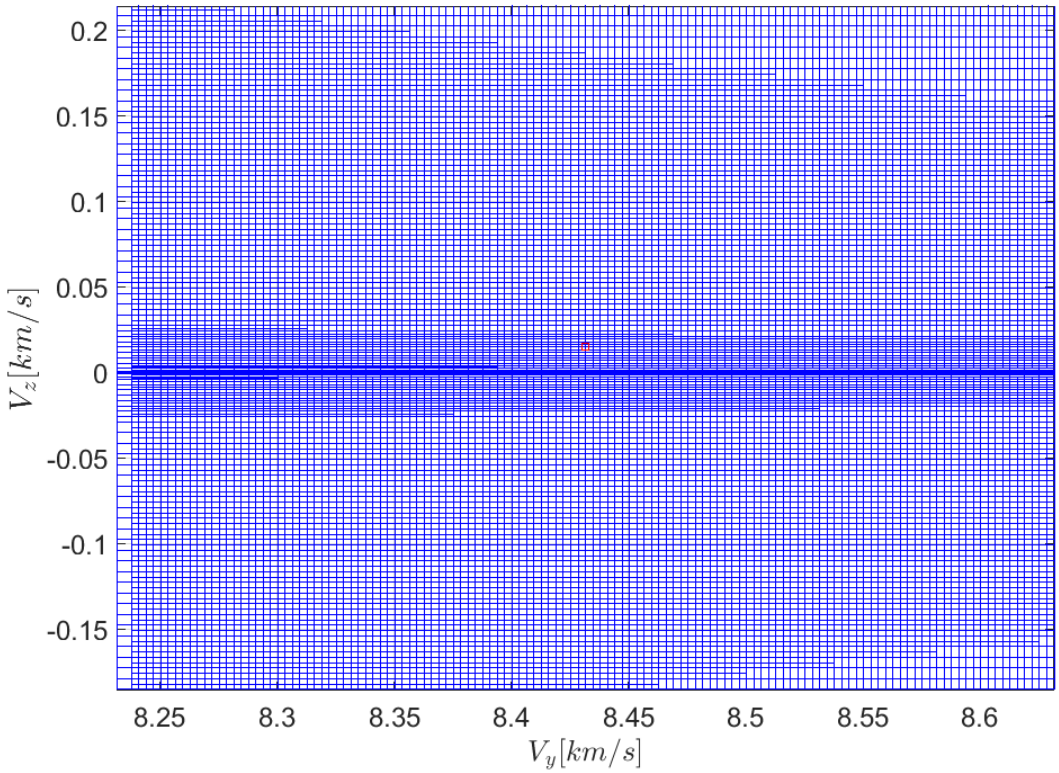


Figure 5.14: Resulting phase space domains on the v_y-v_z plane for the conversion from ECI to KEP, the red box is the nominal value

when the orbit becomes equatorial and some keplerian elements are not defined. Due to the singularity is not possible to achieve the conversion for the value $v_z = 0$ but the ADS reaches a good approximation by increasing the number of Patches close to the undetermined points. To deduce supplemental results, let's consider a single perturbation of the state vector along the v_z direction and a 2-nd order expansion. The trivial variation of v_z could cause a modification of the eccentricity of the resulting orbit but as it can be seen in the left side of Figure 5.16 and 5.18 that variation is small.

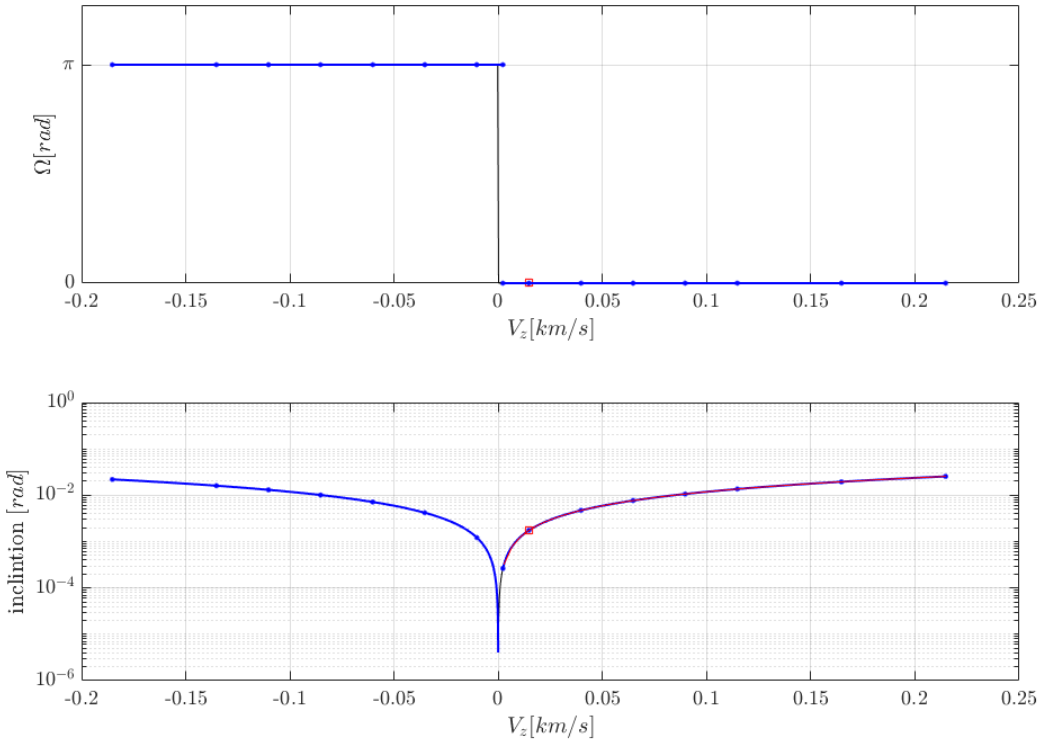


Figure 5.15: Right ascension(upper) and inclination(bottom) with respect to the perturbation of v_z around its mean value(red square) and error threshold 10^{-2}

