uc3m | Universidad Carlos III de Madrid

University Degree in Aerospace Engineering
Academic year 2020-2021

*Bachelor Thesis*

# Lambert's problem algorithms: a critical review

Jorge Martínez Garrido

Supervised by:
Manuel Sanjurjo Rivo

September, 2021
Leganés (Madrid)

# Abstract

This work presents a deep performance comparison from both the analytical and computational point of view of a variety of modern Lambert's problem solvers. The analyzed solvers are Gauss 1809, Battin and Vaughan 1984, Gooding 1990, Avanzini 2008, Arora and Russell 2013, Vallado 2013 and Izzo 2015.

A quick review on the problem's timeline, geometry, possible solutions and singularities are presented at first. This is followed by an explanation on the different elements required by any solver together with a proposed way on how to classify these algorithms.

Each selected solver is analyzed considering the free-parameter employed, the initial guess procedure, the numerical root solver and the way velocity vectors are constructed. Figures for the time of flight as function of the independent variable are provided.

The performance comparison is carried out for all possible combinations of the transfer angle and non-dimensional time considering the total number of iterations, the time per iteration and the total computation time. The results are provided in the form of contour plots. A resume table with a conclusion on the obtained results is included too.

Finally, some ideas about how the performance comparison could be extended and enhanced are included in the last chapter.

**Keywords:** Lambert's problem, two-body problem, orbital boundary value problem, algorithms.

# Contents

# List of Figures

# List of Tables

# 1 Introduction to this work

This first chapter is devoted to some introductory concepts about the problem being addressed, the different procedures used and the goals achieved. All these items will help the reader to have a better understanding of the project's structure. In addition, a collection of real-world applications together with a brief socioeconomic analysis are presented in the last sections in order to justify the importance of the problem in today's world.

## 1.1 Problem description and motivation

The two-body boundary value problem in the framework of orbital mechanics and astrodynamics is known as the Lambert's problem. Solutions to this problem find the orbit which connects two known position vectors over a finite time of flight.

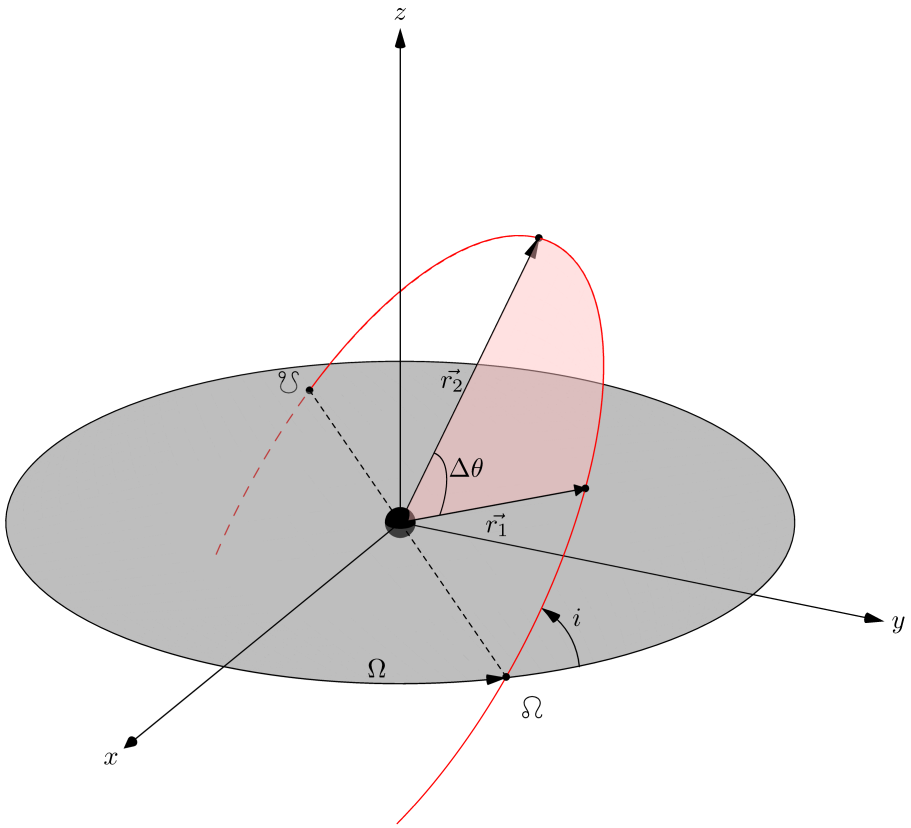


**Fig. 1.1:** The geometry of Lambert's problem as seen from an inertial frame centered in the orbit's attractor. Observation vectors are labeled as $\vec{r_1}$ and $\vec{r_2}$, being $\Delta\theta$ the angle between them and $\Delta t$ the epoch difference.

Since its formulation almost 300 years ago, a plethora of solutions have been proposed. Nevertheless, the problem is still of interest due to its applications (see section 1.3) and many modern authors have addressed it by developing new numerical techniques. Therefore, when it comes to solving for Lambert's problem, a big set of algorithms is available and the following question arises: which one of those performs the best under particular given conditions. The search for a solution to the previous question is the motivation behind this work.

## 1.2 Objectives and goals achieved

The main objective of this project is a critical review and comparison on the most popular solvers for Lambert's problem. To achieve this purpose, the whole process was divided into the following key objectives:

- **Review of Lambert's problem timeline.** The historical background showed which authors have proposed solutions to the problem and when they did so. Important scientific figures arose when carrying out this part of the research.

- **Analysis of problem's geometry, solutions and singularities.** A deep review on the problem was performed by listing all the different solutions their nature. Multi-revolution transfers were also considered together with limiting cases.

- **Collection of modern algorithms.** A mathematical comparison between all of them was carried out taking into account four factors: the selected iteration variable, the numerical method used, proposed initial guess and the way of computing the velocity vectors.

- **Performance framework and comparison.** All computer algorithms were coded under the same programming language, so no undesired additional performance was introduced. Contour maps relating the transfer angle, time of flight and number of iterations to achieve desired accuracy were computed to illustrate the robustness of each routine.

- **Discussion on the obtained results.** Strengths and weaknesses of each algorithm were exposed with previous analysis. All the results were collected and studied to raise new ideas regarding solutions performance of future solvers.

The implementation of all collected algorithms lead to the creation of a Lambert's problem solvers hub in the form of a software library[1], which will be useful for other authors when testing new numerical procedures.

[1] Available under public license at: https://github.com/jorgepiloto/lamberthub

## 1.3 Real-world applications

Lambert's problem solvers have lots of applications because they retrieve the solution to the *boundary value problem* (BVP) of the restricted two-body dynamics problem. This solution is nothing but the conic trajectory which the body will follow as time evolves. The previous fact makes Lambert's problem to be usually included within the initial *orbit determination topic* (IOD).

However, the most common applications for the Lambert's problem are targeting, maneuvering and rendezvous ones. For example, if it is desired to navigate from an initial position to a final one over a finite amount of time then, the orbit which must be followed is the one obtained by solving the Lambert's problem.

In fact, Lambert's solvers routines were implemented in the *Apollo Guidance Computer* (AGC[2]) for the calculation of transfer trajectories and reentry ones. In the *Apollo Experience Report: Onboard Navigational and Alignment Software*, it is claimed that the algorithms (AGC programs P-31 to P-37) took between 15 to 30 minutes to compute a solution, which showed the necessity of having better performance solvers for future missions.

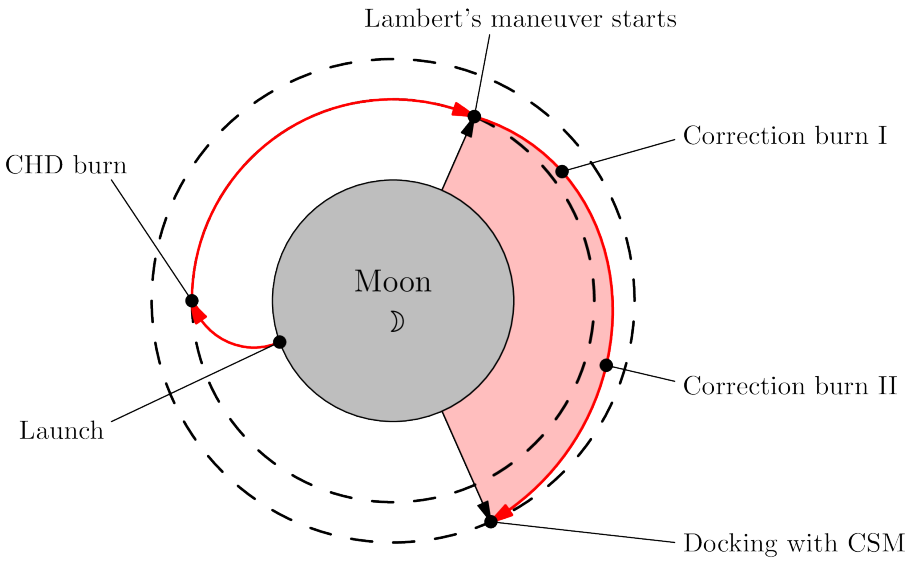


**Fig. 1.2:** After the *Lunar Module* (LM) was launched into a *Constant Delta Height* (CDH) orbit, a set of targetting maneuvers were required to perform a rendezvous with the *Command and Service Module* (CSM). The AGC made use of its own Lambert's solvers routines to calculate the required impulses in order to correct the course of the LM spacecraft.

Course corrections, as shown in figure 1.2, might be required during the transfer, forcing to compute a new solutions by calling the different transfer routines several times in order to reach final aiming location.

[2]The original source code for the AGC has been published under a Public Domain license in https://github.com/chrislgarry/Apollo-11

Last application is closely related with the mission analysis field. Solving Lambert's problem for a given set of launching and arrival dates leads to a collection of particular orbits, each one corresponding to a transfer arc. Those conic trajectories show a different characteristic velocity, directly related with the amount of required energy. Contour maps representing this set of solutions are known as porkchop plots and it is possible to observe that minimum transfer points exist within these figures. The locations in the contour figure are usually selected for space missions launch and arrival dates.

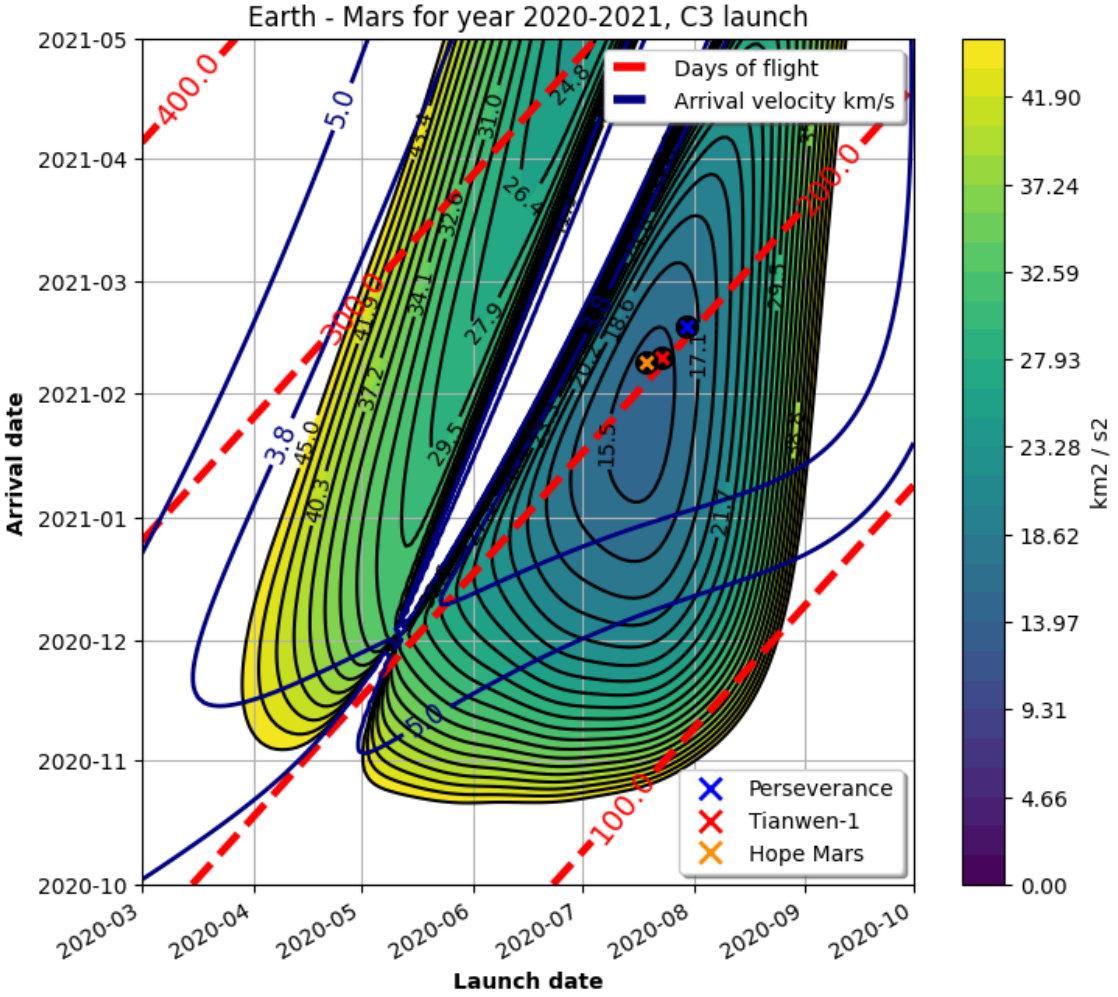


**Fig. 1.3:** Porkchop plot for Earth-Mars transfer arrival energy for years 2020-2021. Each point in the figure is obtained by solving the Lambert's problem in order to compute departure $C_3$ (characteristic energy). Notice the optimal transfer dates are being used by latest missions about this planet: *Perseverance* 2020, *Tianwen-1* 2021 and *Hope* 2021.

With all previous examples on mind, it is clear that Lambert's applications require from high performance solving routines to compute in a fast and reliable way final desired orbit trajectories.

## 1.4 Social and economical context

During the last decades, the amount of unmanned spacecrafts orbiting our planet has dramatically increased. The main reason behind this situation is the rise of private launchers and manufacturers, who have lowered the costs, enabling others to get access to space. This lead to a new aerospace race driven by cheap technologies, usually referred to as *New Space*, which provide agile, responsive and simpler solutions to common aerospace tasks.

In addition, a huge demand of communication services is already being experience by the aerospace market. Nowadays, this sector generates benefits over 100 billion euros due to new improvements in satellites' software and hardware, see *Airbus - New Space* 2021. Communication services have a variety of purposes, from in-flight connectivity to network-centric warfare, which are fulfilled by making use of satellite constellations in order to guarantee a permanent coverage over particular areas of the globe.

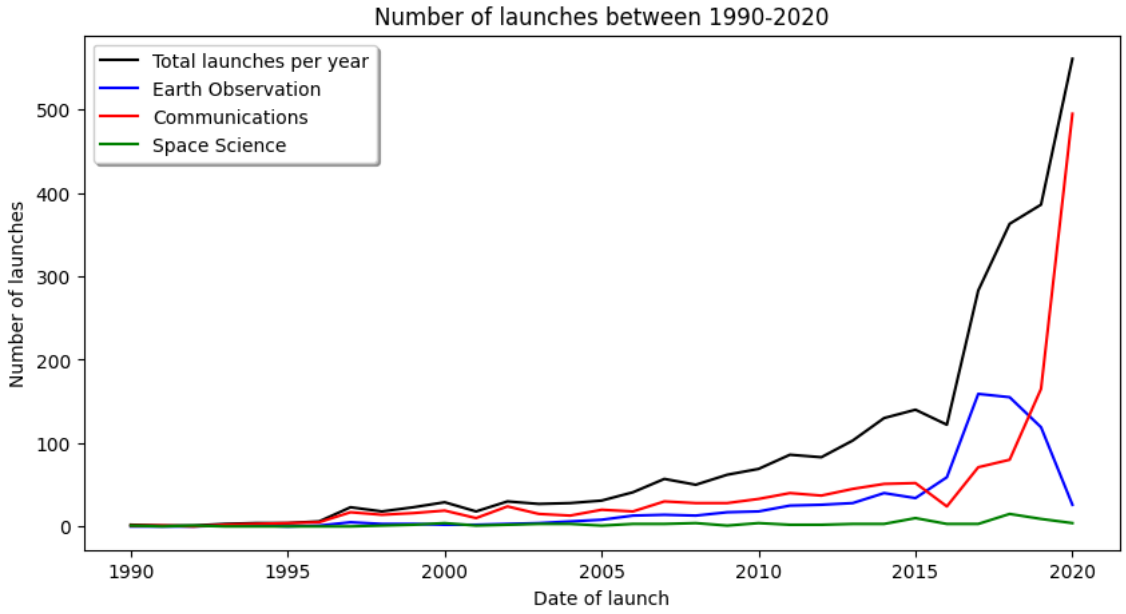


**Fig. 1.4:** Amount of satellites launched during last decades and filtered by their mission purpose: space science, Earth observation and communication. This last one, shows a big growth rate specially during the last five years. Data was obtained and processed from the UCS Satellite Database.

All previous satellites' missions, shown in figure 1.4, are designed to take place at particular locations of space, as some of those orbits require specific altitudes and inclinations: geostationary, sun-synchronous... However, space is a very hostile environment and orbit perturbations exist due to air drag, Earth's oblateness, third body presence or solar radiation pressure among many others. These perturbations might originate a shift in the initial orbit, making it necessary to maneuver the satellite to original desired position. Orbit corrections require from targeting algorithms which, in the end, are Lambert's problem solvers.

Within the frame of aerospace sector, a new professional profile is being demanded

by companies: the computer scientist. These people are usually engineers who have a deep knowledge in both high-performance computing and mathemtatics modelling the problem. This is one of the reasons behind why many modern papers, which devise a new approach for the Lambert's problem, include algorithms in the form of code flowcharts, pseudo-code or links to repositories hosting original source computer routines.

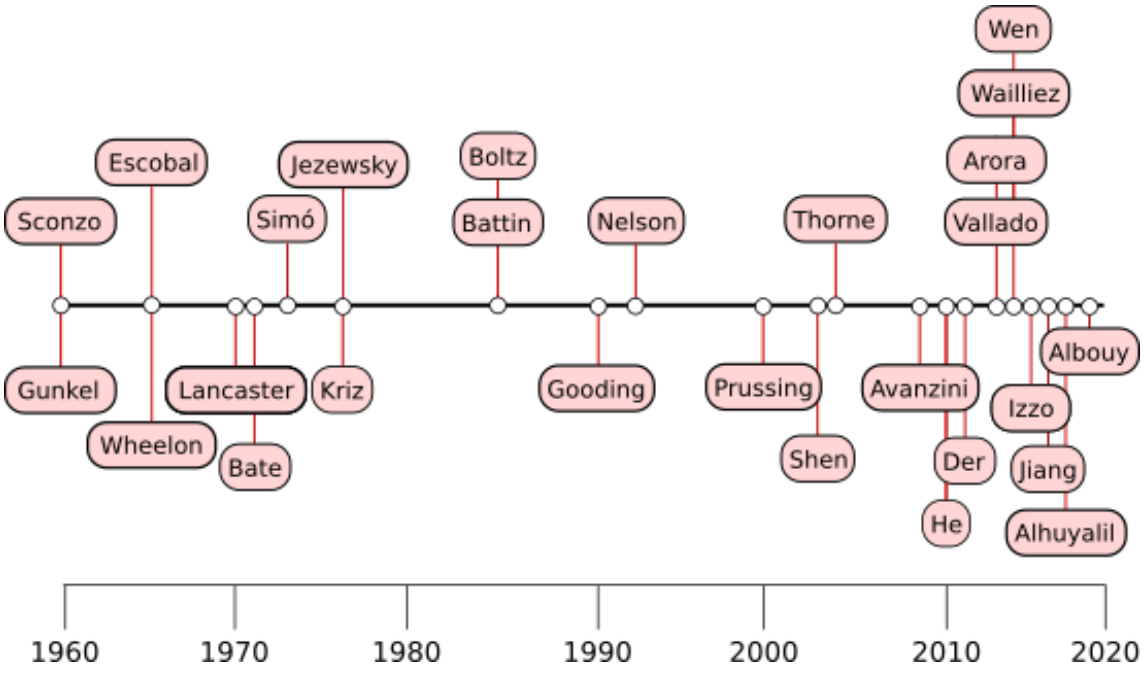


**Fig. 1.5:** More than a dozen of papers have been published over the last twenty years. The nature of the proof solutions has evolved also during time, going from pure geometric demonstrations to graphic processing units based or even machine learning ones.

# 2 Review of Lambert's problem

In this chapter, the problem statement, geometry, solutions and singularities are revisited. The analysis presented here is strongly influenced by Escobal 1965, Hilton 1968 and Battin 1999 works. All scenarios, including multiple revolutions and degenerate ones are considered.

## 2.1 The statement of the problem

Lambert's problem states to find for the Keplerian orbit which connects two known position vectors, $\vec{r_1}$ and $\vec{r_2}$, over a finite amount of time, $\Delta t$, under a gravity field of gravitational parameter $\mu$. The three-dimensional representation of this problem was initially introduced by figure 1.1. This problem is usually named *the direct-arc transfer problem* and it was posed by Johann Heinrich Lambert in a letter to Leonhard Euler as properly pointed by Albouy 2019. Albouy made an outstanding work by reviewing the whole problem's timeline from the very beginning up to the modern days.

The direct transfer problem is the most common one when addressing the Lambert's problem. However, it is also possible to specify the number of revolutions of the orbiting body before reaching the final position vector. This type of problem is known as *the multi-revolution problem.*

It is important to say that perturbations will not be considered during this whole work: only the Keplerian Lambert's problem is studied.

Because the answer to either the direct-arc transfer of the multi-revolution one is an orbit, we then need a total of six parameters (3 translations + 3 rotations) to fully describe its shape, location and orientation. These set of parameters is usually referred to as *state vector*. The state vector can be expressed in different ways:

- RV: composed by the three components of the initial position vector $\vec{r_1}$ and the other three components of the initial velocity vector $\vec{v_1}$.
- COE: known as the *classic orbit elements*. The elements which form this set are the semi-major axis $a$, the orbit's eccentricity $e$, its inclination $i$, right ascension of the ascending node $\Omega$, argument of the perigee $w$ and finally the true anomaly denoted by $\nu$.

Therefore, the Lambert's problem is solved once a full set of orbital elements is retrieved, as the orbit between the two known position vectors is finally known.

## 2.2 The orbital plane

Assuming that the time of flight is not null, and that the position vectors are not coincident neither forming an angle of 180 degrees between them, the whole orbit movement will be contained within a plane. This plane is usually named the orbital plane. The red colored plane in figure 1.1 represents it.

For the case of the Lambert's problem, the orbital plane can be computed from the initial and final position vectors. These two vectors form a basis from which any other vector contained in the plane can be obtained from the relation shown in equation 2.1:

$$\vec{r} = a\vec{r_1} + b\vec{r_2} \tag{2.1}$$

where $a$ and $b$ are two known real values.

As long as both vectors are linearly independent, the orbital plane is defined. This implies that the cross product $\vec{r_1} \times \vec{r_2}$ must be not null, otherwise singularities arise. This justifies previous assumption about not having two position vectors which might be parallel (i.e. forming 0 or 180 degrees).

The orientation of the orbital plane in space with respect to (w.r.t.) is given by the right ascension of the ascending node (RAAN) $\Omega$ and the inclination $i$. Therefore, it is interesting to analyze the Lambert's problem by making use of the COE set instead of the RV one. However, once the COE elements are known, a conversion to the RV ones is required, as the initial elements given in the statement of the problem are the position ones.

Because both the RAAN and the inclination are computed from a vector in the direction and sense of the specific angular momentum of the orbit, $\vec{h}$, the following subsection presents a simple procedure on how to solve for this particular vector.

### 2.2.1 Solving for the unitary specific angular momentum vector

It is not possible to compute the specific angular momentum of the orbit because a velocity at a particular point would be required. This condition would satisfy the RV set, meaning that the orbit is fully determined. However, a unitary vector in the direction and sense of the angular momentum can be computed making use of the two knwon position vectors and the sense of motion of the observed body.

The sense of motion (prograde or retrograde)[3] is an important parameter, since it fixes the sense of the angular momentum of the orbit. The reason is that, when computing a normal vector to the normal plane via equation 2.2

[3]Prograde and retrograde orbits are defined according if their inclinations are lower or greater than 90 degrees respectively.

$$\vec{h_0} = \frac{\vec{r_1} \times \vec{r_2}}{\|\vec{r_1}\|\|\vec{r_2}\|} \tag{2.2}$$

the resultant normal vector has always the sense imposed by the so-called *right hand rule* according to the shortest angle between $\vec{r_1}$ and $\vec{r_2}$. In addition, notice that vector $\vec{h_0}$ goes in the direction of the $\vec{h}$ vector, but does not have neither its sense or magnitude.

Therefore, the value of $\vec{h_0}$ needs to be corrected according if it does not agree with a prograde/retrograde type of motion observed. This implies checking for the sign of the third component of via $h_{0_Z} = \vec{k} \cdot \vec{h_0}$ and apply:

| **Sign of the vertical component** | **Prograde** | **Retrograde** |
|---|---|---|
| $h_{0_Z} > 0$ | $\vec{h_0}$ | $-\vec{h_0}$ |
| $h_{0_Z} < 0$ | $-\vec{h_0}$ | $\vec{h_0}$ |

Table 2.1: Sign correction for the normal specific angular momentum vector.

where vector $\vec{k}$ in the dot product operation is a unitary one in the direction and sense of the positive $Z$ axis of the inertial reference frame selected to study the problem.

With previous correction, the resultant vector $\vec{h_0}$ goes in the direction and sense of the specific angular momentum one but has unitary modulus.

### 2.2.2 Solving for the inclination of the orbit

Once $\vec{h_0}$ has been computed and its sign properly fixed, the inclination of the orbit can be computed by making use of expression 2.3:

$$i = \arccos\left(\frac{h_{0_Z}}{\|\vec{h_0}\|}\right) \tag{2.3}$$

Because the sign correction was previously applied, the value of the inclination must agree with the prograde/retrograde type of motion. Furthermore, the output of the arccos function is bounded to the inclination domain (i.e. 0 and $\pi$ radians).

### 2.2.3 Solving for the RAAN of the orbit

Finally, the right ascension of the ascending node is computed as usual. First, a vector in the direction of the line of nodes is computed taking advantage of the expression 2.4:

$$\vec{n} = \vec{k} \times \vec{h_0} \tag{2.4}$$

With previous vector, it is possible now to compute the RAAN by applying equation 2.5

$$\Omega = \arccos\left(\frac{\vec{i} \cdot \vec{n}}{\|\vec{i}\|\|\vec{n}\|}\right) \tag{2.5}$$

where vector $\vec{i}$ in the dot product operation is a unitary one in the direction and sense of the positive $X$ axis of the inertial reference frame selected to study the problem.

Because the RAAN is bounded between $[0, 2\pi)$, a correction might be required, as the arccos function only returns values between $[0, \pi)$. This correction is triggered if the lateral component of the line of nodes, $\vec{n_j} < 0$, so the final value of $\Omega$ becomes:

$$\Omega = 2\pi - \Omega \Rightarrow \vec{n_j} < 0 \tag{2.6}$$

From the point of view of a computer routine, it is better to make use of the arctan2[4], which returns the proper quadrant between the input coordinates, so the RAAN can be easily computed following relation 2.7:

$$\Omega = \arctan2(n_y, n_x) \tag{2.7}$$

where $n_y$ and $n_x$ are the lateral and longitudinal coordinates of the line of nodes vector.

Once $\Omega$ has been solved together with $i$, the complete orientation in space of the orbital plane is known.

### 2.2.4 Solving for the transfer angle

The computation of the transfer angle requires the knowledge of the sense of motion (prograde or retrograde) of the orbiting body. By taking advantage of the definition of the dot product, it is possible to solve for the angle between the initial and final position vector such us:

$$\Delta\theta_0 = \arccos\left(\frac{\vec{r_1} \cdot \vec{r_2}}{\|\vec{r_1}\|\|\vec{r_2}\|}\right) \tag{2.8}$$

[4]This routine was originally introduced by the FORTRAN language under the name of atan2.

However, notice that equation 2.8 will always output an in the range of $[0, \pi]$ radians. A correction, similar to the one applied to the normal vector needs to be performed according to the sense of motion:

$$\Delta\theta = \begin{cases} \Delta\theta_0 & \text{if prograde and } h_{0_z} > 0 \\ 2\pi - \Delta\theta_0 & \text{if prograde and } h_{0_z} < 0 \\ 2\pi - \Delta\theta_0 & \text{if retrograde and } h_{0_z} > 0 \\ \Delta\theta_0 & \text{if retrograde and } h_{0_z} < 0 \end{cases} \tag{2.9}$$

A graphical representation of the geometry of the transfer angle is discussed by Der 2011, who devotes a whole section for this important but poorly covered topic in Lambert's literature.

### 2.2.5 The problem as seen from the orbital plane

Because the orbital plane is completely known, the following parameters are still required to solve for the orbit: $a$, $e$, $w$ and $\nu$. Notice that the true anomaly, that is $\nu$, only gives information about the current location of the body along its orbit but not about the shape of the orbit itself. Therefore, the problem is just devoted to find the values of $a$, $e$ and $w$.

Since all the motion of the problem is contained in the orbital plane, it is feasible to study the Lambert's problem a reference who's $z = 0$ plane is coincident with previous plane. Reader might have thought about using the *perifocal coordinate system* (PQW) but the argument of perigee is still an unknown of the problem, so it is not possible to use the PQW frame. With previous assumption, we can simplify the geometry given by figure 1.1 into the one shown by figure 2.1.

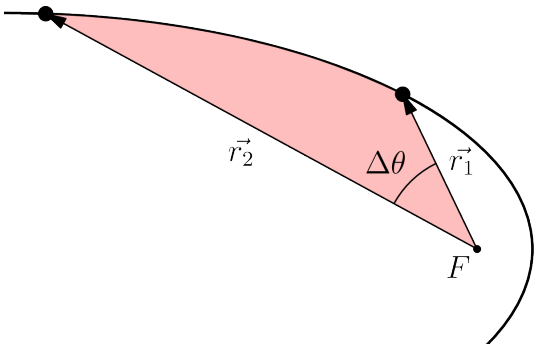


**Fig. 2.1:** The geometry of the Lambert's problem as seen from an inertial frame who's $XY$ plane coincides with the orbital one.

Notice that the attracting body in figure 2.1 is located in the known focus $F$, following the two-body dynamics assumption. The other focus $F'$ is still unknown and needs to be solved to retrieve the full geometry of the orbit. This also introduces the unknown of the $w$, as the line where for the semi-major needs to be guessed.

## 2.3 Lambert's theorem and implicit solution

The Lambert's problem receives its name after Johann Heinrich Lambert formulated it in a letter to Leonhard Euler, see Albouy 2019 for more information about Lambert's problem origin and timeline.

Lambert proof in his *Insigniores orbitae cometarum proprietates* that the time of flight between two position vectors only depends on the sum of these radial distances, the linear distance between them and the semi-major axis of the orbit. This is known as *the Lambert's theorem.*

### 2.3.1 The vis-viva equation in its differential form

To demonstrate Lambert's theorem, the original procedure devised by him is presented in the following lines. Starting from the vis-viva equation in 2.10:

$$v = \frac{dr}{dt} = \sqrt{\frac{2\mu}{r} - \frac{\mu}{a}} \tag{2.10}$$

it is possible to integrate to find that:

$$\Delta t = \frac{1}{\sqrt{\mu}} \int_{s-c}^{s} \frac{r dr}{\sqrt{2r - r^2/a}} \tag{2.11}$$

where in previous equation $s$ is the semi-perimeter and $c$ the chord, being given by:

$$s = \frac{\|\vec{r_1}\| + \|\vec{r_2}\| + \|\vec{c}\|}{2} \qquad\qquad c = \|\vec{c}\| = \|\vec{r_2} - \vec{r_1}\| \tag{2.12}$$

### 2.3.2 Series solution to the vis-viva differential equation

Lambert provided equation 2.11 in the form of a series[5]: 2.13:

$$\Delta t = \frac{1}{\sqrt{\mu}} \left( \frac{\sqrt{2}}{3} [s^{\frac{3}{2}} \mp (s-c)^{\frac{3}{2}}] + \sum_{n=1}^{\infty} \frac{\sqrt{2}}{2n+3} \frac{(2n-1)!}{2^{3n-1} n! (n-1)!} [s^{n+\frac{3}{2}} \mp (s-c)^{n+\frac{3}{2}}] \frac{1}{a^n} \right) \tag{2.13}$$

[5]This equation has been directly taken from Battin 1999, in particular from problem 7-1 of the book.

However, notice that for solving the value of the semi-major axis a numerical method is required, as equation 2.13 is not in explicit form for $a$. It is interesting to see that the first term of the series (the one not depending on $a$) is the solution found by Euler some years before Lambert published his book. This term corresponds to the solution to the pure parabolic transfer, for which the semi-major axis has a value of $a = \infty$, making the rest of the series terms to become null.

Lambert told Euler that he would devise the explicit form of 2.13 (see Albouy 2019) but Lambert's untimely death prevented this.

## 2.4 Geometry of the solutions

Before introducing or proposing any mathematical method for solving equation 2.13, we will assume that we managed somehow to obtain the right value for the semi-major axis of the orbit.

With this value, it is possible to identify the type of orbit, according to the following table:

| $0 < a < \infty$ | $a = \infty$ | $a < -\infty$ |
|---|---|---|
| Elliptic | Parabolic | Hyperbolic |

Table 2.2: Type of orbit according to semi-major axis value.

Knowing the value of $a$ it is possible to estimate the rest of the orbital elements so the observed orbit is fully determined. However, each type of orbit requires from its own solution procedure when determining the eccentricity, argument of perigee and the current true anomaly.

Therefore, in the following subsections, the possible solutions and procedures for each type of orbit are exposed from a geometrical point of view.

### 2.4.1 The elliptic transfer geometry

An elliptic orbit has a semi-major axis who's value lies between $[0, \infty)$ units of length. By recalling that:

$$\overline{FP} + \overline{F'P} = 2a \tag{2.14}$$

being $\overline{FP}$ and $\overline{F'P}$ the distance to the primary[6] and vacant[7] focus, it is possible to find the location of this last one.

[6]The primary focus is the one in which the attractor is placed.
[7]Named like this because it does not hold the attractor.

In order to achieve previous task, we make use of equation 2.14 and evaluate it for each known position vectors:

$$r_1' = 2a - r_1 \qquad r_2' = 2a - r_2 \tag{2.15}$$

The values $r_1'$ and $r_2'$ indicate the distance at which the vacant focus $F'$ is located. Therefore, by tracing a circumference of radius $R_1 = r_1'$ centered at the tip of vector $\vec{r_1}$ and another with $R_2 = r_2'$ at the tip of vector $\vec{r_2}$ it is possible to find graphically where the vacant focus is located.