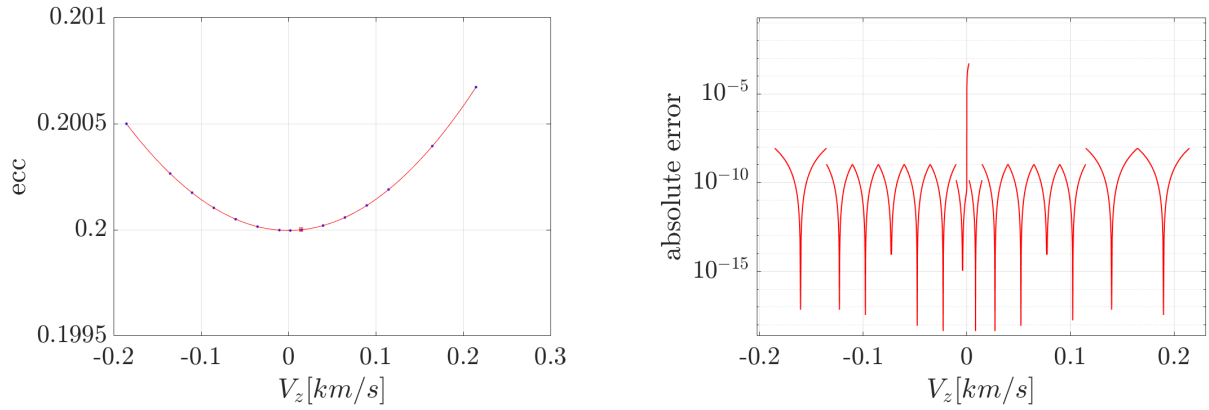


Figure 5.16: Kepler element eccentricity(left) and absolute error(right) with respect to the perturbation of v_z around its mean value(red square) and error threshold 10^{-2}

In Figure 5.15 and 5.17, regardless of the error tolerance chosen for the ADS, the presence of a singular point in the neighborhood of the mean value leads to the indetermination of the right ascension. By reducing the error tolerance the Taylor expansion precision increases as shown on the right side of Figure 5.16 and 5.18. The absolute error is computed at the extremes of the domain space between the actual values and the evaluation of the KEP map. The reduction of the tolerance produces a decrease in error magnitude. Regardless of the reduction, the singular point corresponding to $v_z = 0$ does not allow the Taylor expansion to approximate well enough the Keplerian element, thus next to this point there is a peak of the absolute error which produces a

consecutive split of the domains. The domains of minimum size are located in the neighborhood of the singular point as shown on the left side of Figure 5.16 and 5.18.

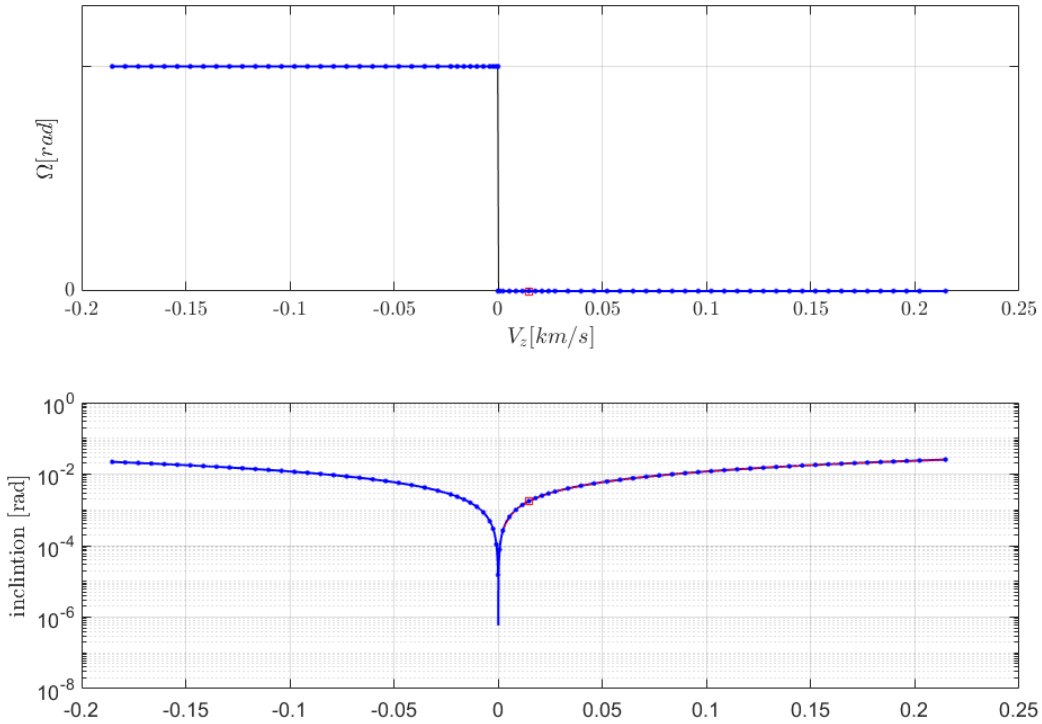


Figure 5.17: Right ascension(upper) and inclination(bottom) with respect to the perturbation of v_z around its mean value(red square) and error threshold 10^{-4}

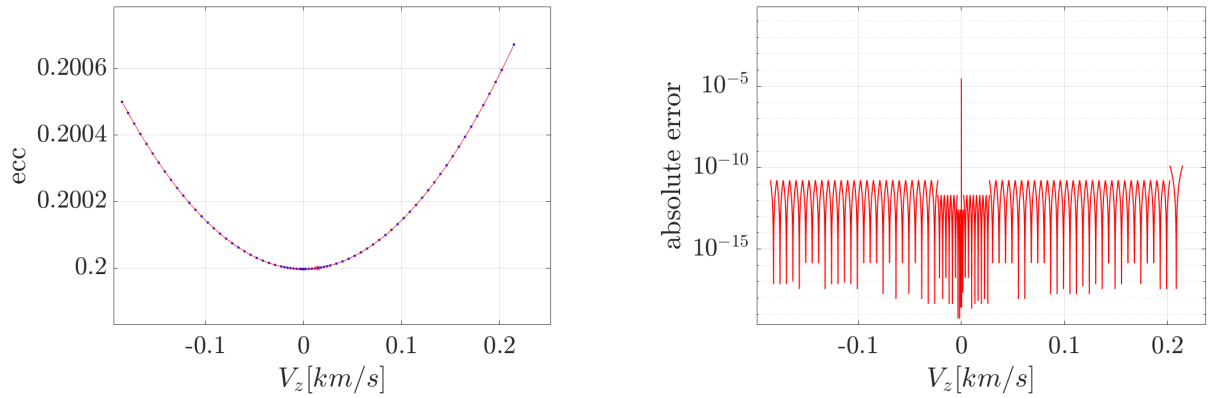


Figure 5.18: Kepler element eccentricity(left) and absolute error(right) with respect to the perturbation of v_z around its mean value(red square) and error threshold 10^{-4}

5.2.3.2 Initial orbit determination (IOD) and Automatic domain splitting (ADS)

In Section 5.2.3 another multi-variable vector function of interest was introduced, which correspond to the solution of the Initial orbit determination (IOD). In this situation, the map is the IOD algorithm itself, defined in Section 3.3. The IOD 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 input is detailed in Section 1.1. The map is a function $f : \boldsymbol{x} \in \mathfrak{R}_1^6 \mapsto f(\boldsymbol{x}) \in \mathfrak{R}_n^6$, with the subscript being the expansion order of the TPS and it can be analysed by the ADS to control the error. The ADS needs to be slightly adjusted to be able to get as input the IOD algorithm and all the parameters needed for it: indeed, a structure parameter is also passed to the routine. It contains the angular observations $(\alpha_1, \delta_1, \alpha_2, \delta_2, \alpha_3, \delta_3)$, the three observation times, the three observation precisions $(\sigma_{P,1}, \sigma_{P,2}, \sigma_{P,3})$ and the observatory position at the three observation times obtained by using SPICE as described in Section 2.1.
The ADS routine with the multi-variable vector function can be validated by choosing the orbit of a real object and using the VO to generate the observation. Given the coordinates of a satellite in term of its TLE (see Table 5.2), six simulated Observation Sets were generated with the VO, each with a different time interval between the single observations.
As a first step, a 6-th order expansions was performed and the sensitivity of the results to the

Table 5.2: Satellite TLE

```
0 CADRE
1 41475U 98067HV  16172.13123003  .00037064  00000−0  49179−3 0  9994
2 41475  51.6443  42.8436 0003066 161.2134 270.8982 15.57867415  5398
```

error tolerance was analysed. In Figure 5.19 the absolute errors for the different Observation Sets are shown. It is evident that the set with a longer time interval between two consecutive observations generate a TPS object state with higher precision. In Figure 5.20 the number of sub-domains generated by the ADS is shown and it can be seen that the number of sub-domains starts increasing at relatively low tolerances, i.e. when the absolute error starts decreasing (see Figure 5.19).
In a second analysis, the error tolerance is fixed to 10^{-5} and the expansion order varied. Figure 5.21 shows the variation of the absolute error. It is clear that the error decreases by raising the order but it is also possible to see in Figure 5.22 that the lowest expansion order is not accurate enough to be able to estimate the error, thus the ADS does not split the initial domain, even though the absolute error is beyond the tolerance.

5.2.3.3 Admissible region (AR) and automatic domain splitting (ADS)

The ADS can also be used to increase the precision of the AR obtained in Section 5.1. Now the first two ARs obtained are re-evaluated with the ADS routine: the accurate GEO observation and the inaccurate GEO observation. The ADS routine is started with a 3-rd order expansion for the TPS. Figures 5.23 and 5.24 refer to the accurate observation while Figures 5.25 and 5.26 refer to the inaccurate observation. The black box on top of Figures 5.23 and 5.25 is the box-shaped AR obtained in Section 5.1, while the boxes shown in Figures 5.24 and 5.26 are the ARs found for each of the sub-domains created by the ADS routine. The bottom plot of each figure represents the evaluation of the ARs: the red dot is the TPS evaluation while the blue triangles are the pointwise values for the range and range-rate, which are computed by solving the IOD algorithm in the variations $\delta\alpha$ and $\delta\delta$. Comparing these values it is possible to see that

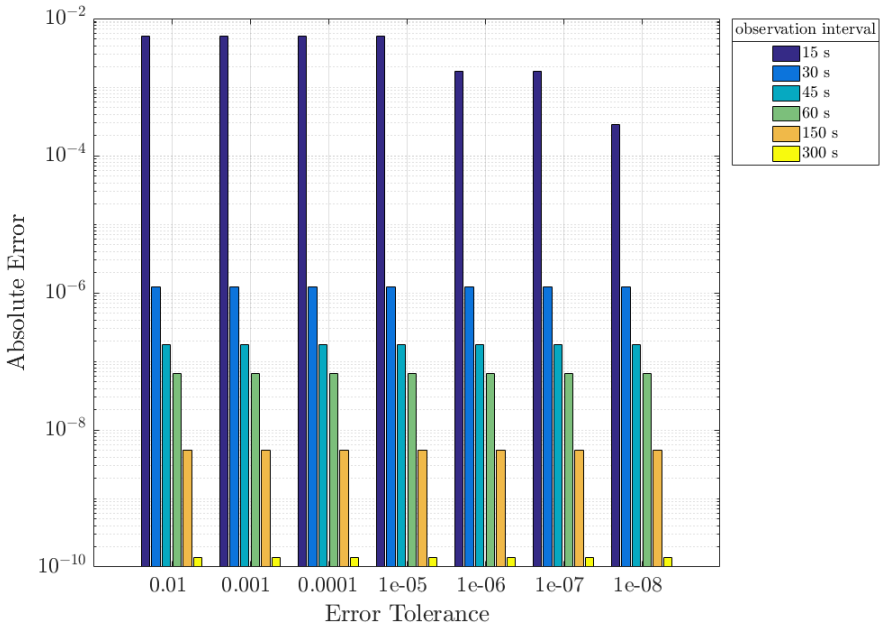


Figure 5.19: Absolute Error for the set of observations with different time interval, 6-th order expansion and varying error tolerance