### Two vacant focus

When both circumferences cross at two points, meaning there are two possible vacant focus, as shown by figure 2.2. Having two focus means having two elliptic paths and a total of four orbits:

- Red orbit and prograde motion.
- Red orbit and retrograde motion.
- Blue orbit and prograde motion.
- Blue orbit and retrograde motion.

By imposing the sense of motion to be prograde or retrograde, the solution space is reduced to two possible orbits. To distinguish between the two, we introduce a new parameter named *the orbit path*, which enables us to select between one or the other. This parameter was introduced by Sun 1977.

In this scenario, it holds that:

$$a > \frac{\|\vec{r_1}\| + \|\vec{r_2}\| + \|\vec{c}\|}{4} \tag{2.16}$$

Finally, it must be said that these solution only appear in the multi-revolution problem.

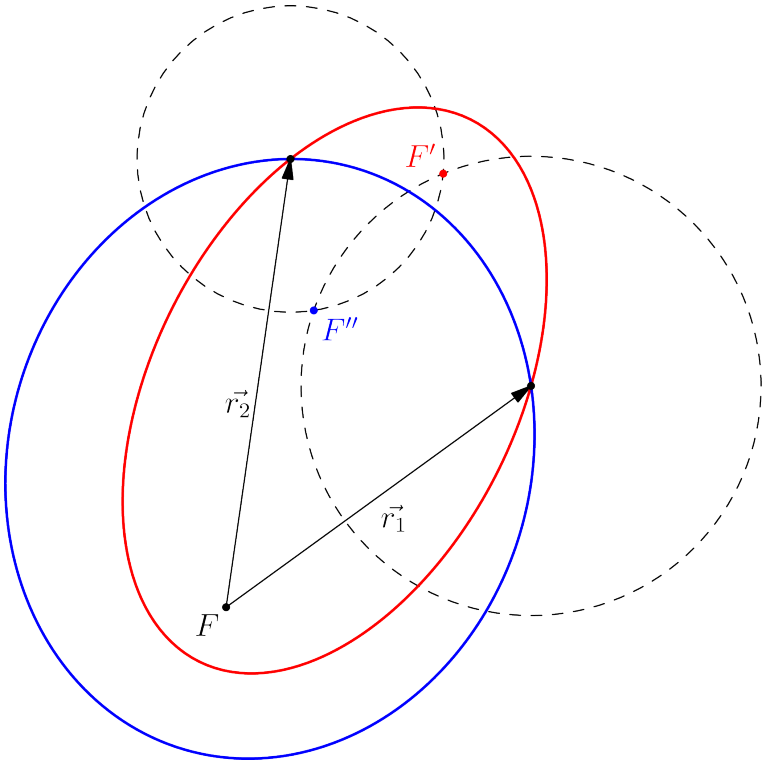


**Fig. 2.2:** In this scenario, there are a total of four solutions considering two possible orbits (blue and red) and the transfer angle (lower or greater than 180 degrees), which is equivalent to prograde or retrograde motion. Fixing the type of motion together with the orbit path parameter will retrieve the current solution to the problem.

### One vacant focus

In this case, both circumferences are tangent to each other, meaning there is only one common point and thus a single vacant focus. See figure 2.3 for a graphical representation.

In this case, only two solutions can be achieved:

- Red orbit and prograde motion.
- Red orbit and retrograde motion.

Similarly to previous case, imposing the sense of motion will retrieve the current solution to the problem. Notice now that the orbit path parameter is no longer required as only one orbit (red one) exists.

For this scenario, the semi-major axis is found to be:

$$a = \frac{\|\vec{r_1}\| + \|\vec{r_2}\| + \|\vec{c}\|}{4} \tag{2.17}$$

A single vacant focus is likely to appear for the direct arc transfer problem.

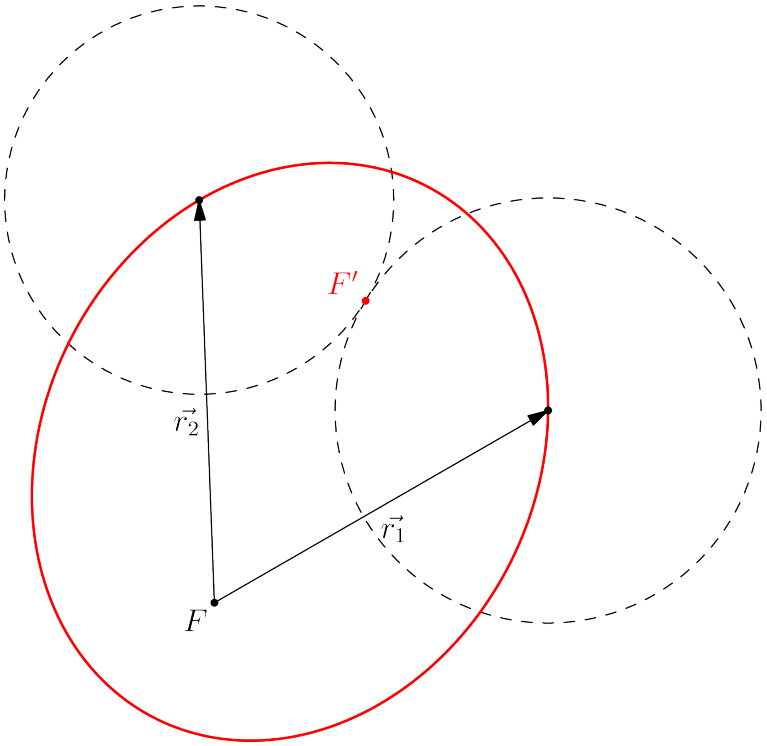


**Fig. 2.3:** In this scenario, there are only two solutions considering a single orbit (red one) and the transfer angle (lower or greater than 180 degrees), equivalent to prograde or retrograde motion. Fixing the type of motion will retrieve the current solution to the problem.

### No vacant focus

In this last situation, there is not a common point between the two circumferences, meaning that the transfer orbit does not exist for that particular geometry.

In fact, the semi-major axis turns out to be:

$$a < \frac{\|\vec{r_1}\| + \|\vec{r_2}\| + \|\vec{c}\|}{4} \tag{2.18}$$

meaning that the problem is unsolvable.

### Computing the rest of the elements in the elliptic case

Once the location of the vacant focus has been identified, it is possible to solve for the focal distance $c$ by applying equation 2.20:

$$c = \frac{\overline{FF'}}{2} \tag{2.19}$$

from which the semi-minor axis $b$ can be obtained:

$$b = \sqrt{a^2 - c^2} \tag{2.20}$$

Since now both axes, the semi-major and semi-minor are known, the eccentricity of the orbit can be computed using 2.28:

$$e = \sqrt{1 - \frac{a}{b}} = \frac{\overline{FF'}}{2a} \tag{2.21}$$

Notice all previous equations can be applied with $\overline{FF''}$ instead of $\overline{FF'}$.

Finally, the last element, that is the argument of perigee, can be computed knowing the angle between the initial position vector and the line of nodes such that:

$$w = \arccos\left(\frac{\vec{n} \cdot \vec{r_1}}{\|\vec{n}\|\|\vec{r_1}\|}\right) + \angle P_1 F F' \pm \pi \tag{2.22}$$

where in equation 2.22, $P_1$ is the point given by the tip of vector $\vec{r_1}$.

### 2.4.2 The parabolic transfer geometry

Any point which belongs to a parabola holds the following relation:

$$\overline{FP} = \overline{lP} \tag{2.23}$$

where in previous equation, $l$ is the *directrix* of the parabola. Equation 2.23 shows that the distance from any point of the parabola is equal to the distance of that point to the directrix.

Therefore, to locate the directrix of the transfer parabola, drawing two circumferences of radius $R_1 = \|\vec{r_1}\|$ and $R_2 = \|\vec{r_2}\|$ centered at the tip of vectors $\vec{r_1}$ and $\vec{r_2}$ respectively. The directrix is given by the tangent lines between these circumferences. The resultant graphic procedure is shown by figure 2.4:

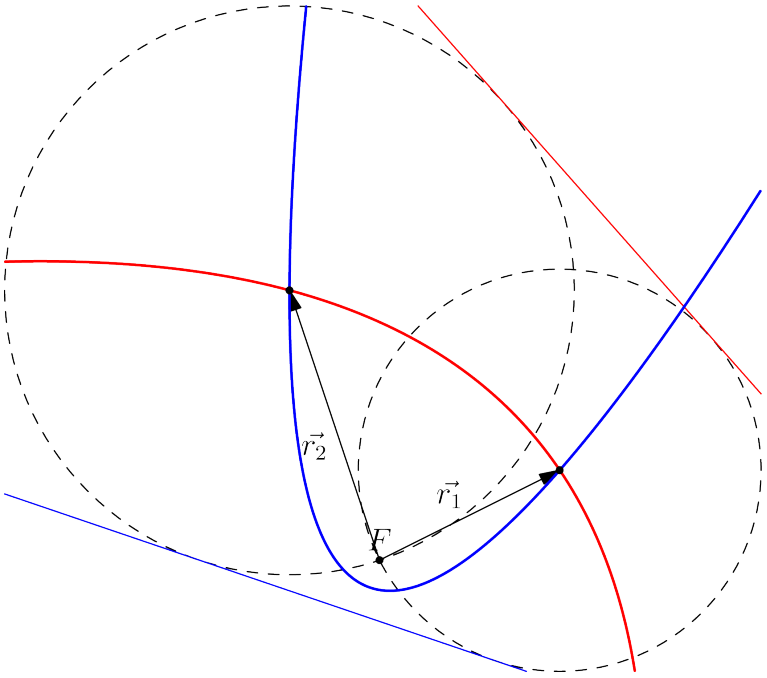


**Fig. 2.4:** For the parabolic scenario two possible solutions are available, even if there exist two orbits. This is because parabolic orbits are not closed and the path can only be followed in one sense. Therefore, fixing retrograde or prograde motion fixes the solution to the red or blue orbits.

In this case, the exact value for the semi-major axis of the parabola is known, being $a = \pm\infty$ together with the eccentricity $e = 1.00$. Therefore, the only left parameter is the argument of perigee, which can be computed

$$w = \arccos\left(\frac{\vec{n}\cdot\vec{r_1}}{\|\vec{n}\|\|\vec{r_1}\|}\right) + \angle P_1 F D' \tag{2.24}$$

where in equation 2.24 the value $D$ refers to the shortest distance between the focus of the parabola $F$ and its directrix.

### 2.4.3 The hyperbolic transfer geometry

Similarly to the elliptic case, for the hyperbola the geometric condition that any point on the curve fulfills is given by equation 2.25:

$$\overline{FP} - \overline{F'P} = 2a \tag{2.25}$$

Therefore, the distance to the vacant focus can be computed as:

$$r_1' = 2a + r_1 \qquad r_2' = 2a + r_2 \tag{2.26}$$

Again, previous values $r'_1$ and $r'_2$ are the radius of the circumferences which must be centered at the tip of vectors $\vec{r_1}$ and $\vec{r_2}$ to properly locate the vacant focus.

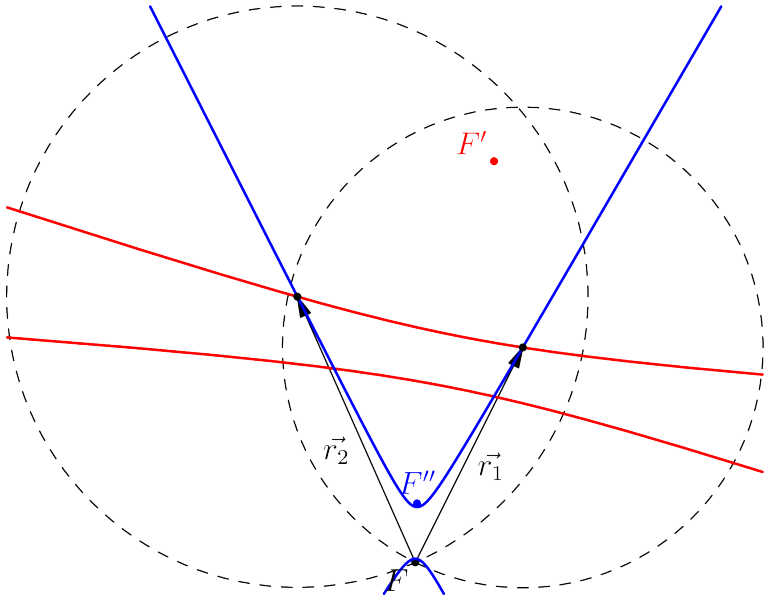


**Fig. 2.5:** For the hyperbolic transfer two solutions are available for a total of two orbits, blue and red ones. This is similar to the parabolic case, because the orbit is not closed. Notice that only the paths of the hyperbolas which pass trough the position vectors are the ones where the orbiting body could be found.

For the hyperbolic orbit, both previous circumferences always meet at two different points, as seen in figure 2.5. This is because:

$$r'_1 + r'_2 = 4a + \|\vec{r_1}\| + \|\vec{r_2}\| > c \tag{2.27}$$

For solving the rest of the elements, and similarly to the elliptic transfer orbit, the following relations can be applied:

$$e = \frac{\overline{FF'}}{2a} \tag{2.28}$$

and:

$$w = \arccos\left(\frac{\vec{n}\cdot\vec{r_1}}{\|\vec{n}\|\|\vec{r_1}\|}\right) + \angle P_1 F F' \tag{2.29}$$

### 2.4.4 Singularities of the problem

The Lambert's problem presents some singularities for particular. That is the case for non-independent position vectors, meaning that the transfer angle between them is either 0 or 180 degrees.

In this case, equation 2.2 becomes null and the direction of the normal vector to orbit plane cannot be properly determined. Not only that, equation 2.3 also suffers from a singularity, as $\|\vec{h_0}\| = 0$ is in the denominator of the expression. Also, the line of nodes and $\Omega$ can not be computed with the available data.

Rectilinear orbits also produce singularities, as they move along a straight path for which all the position vectors are proportional. For these type of orbits, velocity of the orbiting body goes in the direction of the attracting force during the whole motion. From the point of view of real-world cases, this scenario appears when a body is released without initial velocity, so it falls onto its attractor.

All previous singularities are caused by a common reason: a non linearly independent input which makes corner cases to appear. To avoid those

- For two equal vectors and $\Delta t \neq 0$, the point is not moving at all considering these vectors are expressed with respect to an inertial frame[8] centered in the attractor. This is not a valid type of motion for the two-body dynamics.
- For a 180 degrees transfer angle, an assumption needs to be done about the fundamental orbit of the plane, since $i$ nor $\Omega$ are defined. Some authors impose the $XY$ plane of the inertial reference frame to be the one containing all the motion of this particular problem.

Now that all possible transfer orbit scenarios have been introduced from the point of view of geometry, it is possible to present the solution to the problem from an analytical point of view.

## 2.5 The analytical solution to the problem

Although Lambert is considered the be first one to formulate the BVP of the two-body two-point problem he only provided the relation between the time of flight and the geometry of the orbit. No procedure or solution is known to be published by him, even if he ever claimed this to Euler.

### 2.5.1 Lagrange's solution

Some years, after the publication of Lambert's book, Joseph Louis Lagrange published *Méchanique analitique* in which the first procedure to solve Lambert's problem appeared for the first time.

In the following lines, his method is presented so the reader has a modern vision on this classical solver. The procedure presented in here is the one introduced by Jiang, Chao, Wang, and Yang 2016, who carefully revised the algorithm. Only the

[8] For geostationary orbits, position vectors remain the same as seen from a frame centered and rotating with the attractor. However, the orbiting body is seen to move from a inertial frame.

fundamental equations of the problem will be collected here, so reader is encouraged to review Jiang's article for a deep understanding.

Recalling Kepler's equation and Lambert's theorem, it is possible to obtain equation 2.30 between the time of flight and the geometry of the problem:

$$\sqrt{\mu}\Delta t = F(\|\vec{r_1}\| + \|\vec{r_2}\|, \|\vec{c}\|, a) \tag{2.30}$$

where the function $F$ depends on the geometry of the orbit, which is not known at first sight. To do so, the time of flight for a pure parabolic transfer is computed using the first term of the series presented in 2.13 or in a more compact way:

$$\Delta t_p = \frac{1}{\sqrt{\mu}}\left(\frac{1}{6}\left(\|\vec{r_1}\| + \|\vec{r_2}\| + \|\vec{c}\|\right) \pm \left(\|\vec{r_1}\| + \|\vec{r_2}\| - \|\vec{c}\|\right)\right) \tag{2.31}$$

where the $\pm$ indicates if the transfer angle is lower or greater than $\pi$ respectively.

Knowing the parabolic time of flight, it is possible to know if the orbit being addressed is elliptic or hyperbolic by direct comparison with the current time of flight:

- If $\Delta t > \Delta t_p$, then the orbit is *elliptic*.
- If $\Delta t < \Delta t_p$, then the orbit is *hyperbolic*.