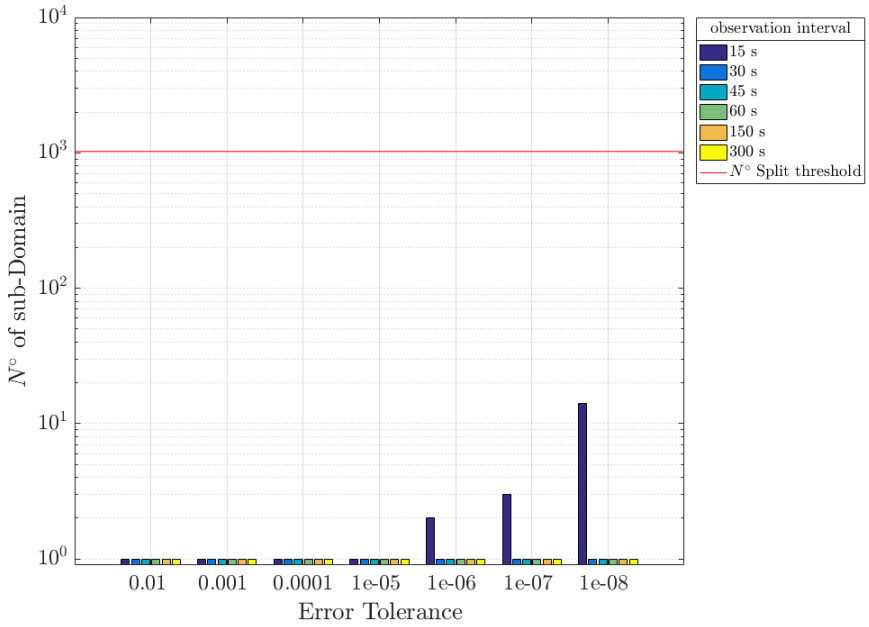


Figure 5.20: Number of sub-domains for the set of observations with different time interval, 6-th order expansion and varying error tolerance

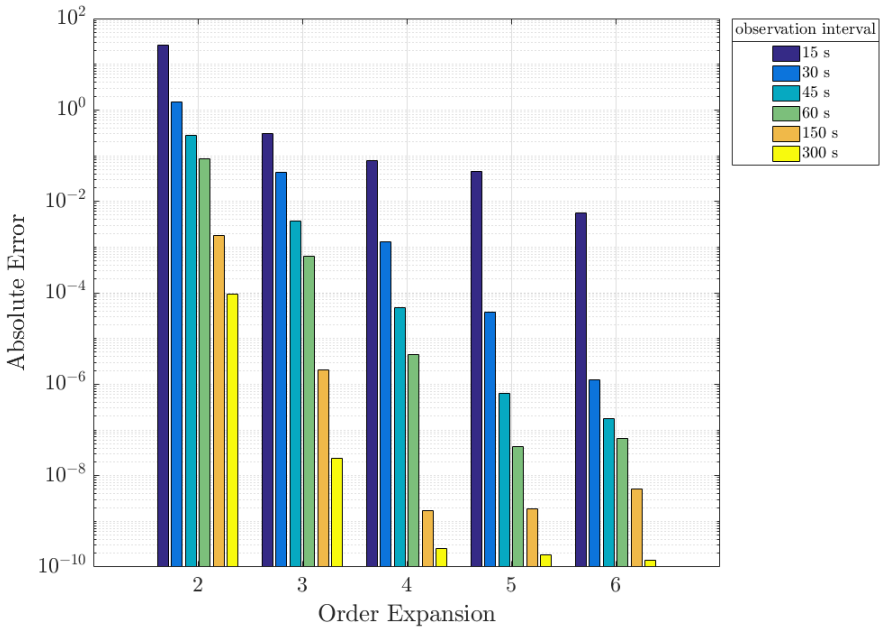


Figure 5.21: Results of Absolute Error for the set of observation with different time interval, 10^{-5} error tolerance and varying expansion order

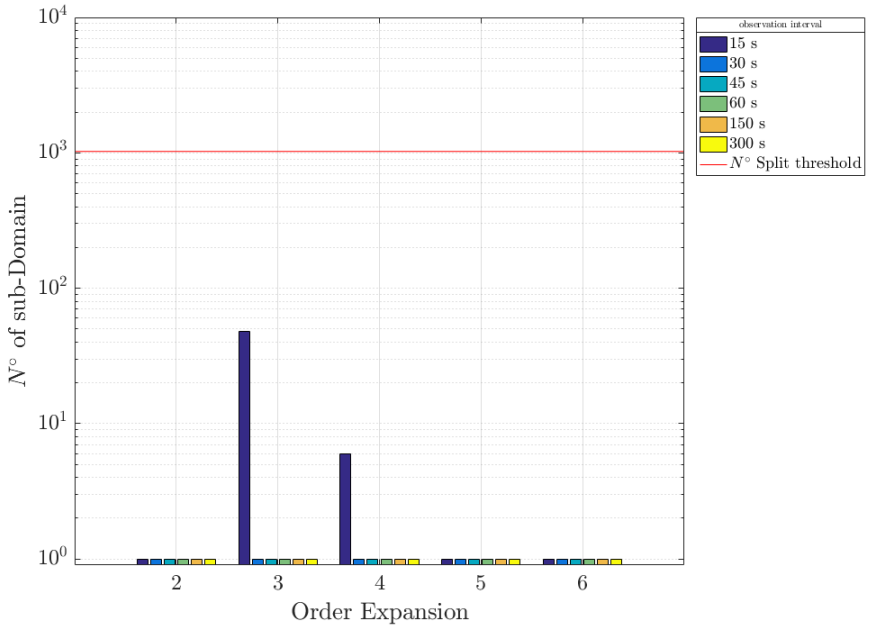


Figure 5.22: Number of sub-domains for the set of observations with different time interval, 10^{-5} error tolerance and varying expansion order

the differences between the actual value and the single domain evaluation one are large, indeed the ADS splits the initial domain and it generates a manifold, which then well describes the AR, given the similarity of the results. Furthermore this result allows to further reduce the size the first AR obtained in Section 5.1, which was already significantly smaller than the AR computed from the literature.