Once the nature of the transfer orbit is known, the value of $F$ can be replaced by the corresponding Kepler's equation:

$$F = \begin{cases} \sqrt{\mu}\Delta t_e = a^{\frac{3}{2}}\left(\left(\alpha - \sin\left(\alpha\right)\right) + \left(\beta - \sin\left(\beta\right)\right)\right) \\ \sqrt{\mu}\Delta t_h = \left(-a\right)^{\frac{3}{2}}\left(\left(\sinh\left(\alpha\right) - \alpha\right) + \left(\sinh\left(\beta\right) - \beta\right)\right) \end{cases} \tag{2.32}$$

where in equation 2.32, the angles $\alpha$ and $\beta$ were introduced by Lagrange. To compute these angles, first the auxiliary values $\alpha_0$ and $\beta_0$ are found using:

$$\alpha_0 = 2 \cdot \arccos\sqrt{\frac{s}{2a}} \qquad \beta_0 = 2 \cdot \arccos\sqrt{\frac{s - \|\vec{c}\|}{2a}} \tag{2.33}$$

for the elliptic case and:

$$\alpha_0 = 2 \cdot \operatorname{arccosh}\sqrt{\frac{s}{-2a}} \qquad \beta_0 = 2 \cdot \operatorname{arccosh}\sqrt{\frac{s - \|\vec{c}\|}{-2a}} \tag{2.34}$$

for the hyperbolic one.

Once computed, a correction needs to be performed depending on the transfer angle:

$$\alpha = \begin{cases} 0 \leqslant \alpha_0 \leqslant 2\pi & \text{for elliptic/hyperbolic orbits always} \end{cases} \tag{2.35}$$

and:

$$\beta = \begin{cases} 0 \leqslant \beta_0 \leqslant \pi & \text{for elliptic orbits if } \Delta\theta \leqslant \pi \\ -\pi \leqslant \beta_0 \leqslant 0 & \text{for elliptic orbits if } \Delta\theta > \pi \\ 0 \leqslant \beta_0 & \text{for hyperbolic orbits if } \Delta\theta \leqslant \pi \\ \beta_0 \leqslant 0 & \text{for hyperbolic orbits if } \Delta\theta > \pi \end{cases} \tag{2.36}$$

If reader is interested in the physical meaning of this angles, a deep study was made by Prussing 1979.

Finally, an initial guess about $a$ is made, named $a_0$. By substituting in the proper equation for the transfer geometry, the function for $F$ is compared with the current value of Kepler's equation for the real time of flight such that:

$$\sqrt{\mu}\Delta t - F = 0 \tag{2.37}$$

Equation 2.37 might not be equal to zero but close to this value. Therefore, once it below a desired accuracy, the proper value of $a$ has been solved and the values $p$, the semi-latus rectum, can be computed using:

$$p = \begin{cases} \frac{4a(s-\|\vec{r_1}\|)(s-\|\vec{r_2}\|)}{\|\vec{c}\|^2} \left(\arcsin\left(\frac{\alpha+\beta}{2}\right)\right)^2 & \text{for elliptic orbits} \\ \frac{-4a(s-\|\vec{r_1}\|)(s-\|\vec{r_2}\|)}{\|\vec{c}\|^2} \left(\operatorname{arcsinh}\left(\frac{\alpha+\beta}{2}\right)\right)^2 & \text{for hyperbolic orbits} \end{cases} \tag{2.38}$$

with this new orbit parameter, the rest can be computed. Finally, a simple transformation from COE to RV can be applied in order to get the initial and final position vectors although Jiang, Chao, Wang, and Yang 2016 provides direct expressions based on the semi-latus rectum.

# 3 Modern Lambert's problem solvers

This chapter is devoted to the study of modern Lambert's problem solvers and it is the core of this work. By introducing their origin and proposing a classification for them, the reader is expected to acquire a deep-knowledge in the state-of-the-art of the available solutions to the BVP problem.

## 3.1 Origin of modern solvers

The applications of Lambert's problem were presented in section 1.3, where it was seen that targeting and maneuvering were the most interesting ones, apart from IOD. However, it was not till the 60s, with the space-race, when the problem was applied with targeting and maneuvering purposes.

Missions requiring docking or rendezvous required to solve for the Lambert's problem as a first approach when solving for the phasing problem[9].

The space-race era shared its time with the development of the first electronic computers[10]. In fact, the first aerospace project/program to make use of integrated circuit boards (ICB) was the Apollo one, in particular in its guidance computer. In order to ease the usage of computers by scientists and engineers, a high-level programming language named Fortran[11] was created in 1975.

All previous facts, in the context of developing new procedures for the Lambert's problem, lead to what are known as *modern Lambert's solvers*. These new solvers were developed in the form of computer algorithms, taking advantage of different numerical methods for finding in a quick way the solution to the problem and avoiding its singularities. In addition, computers allowed to plot contour plots in a more easy and interactive way, giving a deep understanding not only on Lambert's problem but in many other ones.

The computational power grew originally at the rate predicted by Moore in his famous law[12], although nowadays the Huang's law applies[13] The effect of these laws

[9] Phasing problem can be explained as the problem which combines achieving a particular orbit but also a given true anomaly on it.

[10] Analog/mechanical, digital and quantum computers are the main three families.

[11] Fortran means *Formula Translating System* and became really popular between the 80s and 90s. Lots of legacy code are written in this language.

[12] Moore's law states that the amount of transistors per surface unit doubles every two years.

[13] Huang's law claims that GPU development is growing faster than that of the central processing units (CPUs), making Moore's law obsolete.

can be seen mainly in the lower amount of time a today's computer requires to execute a piece of code as opposite to the initial days of space-race and computers era. Nevertheless, there are still efforts to increase computers performance so more complex problems in many different scientific fields can be explored, understood and solved.

Therefore, it is not strange that the amount of published modern solutions, as early introduced by figure 1.5, has increased during the last decades and most of the new techniques and procedures try to beat the ones developed during the 60s.

## 3.2 Main elements and workflow

Each Lambert's problem solver makes use of its own mathematical background to find the solution to the problem. However, all algorithms share a common workflow in which a particular set of elements can be identified. Among these elements, it is possible to find the free-parameter, the numerical method, an initial-guess and the velocity vectors construction procedure.

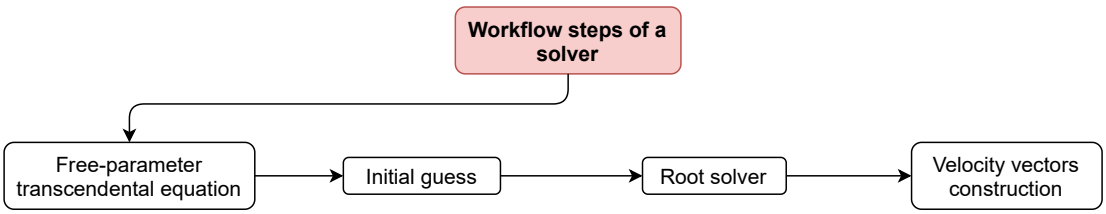


**Fig. 3.1:** Main elements forming the basic structure of any modern Lambert's problem solver.

It must be pointed out that some methods developed along history, which will be presented in future pages, solve the Lambert's problem by making use of a series expansion. These methods express the semi-major axis $a$ as a sum of infinite terms. One might think these do not follow previously cited structure but the truth is that the series expansion was performed around an arbitrary point, which is similar to the initial guess procedure. The accuracy of the method depends on the number of terms considered, so this can be understood as the numerical method used by the solver.

Hence, all algorithms found in literature share the elements presented in figure 3.1.

### 3.2.1 The free-parameter

The free-parameter is the independent variable, also named the iteration variable. This variable appears in an implicit way in a fundamental equation of the algorithm, meaning that a numerical method is required in order to solve for it. Once its value

has been computed, each algorithm will follow a particular approach to solve for the rest of the orbital elements.

Among history, several algorithms have been developed exploiting different free-parameters such us the semi-major axis $a$, the orbit's eccentricity $e$, the orbital parameter or semi-latus rectum $p$, the true anomaly $\nu$, the universal formulation and finally the Kustaanheimo-Stiefel formulation and the flight path angle $\gamma$.

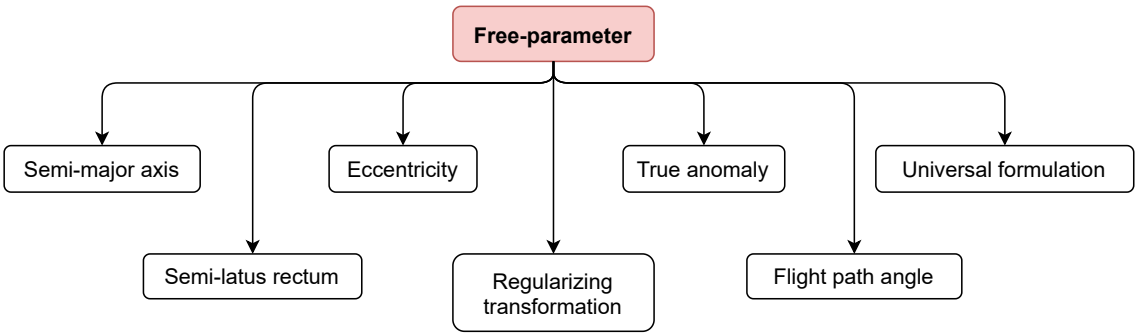


**Fig. 3.2:** All possible free-parameters employed for solving the Lambert's problem.

### 3.2.2 The initial guess

Before the iterative process can start, an initial guess is required. This is the starting value imposed to the free-parameter, so the closer it is to the solution, the lower the amount of iterations till the free-parameter converges to a desired accuracy.

Classic algorithms, all of those previous to the modern ones, made use of an arbitrary initial guess without developing any additional procedure. This is no longer the case with modern ones in which a more complex sub-routine is provided in order to have a reliable initial guess to reduce the computation time and faster achieve the solution.

The initial guess is dependent on the numerical method employed and thus, on the mathematical background too. A total of three common solutions have been identified in literature, being these the rational formulate, linear approach and bi-linear one, as depicted by diagram in figure 3.3.

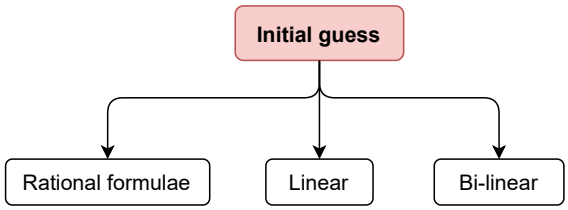


**Fig. 3.3:** Possible approaches to be used in the initial guess procedure.

The rational formulate (also named arbitrary initial guess) is based on the time of flight curves against the free-parameter. By identifying different regions and their

expected time of flight it is possible to compare those against the current one and estimate the solution.

Regarding the linear approach and bi-linear guesses, these simplify the free-parameter transcendental equation so that it becomes an explicit one. Although very useful, not all the equations for the free-parameter might be benefit from this initial guess.

It must be pointed that the level of complexity of initial guess procedure must not be detrimental from the point of view of the total computation time. Otherwise, it is absurd to waste such a valuable time instead of letting the root solver to perform the iteration workload.

### 3.2.3 The numerical method

The previously presented free-parameter is solved by means of a numerical method or root finder. Numerical methods for solving root equations are abundant in the literature, being some of the most common ones bisection, regula falsi, secant, Newton's method, Halley's method and Householder's one[14].

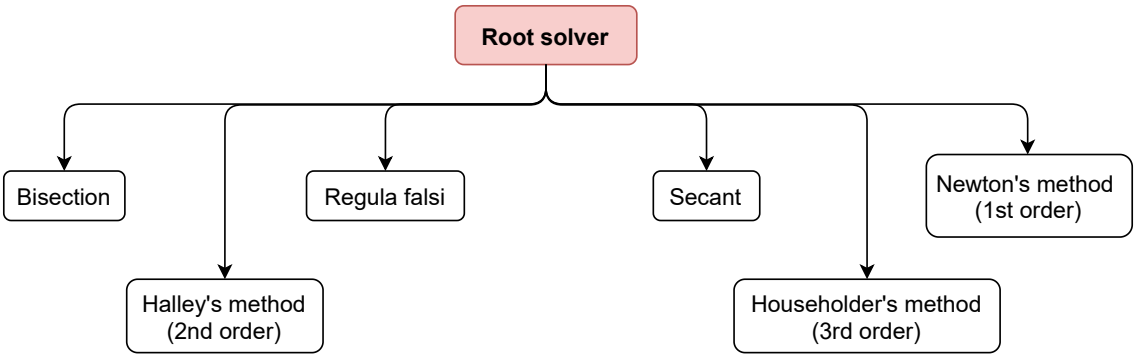


**Fig. 3.4:** Different root finders found in literature to solve for the free-parameter equation.

The last three of the listed methods are the most common ones employed by authors in modern solvers. However, these methods required from the derivative of the free-parameter transcendental equation with respect to the free-parameter itself. Equations 3.1, 3.2 and 3.3 present these three methods where $x_n$ and $x_(n+1)$ inditcate the current and next free-parameter value, $f$ is the function to be evaluated and $f'$, $f''$ its first and second derivatives.

$$\text{Newton's method} \qquad x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{3.1}$$

$$\text{Halley's method} \qquad x_{n+1} = x_n - \frac{2f(x_n)f'(x_n)}{2\left[f'(x_n)\right]^2 - f(x_n)f''(x_n)} \tag{3.2}$$

$$\text{Householder's method} \qquad x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}\left(1 + \frac{f(x_n)f''(x_n)}{2\left[f'(x_n)\right]^2}\right) \tag{3.3}$$

[14]Notice that Newton's and Halley's methods are just Householder's one but of lower order.

Previous methods are preferred due to its performance and accuracy. However, not all the transcendental equations have a simple derivative. In most of the cases this one becomes quite complex and authors end up using the secant method. Not only that, previous methods are singular for values of $x_n$ such that $f'(x_n) = 0$.

When previous situation applies, some authors impose a change in the root solver so that bisection or regula-falsi methods are used. These two methods, once the solution has been bounded, guarantee the convergence to the right value of the free-parameter. However, they rate of converge is slower if compared against Newton's, Halley's or Householder's method.

### 3.2.4 The velocity vectors construction

The last sub-routine employed by any Lambert's solver is the computation of the velocity vectors. One might thought that the computation of COE elements is also valid but, since we are given as initial input two position vectors, it is reasonable that the user expects the velocity vectors as output so the RV set is completely defined.

There are different approaches followed by modern solvers to compute the value of these vectors once the free-parameter has been solved such us using the radial and tangential components, $f$ and $g$ formulation[15] introduced in Bate, Mueller, White, and Saylor 1971 or applying a COE to RV conversion.

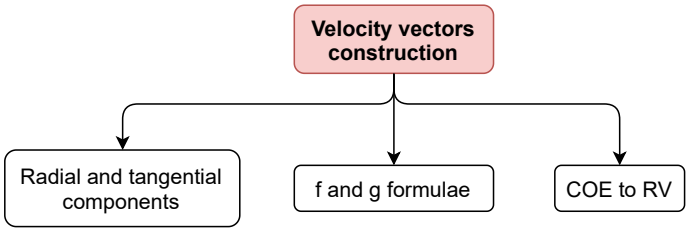


**Fig. 3.5:** Possible ways of constructing the initial and final velocity vectors.

The radial and tangential components can computed from equation 3.4, where $h_0$ is obtained via previously presented 2.2 and the values of $v_{r_1}$, $v_{r_2}$, $v_{t_1}$ and $v_{t_1}$ computed using a particular solver.

$$\vec{v_1} = v_{r_1}\frac{\vec{r_1}}{\|\vec{r_1}\|} + v_{t_1}\frac{\vec{h_0}\times\vec{r_1}}{\|\vec{h_0}\times\vec{r_1}\|} \qquad \vec{v_2} = v_{r_2}\frac{\vec{r_2}}{\|\vec{r_2}\|} + v_{t_2}\frac{\vec{h_0}\times\vec{r_2}}{\|\vec{h_0}\times\vec{r_2}\|} \tag{3.4}$$

Another common way of computing the velocity vectors at the initial and final points is to use the $f$ and $g$ formulation, see 3.5:

$$\vec{v_1} = (\vec{r_2} - f\vec{r_1})/g \qquad \vec{v_2} = (\dot{g}\vec{r_2} - \vec{r_1})/g \tag{3.5}$$

[15] This is also known as Gauss' $f$ and $g$ formulae.

where the values for the $f$, $g$, $\dot{f}$ and $\dot{g}$ can be obtained using the following relations:

$$f = 1 - \frac{a}{\|\vec{r_1}\|}\left(1 - \cos\left(\Delta E\right)\right) \tag{3.6}$$

$$g = \Delta t - \sqrt{\frac{a^3}{\mu}}\left(\Delta E - \sin\left(\Delta E\right)\right) \tag{3.7}$$

$$\dot{f} = \frac{-\sqrt{\mu a}}{\|\vec{r_1}\|\|\vec{r_2}\|} \tag{3.8}$$

$$\dot{g} = 1 - \frac{a}{\|\vec{r_2}\|}\left(1 - \cos\left(\Delta E\right)\right) \tag{3.9}$$

Finally, some solvers just computed the COE set, so a COE to RV conversion needs to be performed. The mathematical background for this conversion is provided in various works such us Bate, Mueller, White, and Saylor 1971, Vallado 2013 or Curtis 2020.

## 3.3 Classification and inheritance diagram

The amount of devised solutions along Lambert's problem timeline is massive. However, after spotting some common points between different solvers, it is possible to classify them according to the free-parameter employed, the numerical method used, the initial guess procedure or the velocity vectors construction. Among these four, the classification based on the free-parameter is the most useful one, as it also allows to track the evolution of a particular solver.

In addition to the classification presented in the following lines, an inheritance diagram has been produced so reader can have a better understanding about the relations between the different published solvers over time. The work by De la Torre 2020 has been extremely useful and expanded.

### 3.3.1 Semi-major axis based solvers

The first methods to solve for the Lambert's problem made use of the semi-major axis as the free-parameter, see previously introduced Lagrange solution in section 2.5.1. In addition, this variable is directly involved in the Lambert's theorem (see section 2.3).