At the end the analysis of the IOD performed with the ADS it can be noted that the sub-domain bounds for the accurate observation are uniform within the AR obtained. This result is connected to the orbital elements of the OS space shown in Figure 5.27: since they are far from the singular point, the ADS routine works to sample the AR and then checks the error to achieve higher precision with the IOD algorithm quite smoothly. On the other hand, the inaccurate observation places the point solution close to the singular elements, as it can be seen for the eccentricity in Figure 5.28. Hence the ADS routine needs to check the error in the TPS but also has to manage the impossibility to determine the OS in the singular point. To support this assertion, it is possible to see the large number of sub-domains close to the value of ρ that corresponds to the singularity on e and ω (compared Figure 5.26 with Figure 5.28).

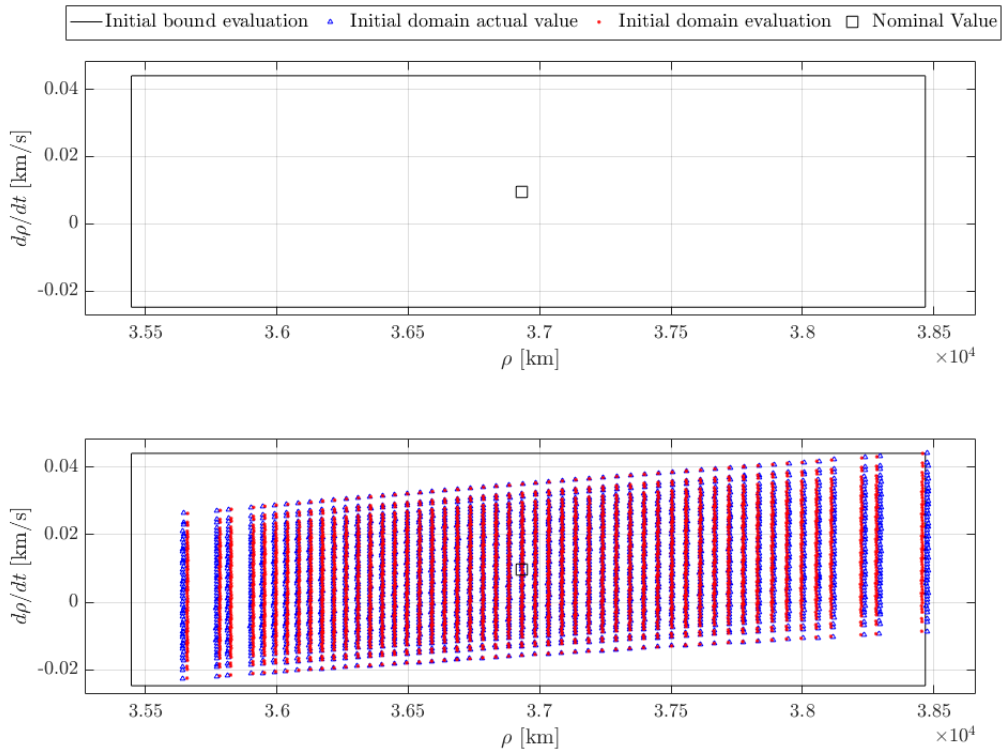


Figure 5.23: AR and ADS results, single domain evaluation vs. actual values. The big black box-shape is the AR, the black square is the mean value of the AR, the red dots are the single domain evaluations and the blue triangles are the actual object state.

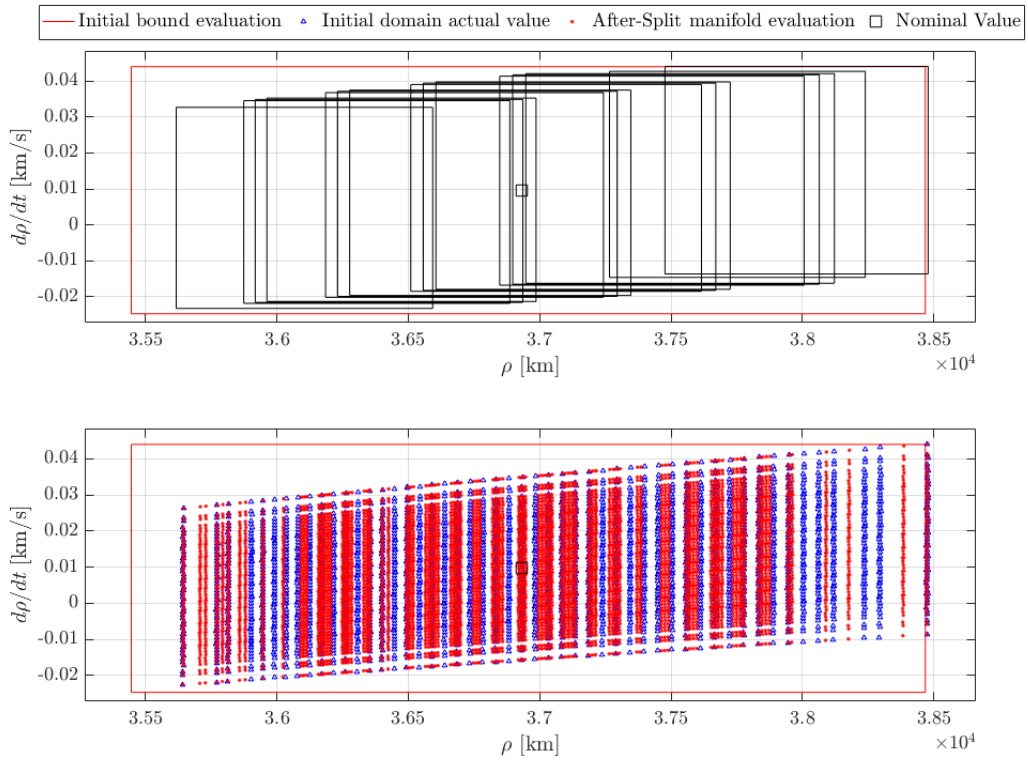


Figure 5.24: AR and ADS results, manifold evaluation vs. actual values. The black box-shape are the bounds of subdomains obtained by ADS are add. Accurate observation.

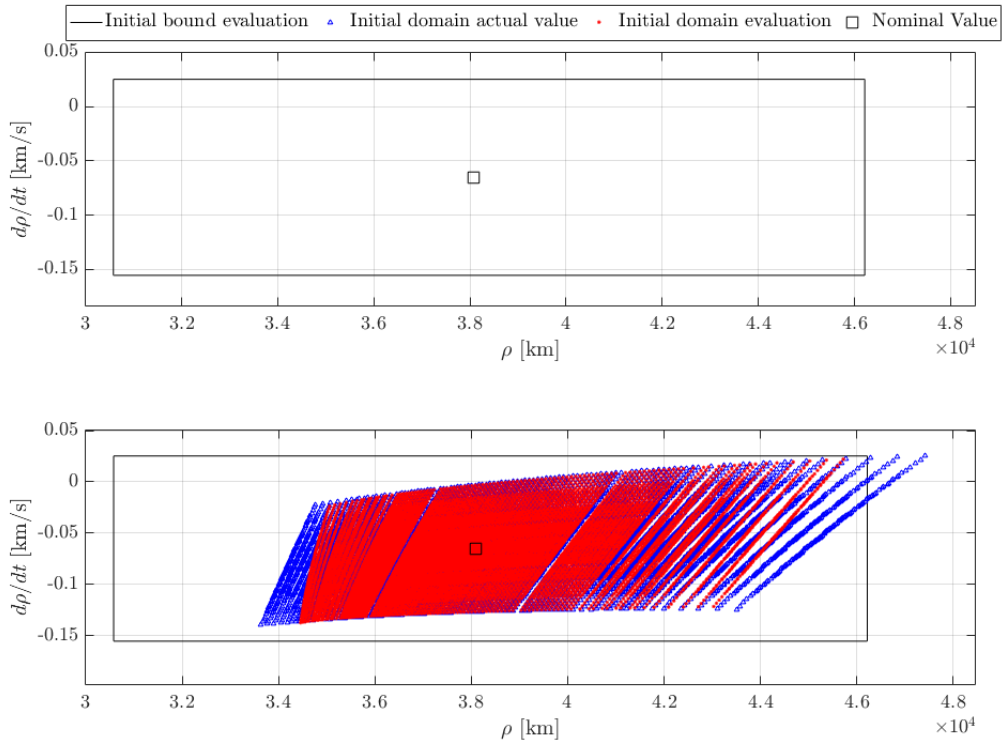


Figure 5.25: AR and ADS results, single domain evaluation vs. actual values. The big black box-shape is the AR, the black square is the mean value of the AR, the red dots are the single domain evaluations and the blue triangles are the actual object state.

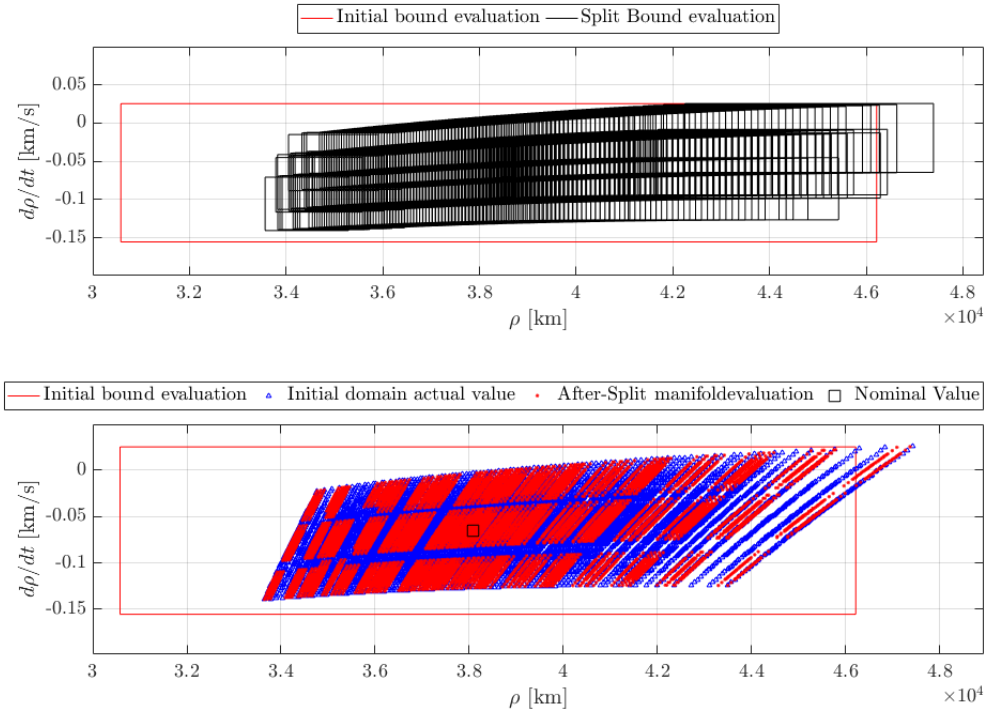


Figure 5.26: AR and ADS results, manifold evaluation vs. actual values. The black box-shape are the bounds of subdomains obtained by ADS are add. Inaccurate observation.