However, the analysis made by Battin 1999 demonstrates that iterating over the semi-major axis is not convenient for the case of multiple revolutions, where a pair of conjugate orbits exist. In addition to this, the derivative of the transfer time becomes singular when the semi-major axis $a = a_m$, being $a_m = s/2$ the semi-major axis of the minimum energy orbit.

The first algorithm found in literature is the one by Lagrange 1788. This author set the basis for others to improve on his method. In particular, Prussing 2000 expanded the solution to the multi-revolution case and Wailliez 2014 improved the convergence of the method by making use of a Householder's root solver over a simple-semi analytical expression for the time of flight. Finally, Jiang, Chao, Wang, and Yang 2016 revisits the problem for both the elliptic and hyperbolic transfer types.

However, another branch within the semi-major axis solvers was started by J. Thorne 1995 when this author devised a series solution for the Lambert's problem. This algorithm was improved by the same author some years later (see J. D. Thorne 2004). A convergence analysis of the series was made again by him in J. D. Thorne 2015.

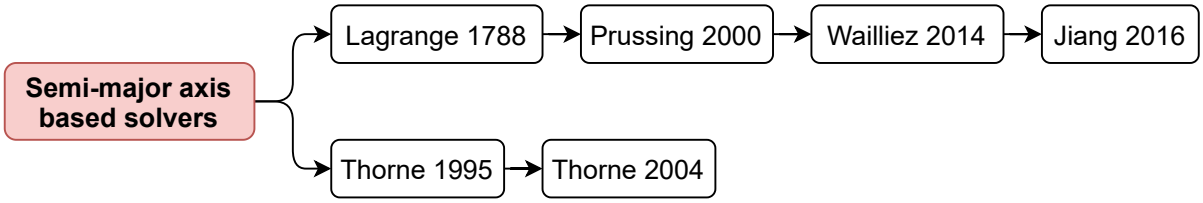


**Fig. 3.6:** Inheritance diagram for semi-major axis based solvers. Two branches exist: the one devised by Lagrange and the series-based one by Thorne.

### 3.3.2 Eccentricity based solvers

Lambert's solvers which iterative over the eccentricity $e$ of the transfer are relatively new if compared to other solvers. The first algorithm of this type in history seems to be the one developed by Escobal 1965. However, **Battin 1999** introduced in his book an important property of the eccentricity vector within the context of Lambert's problem: its projection along the chord-wise direction is kept constant.

Previous statement can be proof from the orbit equation, where $\vec{e} \cdot \vec{r} = p - r$. If evaluated at the two known position vectors, then $\vec{e} \cdot (\vec{r_2} - \vec{r_1}) = \|\vec{r_1}\| - \|\vec{r_2}\|$. Because $\vec{c} = \vec{r_2} - \vec{r_1}$, and the norm of the vectors does not change over time, the projection $\vec{e} \cdot \vec{c} = e_c$ is seen to be constant. Some authors also refer to $e_c$ as $e_F$.

The first author to devise a method based on previous property was Avanzini 2008, introducing a simple algorithm which iterates over the transverse component of the eccentricity vector $e_T$, such that $e = \sqrt{e_c^2 + e_T^2}$. However, this algorithm was improved by He, Li, and Han 2010 who expanded to the multi-revolution case and provided the derivative of Kepler's equation with respect to the free-parameter. Finally, new improvements were made by Wen, Zhao, and Shi 2014 reducing the computational cost and therefore, increasing the overall performance.

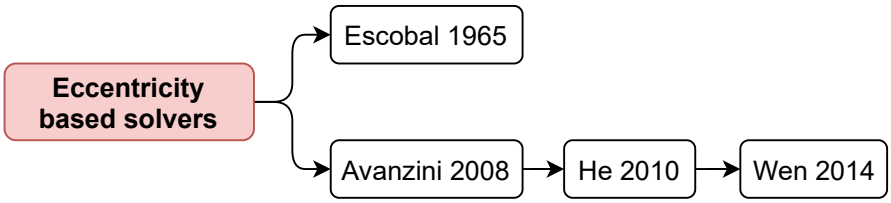


**Fig. 3.7:** Inheritance diagram for eccentricity based solvers. Notice that the branch started by Avanzini is the most modern one and the mother of modern solvers based on this orbit parameter.

### 3.3.3 True anomaly based solvers

Regarding solvers which iterate over the true anomaly of the transfer orbit, only a single one has been found among the whole Lambert's problem literature: the one devised by Gunkel, Lascody, and Merrilees 1960. This algorithm also appears in the book by Escobal.

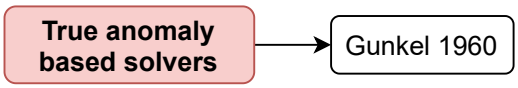


**Fig. 3.8:** Inheritance diagram for true anomaly based solvers. Only one solver is found to belong to this set of solvers as iterating over $\nu$ is not the most popular way of addressing the problem.

### 3.3.4 Semi-latus rectum based solvers

The semi-latus rectum $p$ of an orbit is usually called the orbital parameter and it is related with the semi-major axis of the orbit and its eccentricity via $p = a(1 - e^2)$. Therefore, the semi-latus rectum has been used by some authors as the free-parameter.

We might establish Gauss 1809 the first one to use this parameter. Although the original algorithm proposed by the genius does not iterate over this variable, it is the first orbit element to be obtained after the numerical process. Gauss relates the triangle to area sector to obtain a pair of equations from which the Lambert's problem can be solved. In fact, he used the method to the discovery of Ceres' orbit one year after it was spotted for the first time. However, this method is only valid for angles lower than 90 degrees approximately and it is singular for transfer angles of 180 degrees.

Some years later, Battin and Vaughan 1984 improved the convergence of Gauss' algorithm by expanding it to all types of orbit in a universal variable approach. De la Torre includes this algorithm in the universal branch of solutions but we decided to included it here just to point out its inheritance from the one devised by Gauss. The algorithm by Battin moves the singularity from the 180 to 360, which is a more reasonable corner case. In addition, it improves by far the accuracy of

Gauss solver. Finally, Battin's algorithm was improved to also solve for the multi-revolution scenario once the work of Shen and Tsiotras 2003 was published.

A new branch based on the $p$ parameter was started by Bate, Mueller, White, and Saylor 1971. Among the set of algorithms published in Bate's book, the one iterating over the semi-latus is simple to implement. However, this author did not imposed a particular numerical method to solve for the transcendental equation. The only method found in literature inheriting from Bate's one is the solver devised by Alhulayil, Younes, and Turner n.d. in which the authors make use of a 4th Taylor series expansion. Although they do not cite Bate's solver, the variables used by them are clearly the ones from Bate's book.

The last branch was started by Herrick and Liu 1959. This differs from Bate's solver as it involves the eccentricity during the computation of the free-parameter.

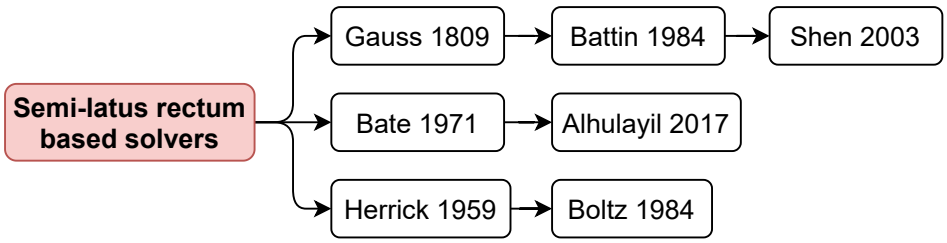


**Fig. 3.9:** Inheritance diagram for semi-latus rectum based solvers. A total of three branches have been devised along history for addressing the problem making use of this free-parameter.

### 3.3.5 Universal formulation based solvers

The usage universal formulation as the free-parameter holds the greatest amount of solvers and the deepest inheritance diagram. In fact, this approach to solve the problem is considered to be the origin of modern Lambert's solvers with the publication of Lancaster and Estes 1970 in which an elegant figure relating the free-parameter and the non-dimensional time of flight appeared for the first time. This algorithm was able to cover all types of transfer orbits (elliptic, parabolic and hyperbolic) and also capable of solving the multi-revolution problem.

Lancaster set the basis for other authors to improve the method and Gooding 1990 pick up the baton by introducing one of the most popular algorithms. In his work, Gooding provided Fortran77 routines so other authors could easily reproduce his work. However, he imposed a total of three iterations after checking that no more were required to obtain great accuracy in the results.

Even when this branch was not expected to be improved, Izzo 2015 proposed his famous solver by introducing one last change of variable and upgrading the numerical method employed to a Householder's one, as opposite to Gooding who employed Halley's method.

The second branch within solvers using universal formulation was started by Bate, Mueller, White, and Saylor 1971. Again, in his book, this author introduced a new solver. No root solver was imposed although bisection and Newton's method were cited. However, some years later Vallado 2013 improved the convergence of the method by imposing a bisection method which, as opposite to Newton's method, converges always to the expected value.

Finally, a great progress was made by Arora and Russell 2013 by introducing an smart cosine transformation. This new algorithm was capable of solving the multi-revolution problem and employed an strong rational formulae initial guess.

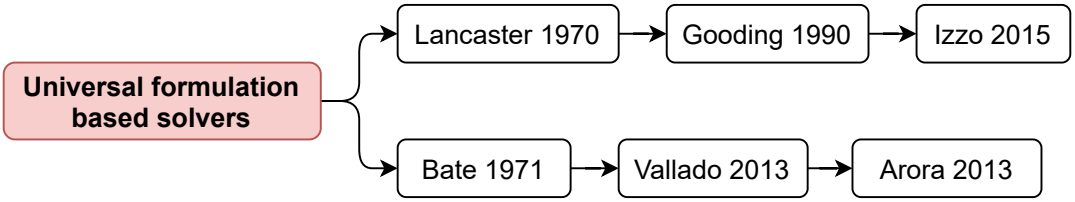


**Fig. 3.10:** Inheritance diagram for universal formulation solvers based solvers. The two branches initiated by Bate and Lancaster are visible.

### 3.3.6 Regularizing transformation based solvers

Solvers based on regularized transformations are not abundant in literature. The main goal of these solvers is to avoid the singularities in the Lambert's problem by moving it from $\Re^N \rightarrow \Re^{N+1}$. At least three types of regularizing transformations have been used in the Lambert's problem:

- Levi-Civita.
- Kustaanheimo-Stiefel.
- Radial-Inversion.

The mathematical background for the regularizing transformation is out of the scope of this work but reader is encouraged to refer to Chapter 3 of Celletti 2002 book's.

The first known solver to take advantage of one of the transformations is Simó 1973. The algorithm by this author makes use of the Levi-Civita, covering degenerate conics too. This algorithm was enhanced by De La Torre, Flores, and Fantino 2018 by introducing the multi-revolution scenario and enhancing the initial guess routine.

Another branch found in literature using the regularizing transformation was started by Kriz 1976. As opposite to previous solvers, this algorithm makes use of the Kustaanheimo-Stiefel transformation. That same year, Jezewski 1976 published another solver exploiting the same principle as the one suggested by Kriz.

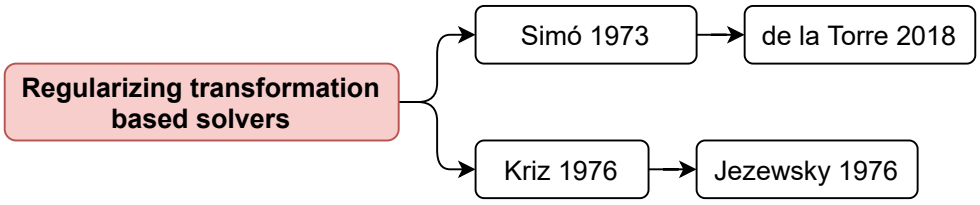


**Fig. 3.11:** Inheritance diagram for regularized transformation solvers. Only the transformations via Levi-Civita and Kustaanheimo-Stiefel are the ones serving as base for these type of solvers.

### 3.3.7 Flight path angle based solvers

The flight path angle $\gamma = \arccos\left(\frac{r_p v_p}{rv}\right)$, where the sub-index $p$ denotes the perpendicular direction to the current radial distance, is the last of the variables used as free-parameter. We already discussed about the applications of Lambert's problem within the rendezvous and intercepting maneuvers. Algorithms using the flight path angle were usually associated with ballistic missiles, benefiting from previous commented applications.

Wheelon 1959 was the first to publish a solver using the flight path angle. He published his article not referring to the Lambert's problem but to the determination of the trajectory of ballistic missiles. Therefore, this is a case were the IOD problem is solved via the BVP.

Nevertheless, some years later Nelson and Zarchan 1992 published his article about the solution of Lambert's problem explicitly. By iterating over the flyout angle, Nelson's solver computes the initial velocity vector so the full orbit is determined.

Another contribution to this set of solvers was made by Arlulkar and Naik 2011. This algorithm inherits from both previous ones. He also provided the multi-revolution scenario.

Finally, the last algorithm of this set was devised by Ahn and Lee 2013. Although using the flight path angle as free-parameter, the main feature of this solver is the usage of analytic gradients as the way of computing the solution.

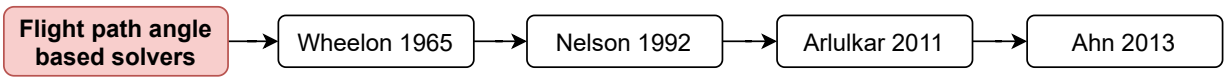


**Fig. 3.12:** Inheritance diagram for flight path angle based solvers. Only a single branch is known within this set of solvers.

## 3.4 Inputs and outputs of a Lambert's solver computer algorithm

At the very first section of Chapter 2, see 2.1, the statement of Lambert's problem. However we only discussed it from the mathematical point of view and not from a computational one. In this section, the input and output parameters that any Lambert's solver algorithm must accept and return are exposed and justified.

### 3.4.1 Input parameters

The input parameters of a solver are all the variables which the algorithm requires in order to perform the computation. Among these parameters we can find the following ones:

- **Gravitational parameter** $\mu$: equivalent to $MG$, that is the mass of the attracting body time the gravitational constant. It can be understood as the strength of the Newtonian gravity field created by the attractor. It is a floating value.
- **Initial position vector** $\vec{r_1}$: the first observation vector as seen from an attractor centered inertial frame. Modern computers deal with vector representation in the form of pointers, lists or arrays. Three components are required, as the problem belongs to $\Re^3$.
- **Final position vector** $\vec{r_2}$: the second observation vector, as seen from an attractor centered inertial frame too. Similarly to the initial vector, three components are required.
- **Time of flight** $\Delta t$: the time of flight between the initial and final position vectors. Should be of the type *float*.
- **Number of revolutions** $M$: if the direct transfer is desired, the value of $M = 0$ while for the multi-revolution scenario, the minimum number of revolutions is given by $M \geq 1$. Therefore, this parameter only accepts *integer* types.
- **Sense of motion**: this boolean[16] parameter indicates if the orbit is prograde or retrograde so the proper normal vector to orbital plane can be identified together with the transfer angle. It is advisable to name this parameter as *is_prograde* so the user is aware of its boolean nature.
- **Type of path**: when dealing with the multi-revolution case, when a solution is found, two possible paths exist. Authors such us Sun 1977 usually refer to these two paths as *lower* or *higher* one. Similarly to the sense of motion, this is a boolean variable used to filter out the solution. The advice is to name this variable *is_low*.

[16] Boolean parameters only have two possible values: *True* or *False*.

- **Maximum number of iterations**: if exceeded, the algorithm did not converged to desired tolerance or simply diverged. This parameter is of the *integer* type.
- **Absolute tolerance**: this floating value is used to check if the free-parameter value computed after a given iteration is within the expected limits.
- **Relative tolerance**: due to floating point errors, a relative tolerance is required to properly identify if the solution is within the expected limits of error, see Dawson 2021 and Ericson 2008.

### 3.4.2 Output parameters

Regarding the outputs of a computer solver, only a pair of vectors are expected, being each one the initial and final velocity ones.

- **Initial velocity vector $\vec{v_1}$:** it is used to define the orbit together with the initial position vector. Again, a pointer, array or list must be used to store its components.
- **Final velocity vector $\vec{v_2}$:** it is useful for rendezvous applications and checking that the orbit holds both the final position and this one after being propagated the time of flight.

Figure 3.13 a Lambert's algorithm is being executed with all the previously presented input parameters. The name for the routine is composed by the solver's author and the year of publication.

```
(v1, v2) = authorXXXX(
    mu, # The gravitational parameter, product of GM
    r1, # Initial position vector [r1x, r1y, r1z]
    r2, # Final position vector [r2x, r2y, r2z]
    tof, # Observed/desired time of flight
    M=0, # Number of revolutions
    is_prograde=True, # Observed/desired sense of motion
    is_low=True, # For filtering out the solution if two are found
    maxiter=35, # Maximum number of iterations
    atol=1e-5, # Abslute tolerance of the numerical method
    rtol=1e-7, # Relative tolerance of the numerical method
)
```

**Fig. 3.13:** A terminal emulator showing the execution of a Lambert's problem solver.

### 3.4.3 Note on the usage of units in computer programs

In the orbital mechanics and astrodynamics subjects it is advisable to make use of quantities with the same characteristic order of magnitude in order to avoid integration problems. However, there is a better approach: use canonical formulae of the problem. By doing so, every problem can be translated to a common space in which no units are required to perform the iteration workload. Only in the last final step, the computed solution values are translated into the original problem space. This reduces the propagation of errors during the computation. Reader is encouraged to refer to Wiesel 2010, who devotes a section to this topic in his book about modern astrodynamics.