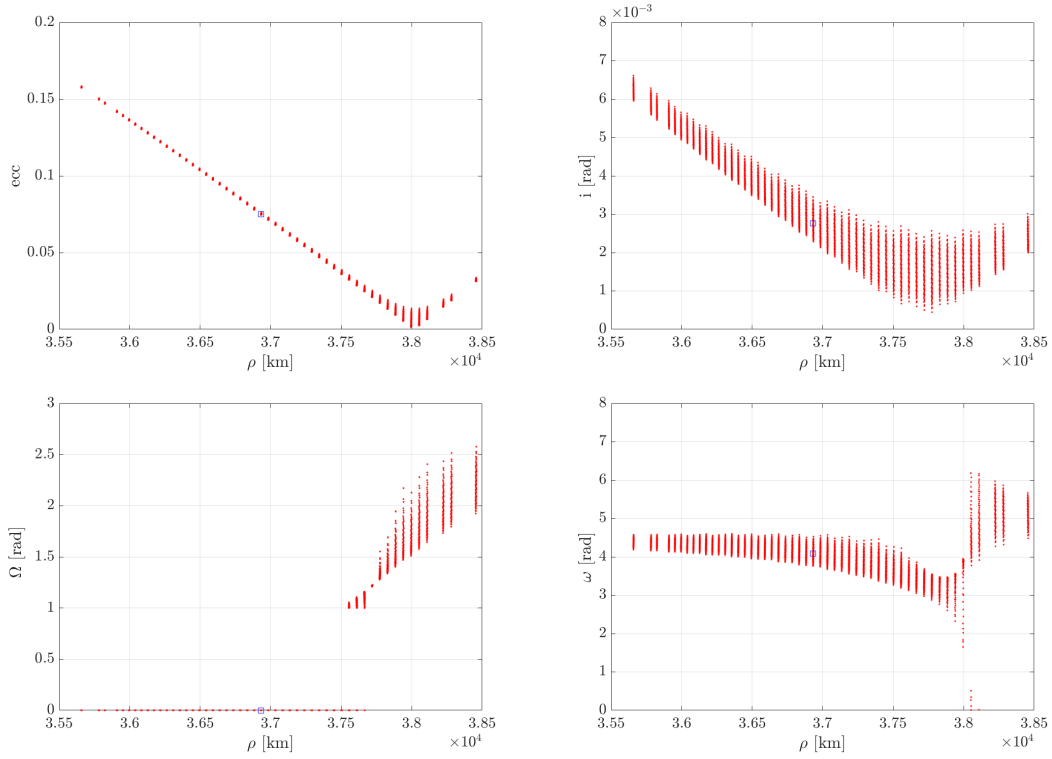


Figure 5.27: Orbital element of the corresponding actual object state of Figure 5.25. Accurate observation.

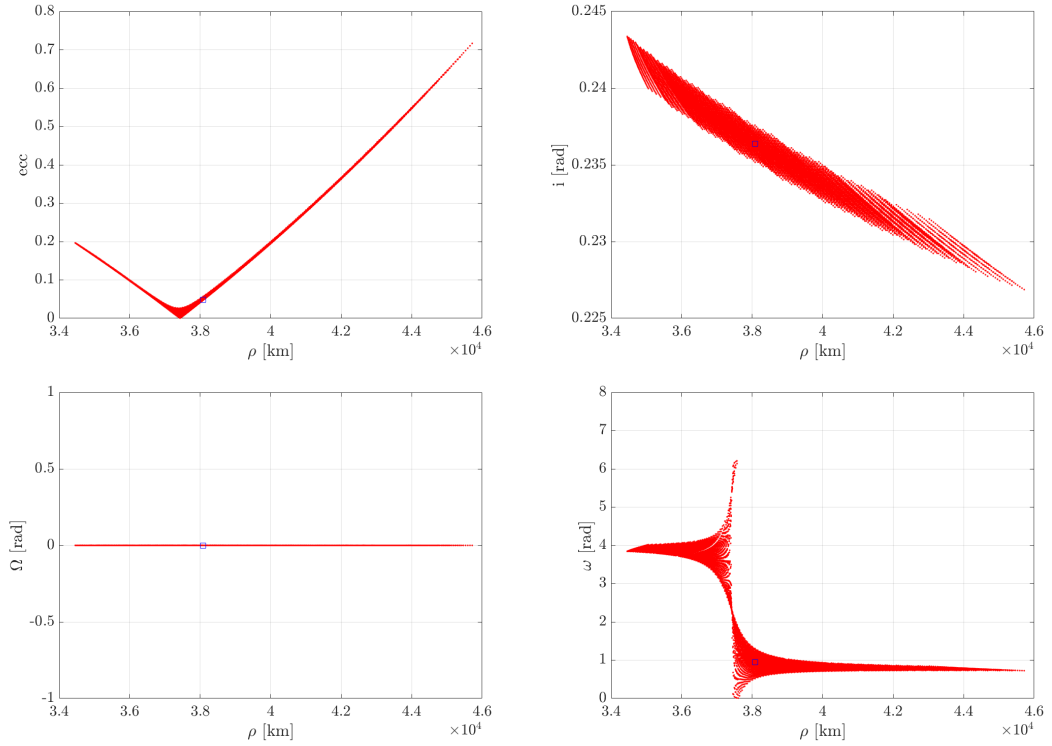


Figure 5.28: Orbital element of the corresponding actual object state of Figure 5.23. Inaccurate observation

Chapter 6

Conclusion

This report has detailed the work of the team in the period June, 16 2016 - January, 16 2017. The Universidad de la Rioja (UR) team has completed the development of the IOD algorithm and it was implemented for the optical observation. This algorithm was developed by exploiting the well-known Gauss's and Lambert's problem and translating them in theDA environment. To validate the IOD algorithm the VO was used. The VO gets the TLEs of real known objects and generates the Observation Set for a given Δt as detailed in Section 2.2. The results obtained by the IOD were analyzed in terms of the AR, already known in literature. The model was compared with the previous approaches as detailed in Section 5.1, and it showed a significant improvement in the description of the uncertainty area associated with the IOD solution.
The PoliMi team focused on the DA representation of the domain, the TPS evaluation and the domain propagation in the DA environment. In particular the DA representation of a mathematical manifold has been implemented, with the creation of the DAmanifold as highlighted in Section 4.1 and 4.2. The DA manifold is useful to represent a domain and its map. After defining the DA Manifold, it was possible to implement the ADS tool with the new DA structure, thus obtaining a better description of the linkage between the domains and their maps during the sampling. The new structure was implemented for the single and multi-variable scalar function and for the multi-variable vector function. Thanks to the improvements introduced, a single routine is able to manage these three different cases. Furthermore, a set of functions to transform between different orbital state representation was implemented in the DA framework.
The last part of the activity focused on merging the IOD and the ADS routines. The objective was to obtain an algorithm that allows for suitably controlling the error in the IOD thanks to the ADS tool. The results with the new merging model were compared with the AR from the literature and with the AR on a single domain. As highlighted in Section 5.2.3.2, the IOD performed with the ADS routine is able to obtained a reduction of the AR, thus a more accurate OS, both for accurate and inaccurate observations.