Another option is to let the user to manage the units. For example, if the gravitational parameter $\mu$ has [km3/s2], the input values for the position vectors $\vec{r_1}$ and $\vec{r_2}$ should be in [km] and the time of flight $\Delta t$ in [s]. These input quantities will return the velocity vectors $\vec{v_1}$ and $\vec{v_2}$ in [km/s], a common unit used for maneuvering speeds.

# 4 Introduction to selected solvers

The selected solver are presented in a detailed way in this chapter. The fundamental equations relating the non-dimensional time of flight with the free-parameter are provided for each one together. By plotting those for various transfer angles, it is possible to build the so-called *time of flight curves* also included for each solver. Finally, a resume table at the end of the chapter collects the main elements used by every of the selected solvers so the comparison among them is easier for the reader.

## 4.1 Selected solvers

In order to carry out the performance comparison, a set of solvers is required. Implementing all the algorithms cited during the 3.3 section would be not only a huge task but also a non optimum one. The reason is that some of the cited solvers already include a brief comparison against other similar or popular solver.

After a deep review and study of Lambert's problem bibliography, the following solvers were selected: Gauss 1809, Battin and Vaughan 1984, Gooding 1990, Avanzini 2008, Arora and Russell 2013, Vallado 2013 and Izzo 2015. From now on, the first author of each solver together with its publication date will be used to reference the algorithm.

In the sub-sections below these lines, a deep analysis on each one of these solvers is made and the justification of its selection is presented to the reader. The *curves for the time of flight* are included too. These provide the relation between the free-parameter and the non-dimensional time of flight or the dependent variable.

### 4.1.1 Gauss 1809

The algorithm devised by Gauss in 1809 became very popular in its days. Gauss developed an algorithm based on three observed angles to compute the orbit of Ceres, see Beďatš 2021 and provided one for solving Lambert's problem.

His algorithm exploited the ratio between the sector triangle areas. By its time, it was considered to be a great improvement within initial orbit determination subject. However, the algorithm is also known for its low accuracy when the transfer angle exceeds around 90 degrees although performs well for lower transfer angles and times.

In this subsection, only the fundamental equations required for solving this method will be presented. However, if reader wants to review the original work made by Gauss, then refer to his famous book *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*. A modern revision of the problem was made by Teets and Whitehead 1999 using the original notation when possible. Let us present in the following lines, the basic expressions for the method devised by Gauss. However, here the background presented by Bate, Mueller, White, and Saylor 1971 is used.

The independent variable employed by Gauss is named $x$, while the dependent one is $y$. These two variables are related via the so-called *two equations of Gauss*, being those:

$$x = \frac{w}{y^2} - s \qquad y = 1 + X(s + x) \tag{4.1}$$

where in previous equation, the auxiliary variables are the semi-perimeter $s$ and a variable $w$ related with the non-dimensional transfer time:

$$s = \frac{\|\vec{r_1}\| + \|\vec{r_2}\|}{2\sqrt{\|\vec{r_1}\|\|\vec{r_2}\|\cos\left(\frac{\Delta\theta}{2}\right)}} - \frac{1}{2} \qquad w = \frac{\mu\Delta t^2}{\left(2\sqrt{\|\vec{r_1}\|\|\vec{r_2}\|}\cos\left(\frac{\Delta\theta}{2}\right)\right)^3} \tag{4.2}$$

Finally, $X$ is given by the expression developed in Moulton 1970:

$$X = \frac{4}{3}\left(1 + \frac{6}{5}x + \frac{6\cdot 8}{5\cdot 7}x^2 + \frac{6\cdot 8\cdot 10}{5\cdot 7\cdot 9}x^3 + ...\right) \tag{4.3}$$

For solving the value of $x$, an initial guess about $y$ needs to be made, such that $y_0 = 1.00$ for example. Then, the value of $x(y_0)$ is solved via equation 4.1 so a new value of $X(x)$ can be computed. Finally, a new value of $y(x)$ is found. This value is compared with the initial one $y_0$ used for the iteration. If $\|y - y_0\| < \text{atol}$, then the method has converged and the value for the free-parameter has been found. In figure 4.1, the relation between the $x$ and $y$ parameter is shown graphically for various transfer angles. The singular ones, such us any multiple of $2\pi$ and $\pi$ have been omitted.

Finally, Bate, Mueller, White, and Saylor 1971 suggests to use the semi-latus rectum for computing the $f$ and $g$ functions, so the velocity vectors at the initial and final positions can be obtained.

The decision to include this solver within the performance comparison was based on two facts: the main one is its classic nature, so the solutions provided by modern solvers can be better seen, the later is that Battin and Vaughan 1984 improved this solver.

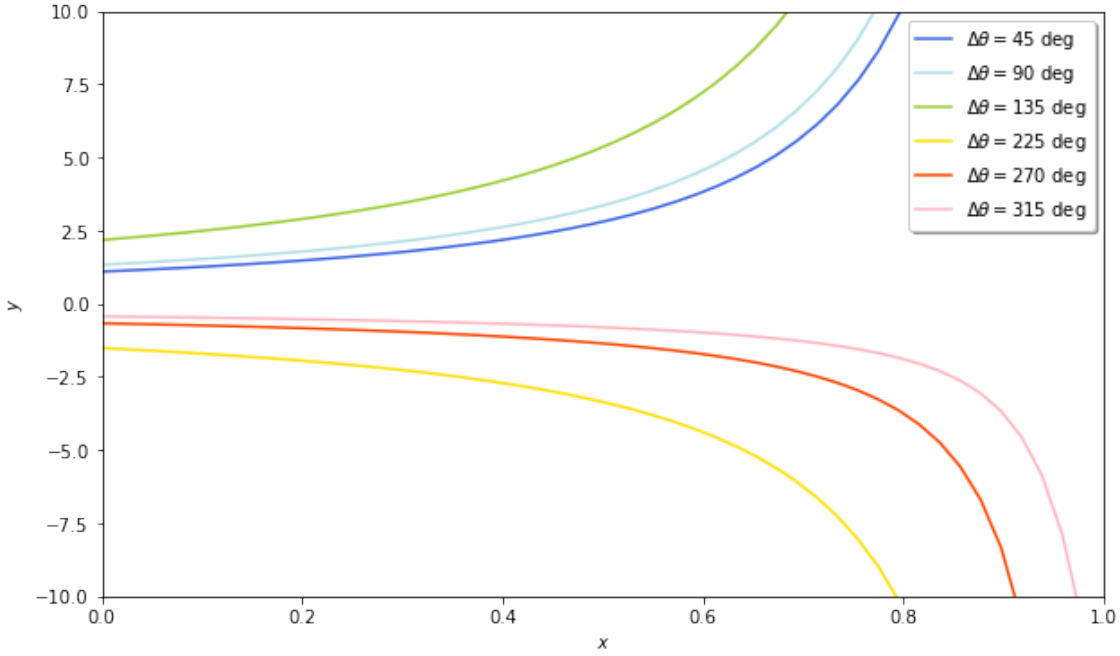


**Fig. 4.1:** Gauss 1809 time of flight curves for various transfer angles. This figure was obtained for $\rho = r_2/r_1 = 2$ with unitary $\mu$ and $\Delta t$.

### 4.1.2 Battin 1984

Batin is one of the author's who has devoted a lot of time reviewing and studying the Lambert's problem. Chapters 6 and 7 from his book *An introduction to the mathematics and methods of astrodynamics* cover the whole geometry of the problem while provide a new algorithm following the approach initiated by Gauss. Even if appearing in the book, this algorithm is the result of a PhD thesis developed by Vaughan, under the supervision of Battin, see Vaughan 1983. The algorithm was published by both just a year later in Battin and Vaughan 1984 and it is usually known as the Battin-Vaughan solver.

This new algorithm moves the singularity for transfer angles of 180 degrees to the less usual case of 360 degrees. Although it no longer makes use of ratio of sector to triangle areas, the notation used in the equations and the solution procedures is quite similar to the solver devised by Gauss. No only that, this solver is capable of fully working with all possible geometries and it is known to have a better accuracy and robustness than Gauss one.

The so-called Battin's first and second equation, again following the ideas introduced by Gauss, are given by expressions in 4.4:

$$x = \sqrt{\left(\frac{1-l}{2}\right)^2 + \frac{m}{y^2}} - \frac{1+l}{y^2} \qquad y^3 - y^2 - h_1(x)y^2 - h_2(x) = 0 \tag{4.4}$$

where the auxiliary variables are given by 4.5:

$$\lambda = \pm\frac{\sqrt{s(s-c))}}{s} \qquad l = \left(\frac{1-\lambda}{1+\lambda}\right)^2 \qquad m = \frac{8\mu\Delta t^2}{s^3(1+\lambda)^6} \tag{4.5}$$

and the $h$ functions can be evaluated using expression 4.6:

$$h_1 = \frac{(l+x)^2(1+3x+\xi)}{(1+2x+l)(4x+\xi(3+x))} \qquad h_2 = \frac{m(x-l+\xi)}{(1+2x+l)(4x+\xi(3+x))} \tag{4.6}$$

being $\xi$ computed from a continued fraction defined in the original Battin's article but simplified to:

$$\xi(x) = \begin{cases} \frac{4x(1-\mathrm{ATANR}(x))}{(3+x)\mathrm{ATANR}(x)-3} & x \neq 0 \\ 5 & x = 0 \end{cases} \tag{4.7}$$

where the function ATANR is an hypergeometric function defined as:

$$\mathrm{ATANR} = F(\frac{1}{2}, 1, \frac{3}{2}, -x) = \begin{cases} \frac{\operatorname{arctanh}\sqrt{-x}}{\sqrt{-x}} & -1 < x < 0 \\ 1 & x = 0 \\ \frac{\arctan\sqrt{x}}{\sqrt{x}} & x > 0 \end{cases} \tag{4.8}$$

following Allen and Wenzel 2015 article. This approach is cited here as implementing continued fractions in a computer might be a bit tricky, leading to non-optimized code. Modern computers do not struggle with the computations of trigonometric functions as in the old days.

Battin also improved previous equations at the end of his report to improve even more the convergence of the method, so reader is encouraged to review them. The implementation of the solver corresponds to these cited approach. The procedure when solving the free-parameter $x$ is the same one as for the Gauss solver. Curves for the time of flight are given figure 4.2.

This solver was selected considering its enhancements over the Gauss one. Not only that, it is one of the most moderns of the semi-latus rectum based solvers, see subsection 3.3.4.

### 4.1.3 Gooding 1990

The algorithm by Gooding 1990 was considered to have a high performance, as claimed in performance reports of its days such us Klumpp 1999. This is the main reason behind its selection for the performance comparison.

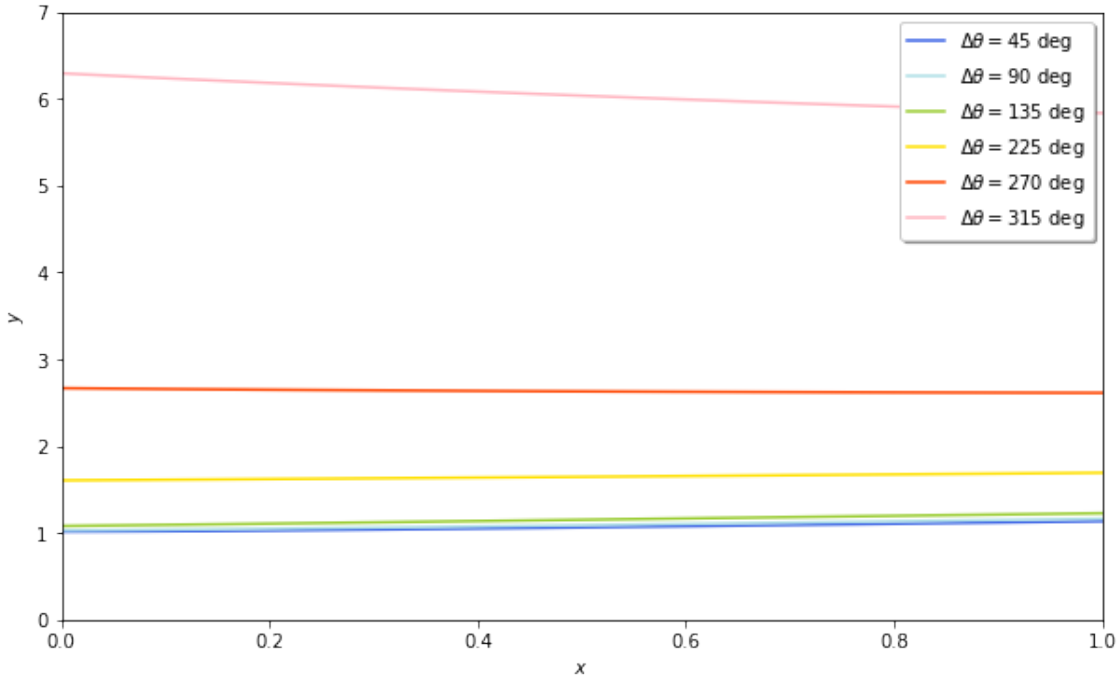


**Fig. 4.2:** Battin 1984 time of flight curves for various transfer angles. This figure was obtained for $\rho = r_2/r_1 = 2$ with unitary $\mu$ and $\Delta t$.

Gooding's solver is based on the universal formulation branch, presented in 3.3.5. This formulation enables to solve for elliptic, parabolic or hyperbolic orbits without splitting the workflow into an excessive amount of sections. Gooding's inherit from Lancaster and Estes 1970 solver and dramatically improves its convergence and accuracy up to thirteen decimal places.

The equation relating the canonical time of flight, named $T$ in Gooding's article, and the free-parameter $x$ is complex. The equations collected by Torre Sangrà and Fantino 2015 have been very useful and presented in 4.9:

$$\Delta t = \begin{cases} 4/3(1-q^3) & \text{Parabolic solution} \\ \sigma_x(f(x)) - q^3\sigma_x(q^2 f(x)) & \text{Near-parabolic} \\ \frac{2}{E(x)}\left(x - \lambda z(x) - \frac{d(x)}{y(x)}\right) & \text{Lancaster and Blanchard} \end{cases} \tag{4.9}$$

Gooding's solver makes use of a bi-linear initial guess, which makes the initial iteration value for the free-parameter to be closer to the final solution. Together with the usage of a Halley's method, the convergence of the method is quite fast. The construction of the velocity vectors is made using the radial and tangential components. The curves for the time of flight, in figure 4.3, are exactly the same ones as Lancaster and Estes 1970, since Gooding only devised and improved algorithm and did not introduce any additional steps.

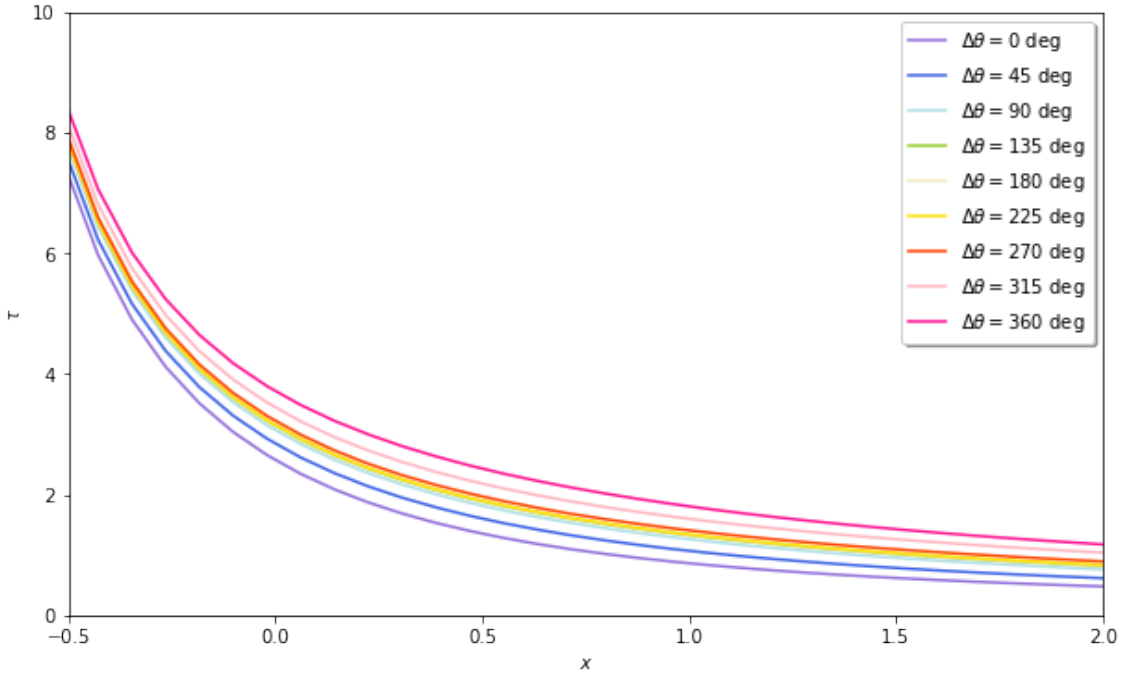


**Fig. 4.3:** Gooding 1990 time of flight curves for various transfer angles. This figure was obtained for $\rho = r_2/r_1 = 2$ with unitary $\mu$ and $\Delta t$. Only the direct transfer region is shown for the purposes of this work.

### 4.1.4 Avanzini 2008

Among the different Lambert's problem solvers, the one devised by Avanzini 2008 truly unleashes the geometry of the problem. This solver is the very first of its branch within the set of eccentricity based ones, see subsection 3.3.2. By exploiting the conservation of the eccentricity vector projection onto the chord one, it is able to solve for the problem no matter the geometry of the orbit.

Avanzini's starts defining the fundamental eccentricity of the orbit $e_F$ (the component along the chord direction) and the transverse one $e_T$. This is the one which will be used as free-parameter. The relations given by 4.10 apply:

$$e_F = \frac{\|\vec{r_1}\| - \|\vec{r_2}\|}{\|\vec{c}\|} \qquad \|\vec{e}\| = \sqrt{e_F^2 + e_T^2} \tag{4.10}$$

The free-parameter is defined as:

$$x = \begin{cases} \frac{e_P e_H}{e_P + e_H} \log\left(\frac{e_P(e_T+e_H)}{e_H(e_P-e_T)}\right) & \text{if } \Delta\theta < \pi \\ -e_P \log\left(1 - \frac{e_T}{e_P}\right) & \text{if } \Delta\theta > \pi \end{cases} \tag{4.11}$$

where $e_P$, and $e_H$ can be computed using expressions 4.12:

$$e_P = \sqrt{1 - e_F^2} \qquad e_H = \sqrt{(e_{\max}^2 - e_F^2)} \qquad e_{\max} = \frac{-1}{\cos \Delta\theta/2} \tag{4.12}$$

with all previous relations, it is possible to estimate the value of the transverse eccentricity from the free-paramter $x$ by making use of equation 4.13

$$e_T = \begin{cases} e_P e_H \frac{X-1}{e_p + e_H X} \text{if } \Delta\theta < \pi \\ e_P \left(1 - \exp\left(\frac{-x}{e_P}\right)\right) \end{cases} \tag{4.13}$$

being $X$ defined as:

$$X = \exp\left(\left(\frac{1}{e_H} + \frac{1}{e_P}\right) x\right) \tag{4.14}$$

The starting value to be used for $x$ was imposed by Avanzini to be $x_0 = 0$. The root solver declared by this author was the secant method, see subsection 3.2.3.

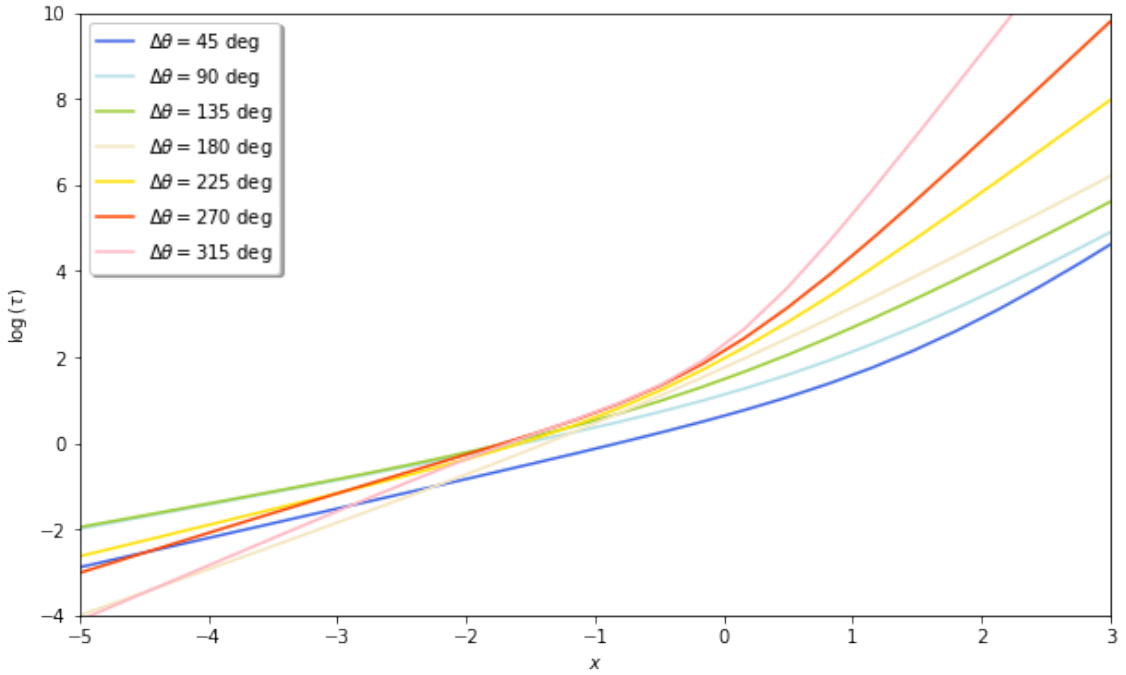


**Fig. 4.4:** Avanzini 2008 time of flight curves for various transfer angles. This figure was obtained for $\rho = r_2/r_1 = 2$ with unitary $\mu$ and $\Delta t$.

Once the value of $x_0$ is known, the one for $e_T$ can be computed and then the eccentricity of the orbit $e$, as the fundamental component is already known. Therefore, to check if the value of $x_0$ is the right one, the current time of flight and the one coming from the evaluation of Kepler's equation are compared within some tolerance. As happens with other solvers, if the distance between the two values is excessive, a

correction is made to the free-parameter and a new iteration begins till the method converges.

This solver was included due to its simplicity together with the fact of being the first one of its class.

### 4.1.5 Arora 2013

Some years ago, Arora and Russell 2013 published a new Lambert's solvers based on a cosine transformation. The new algorithm was strongly influenced by Bate, Mueller, White, and Saylor 1971 one and compared in performance against Gooding 1990 one in Arora's publication. This solver belongs to the 3.3.5 set.

Arora introduced the $k$ free-paramter, which is related to the eccentric and hyperbolic anomalies. The canonical time of flight can be computed as:

$$\text{TOF} = S\sqrt{1-k\tau}(\tau + (1-k\tau)W) \tag{4.15}$$

where the auxiliary parameters $\tau$, $S$ and $W$ are obtained via:

$$S = \sqrt{\frac{(\|\vec{r_1}\| + \|\vec{r_2}\|)^3}{\mu}} \qquad W = \frac{q}{\sqrt{m^3}} - \frac{k}{m} \qquad m = 2 - k^2 \tag{4.16}$$

Avanzini improved the convergence of the method by converting some expressions into series expansion- However, the core of this solver lies in the strong and robust initial guess procedure devised by the author. Although based on a rational-formulae (arbitrary initial guess), the solver rapidly converges in a couple of iterations to the final solution.

The solution procedure is similar to Avanzini's solver, in the sense that Kepler's equation is evaluated for various values of $k$ till it matches the current time of flight.

The algorithm was also devised for working within the multi-revolution scenario but as introduced early, the analysis of this region is out of the scope of this work. Therefore, the curves for the time of flight for this solver presented in figure 4.5 only show the direct transfer region.

The numerical method used by Arora is Halley's one and the subroutine for the computation of the initial and final velocity vectors is made using the $f$ and $g$ functions.

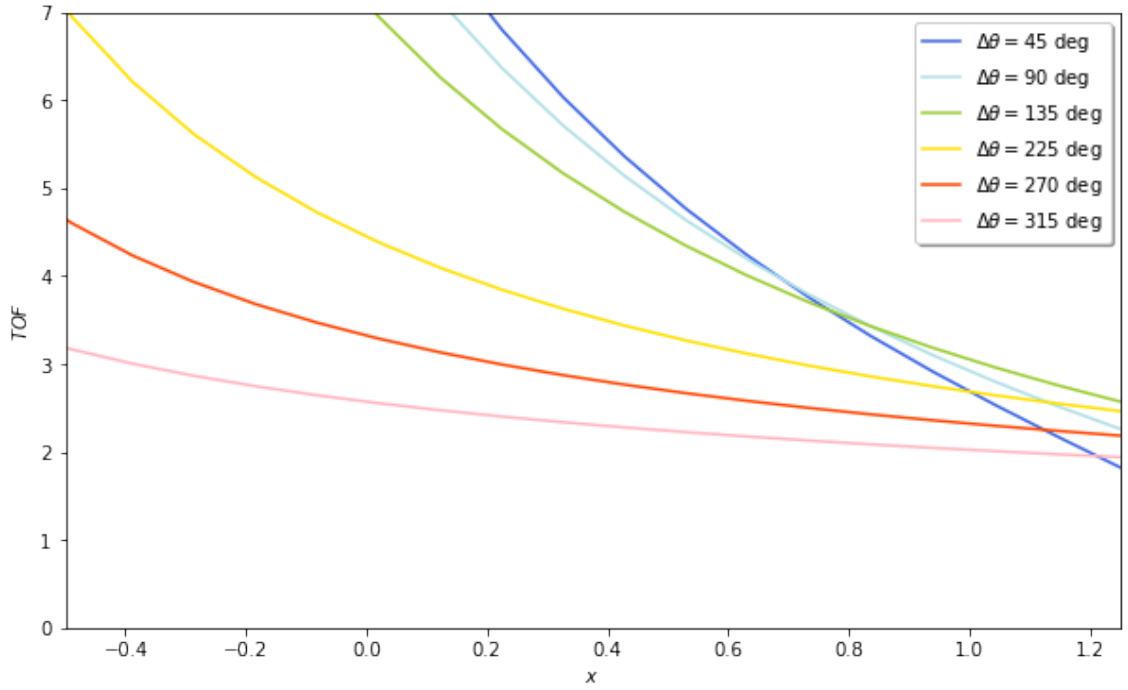


**Fig. 4.5:** Arora 2013 time of flight curves for various transfer angles. This figure was obtained for $\rho = r_2/r_1 = 2$ with unitary $\mu$ and $\Delta t$.

### 4.1.6 Vallado 2013

One of the most popular books about modern astrodynamics is the *Fundamentals of astrodynamics and applications* by which also provides different pseudo-code for implementing a variety of useful computer algorithms related with orbital mechanics. One of these algorithms was an improved version of the Bate, Mueller, White, and Saylor 1971 universal one, for which Vallado imposed a bisection method. By doing so, the method is ensured to converge no matter the type of orbit.

Vallado introduces the $\psi$ variable as the free-parameter. This variable is related to the universal Kepler's equation via equation 4.17:

$$t = \frac{\chi^3 c_3 + A\sqrt{y}}{\sqrt{\mu}} \tag{4.17}$$

where the variables $\chi$ and $y$ are given by:

$$\chi = \sqrt{\frac{y}{c_2}} \qquad y = \|\vec{r_1}\| + \|\vec{r_2}\| + \frac{A(\psi + c_3 - 1)}{\sqrt{c_2}} \tag{4.18}$$

and the transfer angle parameter $A$ is computed as:

$$A = \begin{cases} \sqrt{\|\vec{r_1}\|\|\vec{r_2}\|(1 + \cos(\Delta\theta))} & \text{if } \Delta\theta < \pi \\ -\sqrt{\|\vec{r_1}\|\|\vec{r_2}\|(1 + \cos(\Delta\theta))} & \text{if } \Delta\theta > \pi \end{cases} \tag{4.19}$$

In previous equations, the values for the $c_2$ and $c_3$ variables can be computed using the corresponding Stumpff coefficients evaluated at a particular value of $\psi$.

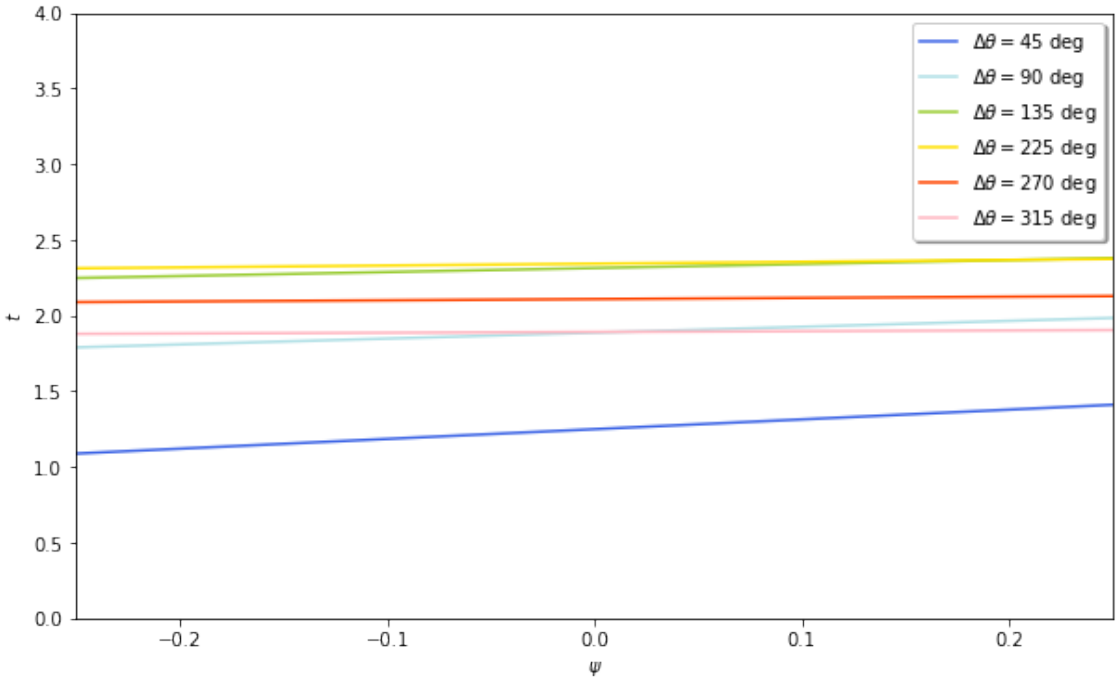


**Fig. 4.6:** Vallado 2013 time of flight curves for various transfer angles. This figure was obtained for $\rho = r_2/r_1 = 2$ with unitary $\mu$ and $\Delta t$.

Similarly to other solvers, the goal is to iterate on the free-parameter till the time computed using the universal Kepler's equation is equal to the expected one imposed by the statement of the problem within some tolerance. Because this method makes use of bisection, the solution is guaranteed to converge always. However, a huge number of iterations is required for the solver to achieve accurate results.

This method is belongs to the same branch as the one for Arora and Russell 2013, see the diagram from subsection 3.3.5. Therefore, Vallado's solver was selected due to its simplicity against Arora's one so a performance can be between them.

### 4.1.7 Izzo 2015

The last of the solvers for this work is the one devised by Izzo 2015. This solver quickly became popular because it enhanced Lancaster and Estes 1970 and Gooding 1990 ones by introducing one last change of variable. Not only that, the Halley's method was upgraded to a Householder's one while the initial guess made use of a linear approximation instead of a bi-linear one.

Izzo's solver belongs to the 3.3.5 and names the free-parameter $\xi$. This variable is related to Gooding's one $x$ via the transformation given in 4.20:

$$\xi = \begin{cases} \log(1+x) & \text{if } M = 0 \\ \log\left(\frac{1+x}{1-x}\right) & \text{if } M > 0 \end{cases} \tag{4.20}$$

Regarding the non-dimensional time of flight, this variable is also modified making use of expression 4.21:

$$\tau = \log(T) \tag{4.21}$$

Izzo had to deal with the computation of the high-order derivatives of $\tau$ with respect to $\xi$. These are not provided here but can be checked in the original report.

All previous modifications, ended up showing the relation between the free-parameter and non-dimensional transfer angle which is presented in figure 4.7.

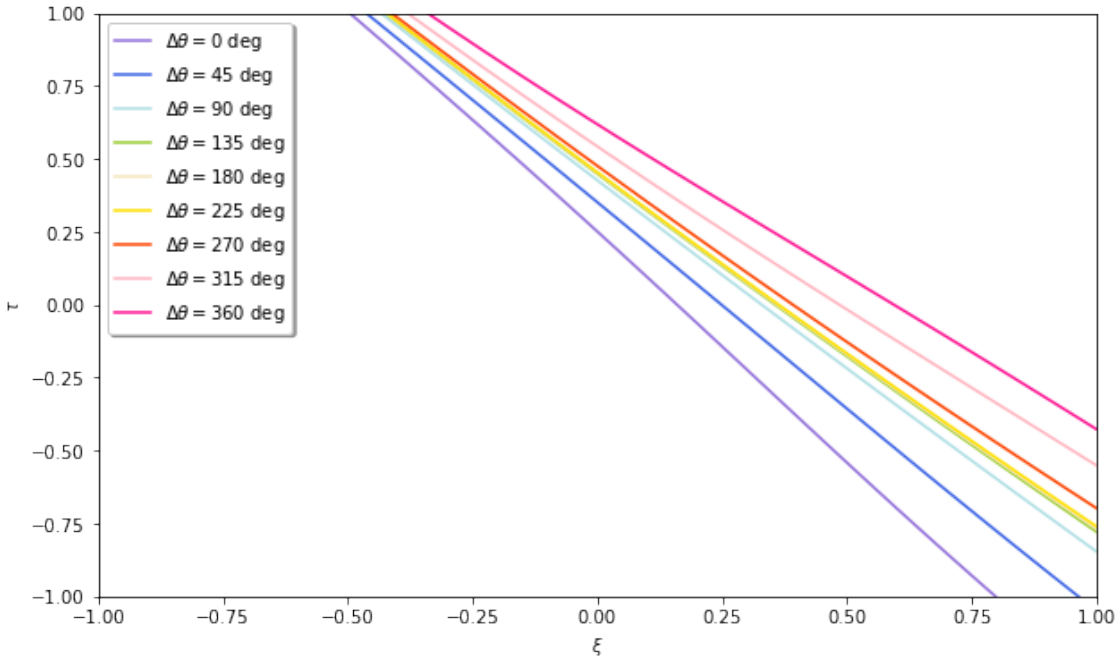


**Fig. 4.7:** Izzo 2015 time of flight curves for various transfer angles. This figure was obtained for $\rho = r_2/r_1 = 2$ with unitary $\mu$ and $\Delta t$.