Bibliography

R. Armellin, P. Di Lizia, F. B. Zazzera, and M. Berz. Asteroid close encounter characterization using differential algebra: the case of aphophis. *Celestial Mechanics and Dynamical Astronomy*, 107(4), 2010.

Roberto Armellin, Pierluigi Di Lizia, and Michele Lavagna. High-order expansion of the solution of preliminary orbit determination problem. *Celestial Mechanics and Dynamical Astronomy*, 112 (3):331–352, 2012. ISSN 09232958. doi: 10.1007/s10569-012-9400-8.

Roberto Armellin, Pierluigi Di Lizia, and Renato Zanetti. Dealing With Uncertainties in Initial Orbit Determination. *Aas 15-734*, pages 1–19, 2015.

Roger R. Bate, Donald D. Mueller, and Jerry E. White. *Fundamentals of Astrodynamics*. Dover Publications, INC., 1971. ISBN 0-486-600061-0.

R.H. Battin. *An Introduction to the Mathematics and Methods of Astrodynamics*. AIAA Education Series, Reston, VA, 1999.

M. Berz. *The new method of TPSA algebra for the description of beam dynamics to high orders*. Los Alamos National Laboratory, 1986. Technical Report AT-6:ATN-86-16.

M. Berz. The method of power series tracking for the mathematical description of beam dynamics. *Nuclear Instruments and Methods A258*, 1987.

H.D. Curtis. *Orbital Mechanics: For Engineering Students*. Aerospace Engineering. Elsevier Science, 2015. ISBN 9780080470542. URL `https://books.google.es/books?id=6aO9aGNBAgIC`.

Kohei Fujimoto and Daniel J. Scheeres. Correlation of Optical Observations of Earth-Orbiting Objects and Initial Orbit Determination. *Journal of Guidance, Control, and Dynamics*, 35(1): 208–221, 2012. URL `http://arc.aiaa.org/doi/abs/10.2514/1.53126`.

L. Isserlis. On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika*, 12(1 and 2), 1918.

Dario Izzo. Revisiting Lambert's Problem. 2014. doi: 10.1007/s10569-014-9587-y. URL `http://arxiv.org/abs/1403.2705{%}0Ahttp://dx.doi.org/10.1007/s10569-014-9587-y`.

Jeffrey M.Lee. *Manifold and Differential Geometry*, volume I. American Mathematical Society, Providence, Rhode Island, 2009.

Andrea Milani and Giovanni Gronchi. *The theory of orbit determination*. 2009.

Andrea Milani, Giovanni Gronchi, Mattia de' Michieli Vitturi, and Zoran Knezevic. Orbit determination with very short arcs. I admissible regions. *Celestial Mechanics and Dynamical Astronomy*, 90(1-2):59–87, 2004. ISSN 09232958. doi: 10.1007/s10569-004-6593-5.

NAIF. Spice: An observation geometry system for planetary science missions, 2016. URL http://naif.jpl.nasa.gov/naif/index.html.

R.S. Park and D.J. Scheeres. Nonlinear mapping of gaussian statistics: theory and applications to spacecraft trajectory design. *Journal of Guidance, Control and Dynamics*, 29(6), 2006.

David A Vallado and Wayne D McClain. *Fundamentals of astrodynamics and applications*, volume 12. Springer Science & Business Media, 2001.

M Valli, R Armellin, P Di Lizia, and MR Lavagna. Nonlinear mapping of uncertainties in celestial mechanics. *Journal of Guidance, Control, and Dynamics*, 36(1):48–63, 2012.

Alexander Wittig. *Astrodynamics Network AstroNet-II*, volume 44 of *Astrophysics and Space Science Proceedings*, chapter An Introduction to Differential Algebra and the Differential Algebra Manifold Representation, pages 293–309. Springer International Publishing, 2016.

Alexander Wittig, Pierluigi Di Lizia, Roberto Armellin, Kyoko Makino, Franco Bernelli-Zazzera, and Martin Berz. Propagation of large uncertainty sets in orbital dynamics by automatic domain splitting. *Celestial Mechanics and Dynamical Astronomy*, 122(3):239–261, 2015. ISSN 1572-9478. doi: 10.1007/s10569-015-9618-3. URL http://dx.doi.org/10.1007/s10569-015-9618-3.

Johnny L. Worthy III and Marcus J. Holzinger. Probability Density Transformations on Admissible Regions for Dynamical Systems. pages 1–20.

List of Acronyms

ADS	automatic domain splitting
AIDA	Accurate Integrator for Debris Analysis
AOD	accurate orbit determination
AR	admissible region
CAR	constrained admissible region
DA	differential algebra
DAIOD	Differential Algebraic Initial Orbit Determination
DAvector	differential algebraic vector
ECI	Earth-centered inertial
FOV	field of view
GEO	geostationary Earth orbit
GTO	geostationary transfer orbit
HEO	high-Earth orbit
IOD	initial orbit determination
KEP	Keplerian classical Element
LEO	low-Earth orbit
MEE	Modified Equinoctial Element
MEO	medium-Earth orbit
NAIF	Navigation and Ancillary Information Facility
NASA	National Aeronautics and Space Administration
OD	orbit determination
ODE	ordinary differential equation
OS	orbit set
PDF	probability density function
SA	short arc
SPICE	Spacecraft Planet Instrument C-matrix Events
TLE	two-line elements
TPS	truncated power series

TSA	too short arc
UR	Universidad de la Rioja
VO	virtual observatory
VSA	very short arc