## 4.2 Comparative table of elements

After introducing all solvers in a more detailed way, it is convenient to collect all the information an present it to reader in a more compact way. This is the goal of table 4.1 provided at the end of this section, in which the different elements and steps used by each solver have been collected.

Although the free-parameters are various, it is possible to see some common elements between the selected modern solvers. For example, those ones using a numerical method make use of high order one to ensure quick convergence or a bounded

one, so the solution is always achieved. Not only that, for the velocity vectors construction, the preferred approach seems to be through the usage of $f$ and $g$ functions. Regarding the initial guess, rational formulae is the most popular among the solvers presented in this work. Other initial guess procedures like linear or bi-linear can also be found.

| **Method** | **Step** | | | |
|---|---|---|---|---|
| | **Free-parameter** | **Initial guess** | **Numerical method** | $\vec{v_1}$ **and** $\vec{v_2}$ |
| Gauss 1809 | $x$ | Rational formulae | System of equations | $f$ and $g$ |
| Battin 1984 | $x$ | Rational formulae | System of equations | $f$ and $g$ |
| Gooding 1990 | $x$ | Bi-linear | Halley's method | $v_{r_i}$ and $v_{t_i}$ |
| Avanzini 2008 | $e_T$ | Interval | Regula-falsi | COE to RV |
| Arora 2013 | $k$ | Rational formulate | Halley's method | $f$ and $g$ |
| Vallado 2013 | $\psi$ | Interval | Bisection | $f$ and $g$ |
| Izzo 2015 | $\xi$ | Linear | Householder's method | $v_{r_i}$ and $v_{t_i}$ |

Table 4.1: Basic elements of each selected solver.

# 5 Results

This chapter is devoted to the presentation of the performance results. Three different metrics were developed to measure the performance of a solver: the number of iterations, the total time per iteration and the total computation time. Contour plots evaluating each of the previous metrics are provided relating values of the non-dimensional transfer angle with the non-dimensional time of flight. A comparison table is provided at the end of the chapter as a resume.

## 5.1 Performance comparison

This section is the core of this work. All the metrics developed and used for the carrying out the performance comparison are explained in detail together with the justification on why they were selected.

Previous performance studies made by Klumpp 1999, Torre Sangrà and Fantino 2015 and in Martínez and Sanjurjo 2021 were used as the main source of information for this section and their analysis was improved.

All algorithms have been implemented under the Python programming language and a library named *lamberthub*[17] created under a public license, so any author can access it. This software package ships with all the necessary tools to evaluate the performance of a solver too and it has been heavily tested again literature cases not only for the direct transfer problem but also for the multiple revolutions one.

The simulations were performed considering a non-dimensional radius of $\rho = \|\vec{r_2}\|/\|\vec{r_1}\| = 2$ and unitary gravitational parameter $\mu = 1$ [L3, T2], where L3 and T2 stand for units of cubic length and squared time respectively. The problem was solved for a variety of transfer angles $\Delta\theta$ between 0 and $2\pi$ and non-dimensional transfer times $\tau$ within the same range of values. For this last parameter, equation 5.1:

$$\tau = \Delta t \cdot \sqrt{\frac{8\mu}{s^3}} \tag{5.1}$$

By making use of previous equation, each revolution solution lies within an interval which is a multiple of $2\pi$. For example, the direct transfer problem will have the

[17] The software can be accessed in https://github.com/jorgepiloto/lamberthub. Latest documentation of the project is hosted in https://lamberthub.readthedocs.io/en/latest/

solution within the $[0, 2\pi)$ interval, while for the multiple-revolution scenario, the interval changes to $[2\pi M, 2\pi(M+1))$.

As it has been introduced in several sections, this work only covers the direct arc transfer, meaning that the multi-revolution scenario is not studied. All orbits were assumed to be *prograde* and no distinction between *low* or *short* paths was required as multiple solutions only appear in the multi-revolution problem. The absolute tolerance was set to $1e-5$ and the relative one to $1e-7$.

The three metrics used in this report are:

- **Number of iterations**: this metric provides information about the convergence of the numerical method but also about the initial guess. Because the initial guess outputs an estimate solution, the accurate this routine is, the lower the amount of iterations required by the numerical method.
- **Time per iteration**: which measures the mean time required for computing a new approximate value of the free-parameter. This metric reveals more information about the time consumed by the numerical method.
- **Total computation time**: which computes the total time required by the algorithm when computing the solution to the problem. When comparing this value to the time employed for the iteration workload, it is possible to identify bottlenecks and weak implementation points.

Finally, the computer used to perform all the different computations hold the following specifications: Thinkpad X230 Intel Core i5-3320M CPU at 2.60G EndevourOS.

### 5.1.1 Number of iterations

The first metric used for the performance comparison is the amount of iterations it takes for a solver to achieve the solution for a given combination of $\Delta\theta$ and $\Delta\tau$. Black regions show for which points the algorithm did not converged, either because of the accuracy or the method itself.

Gauss' solver, in figure 5.1, reveals that the method is only valid for lower transfer angle and non-dimensional times. This situation was already known and claimed by several authors in literature. On the other hand, the improved version of the method by Battin, see 5.2 converges for any case and the imposed absolute tolerance. These last method is seen to be more robust and accurate than Gauss one.

The solver devised by Gooding shows an stable behavior for most of the solution space, only requiring between two and three iterations, 5.3. This is due to the strong initial guess together with the usage of Halley's method.

Regarding Avanzini's solver, this algorithm exhibits a lower number of iterations for transfer angles of the same value of the non-dimensional time of flight. For

corner cases, see 5.4, the iterations increase up to eight or nine. Even with that, the solution is found in all of the cases, meaning that the algorithm converged without any problem.

Arora's algorithm, in figure 5.5, is seen to be the most stable one, requiring a mean of only two iterations. This algorithm holds the most complex initial guess procedure of all the solvers presented in this work, which together with Halley's method explains the lower amount of iterations. Convergence is ensured with this solver.

Vallado's solver also converged to the majority of the cases, except for those ones with low time of flight. However, due to the bisection method employed, the number of iterations increases dramatically. This can be seen in figure 5.6.

Finally, Izzo's solver in figure 5.7 is also seen to be stable and robust for the majority of the cases. Only values around $\tau = \pi/2$ require more than two iterations, no matter the transfer angle.

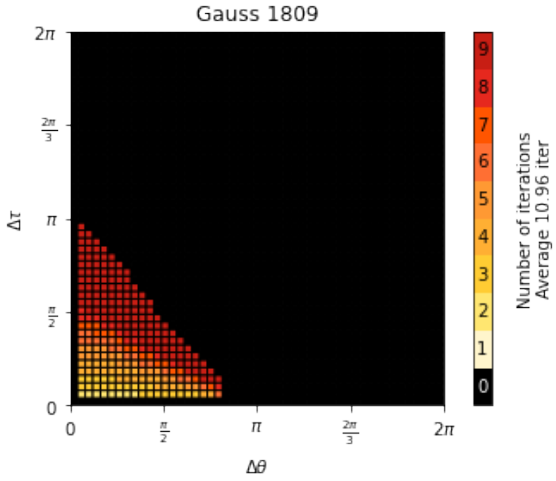


**Fig. 5.1:** Gauss' iterations.

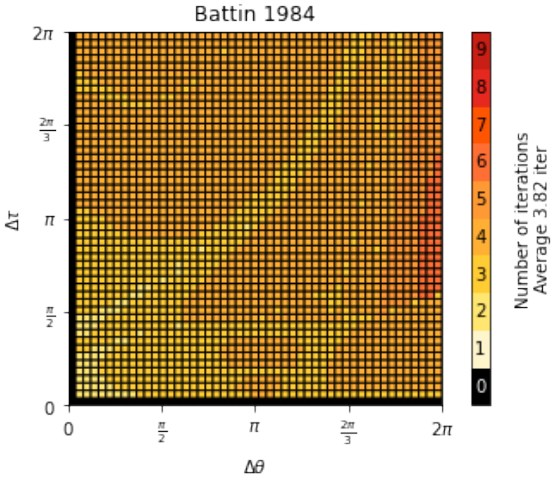


**Fig. 5.2:** Battin's iterations.

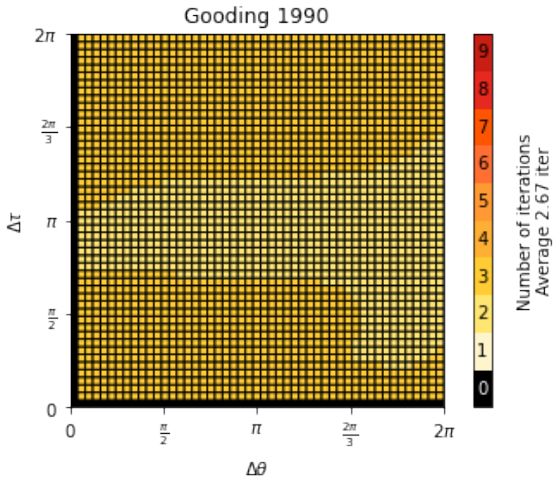


**Fig. 5.3:** Gooding' iterations.

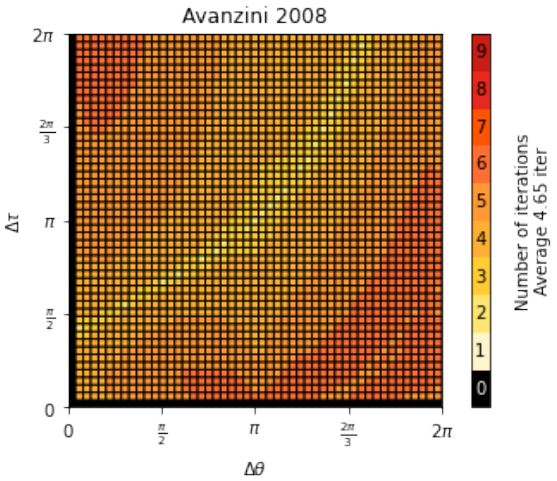


**Fig. 5.4:** Avanzini's iterations.

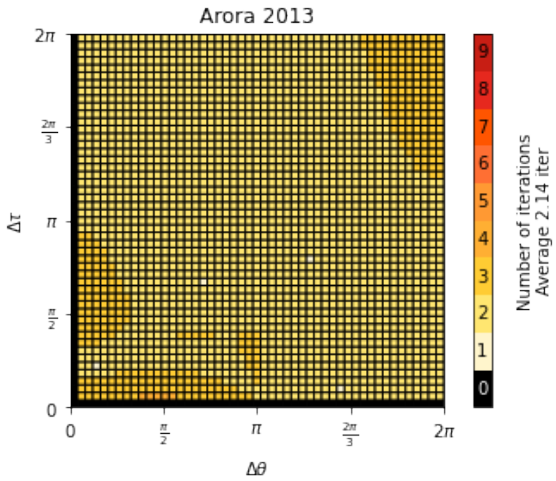


**Fig. 5.5:** Arora' iterations.

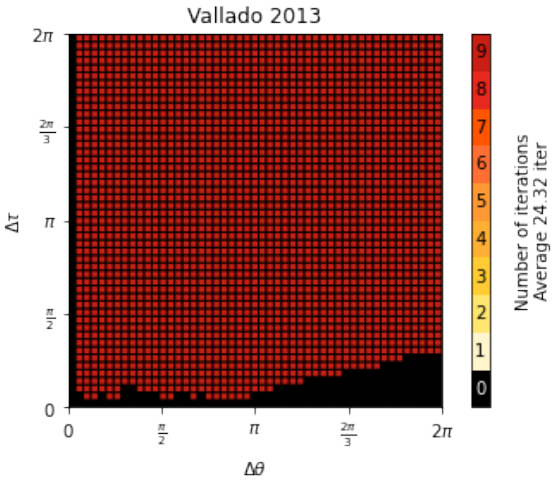


**Fig. 5.6:** Vallado's iterations.

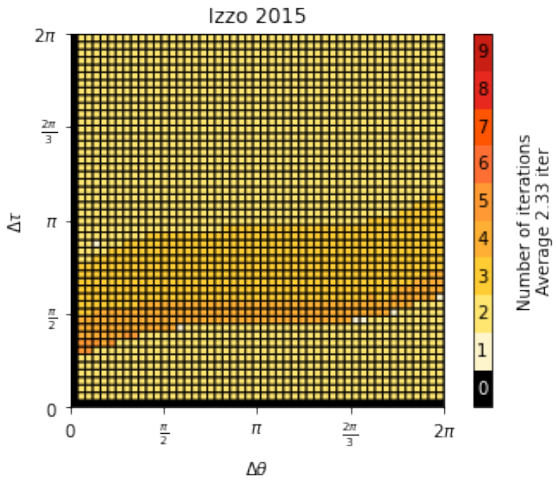


**Fig. 5.7:** Izzo's iterations.

### 5.1.2 Time per iteration

A solver may require a huge number of may require a huge number of iterations. However, if each one of those only needs a minimum amount of time, the overall computation might be a high performance one. In this subsection, the time per iteration of each one of the solvers is presented.

It is important to state that the time required for a computer to solve a particular problem might slightly vary due to internal processes which have nothing to do with the computation process itself. Therefore, the problem was solved multiple times and the mean iteration time. Let us now present the results.

For the case of Gauss' solver, the time consumed per iteration is stable as depicted by figure 5.8. On the other hand, Battin's solver in figure 5.9 shows a reduction in the time per iteration for low values of the canonical transfer time. However, this

last solver requires more time than Gauss one due to the complexity of the method used. Nevertheless, it converges for all cases, which is a major advantage over speed as a solution is found.

Gooding's solver in 5.10 only shows an critical region near non-dimensional times of $\tau = \pi/2$. For the rest of the cases, the time per iteration is quite low and stable. Halley's method is proof to be fast when carrying out the iteration workload.

The algorithm requiring the greatest amount of time per iteration is Avanzini's one, see figure 5.11. This solver requires more time for the regions in which it employed less iterations.

Arora's solver, on the other hand, shows an overall time per iteration slightly greater than Gooding's one. Even with that, the solver only experiences an increase in the computation of the time per iteration for values below of $\tau = \pi/2$. The diagram for this solver is shown in figure 5.12.

Vallado's solver, in figure 5.13 shows now a closer behavior to the rest of the solvers when discussing the time per iteration. This is due to the bisection method employed by the solver, as this root solver is fast from the time per iteration point of view but requires a huge amount of iterations for achieving the desired absolute tolerance.

The last of the solvers, Izzo's one in figure 5.14 is not as stable as Gooding's or Arora's algorithms, but also shows a great stability and performance for the iteration workload. As similar to Arora's, this solver experiences and increase in the time per iteration for values of $\tau = \pi/2$.

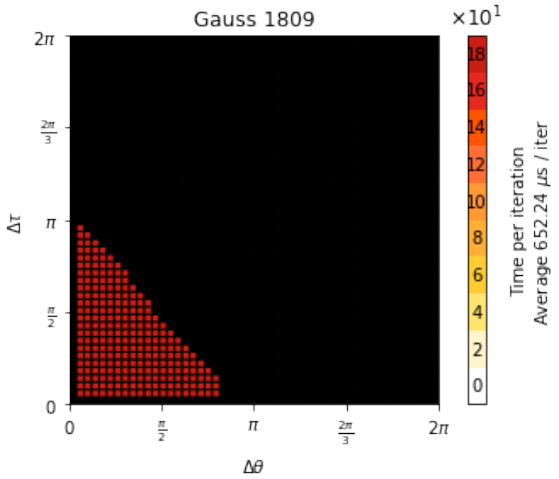


**Fig. 5.8:** Gauss' time per iteration.

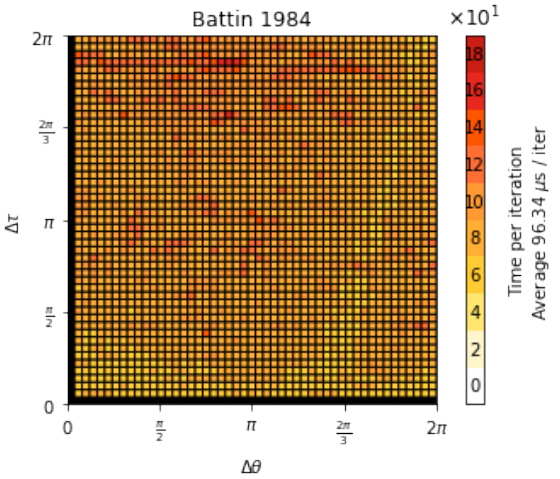


**Fig. 5.9:** Battin's time per iteration.

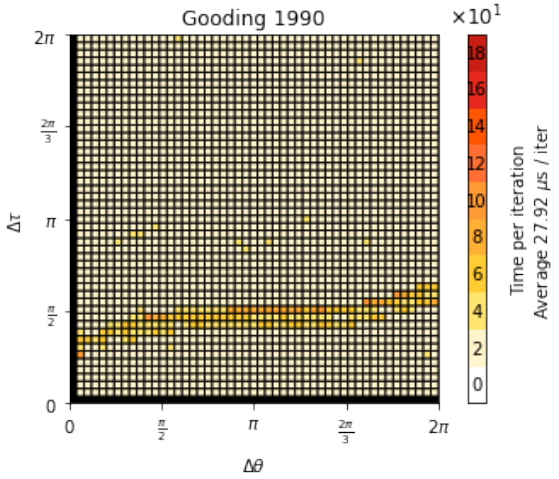


**Fig. 5.10:** Gooding' time per iteration.

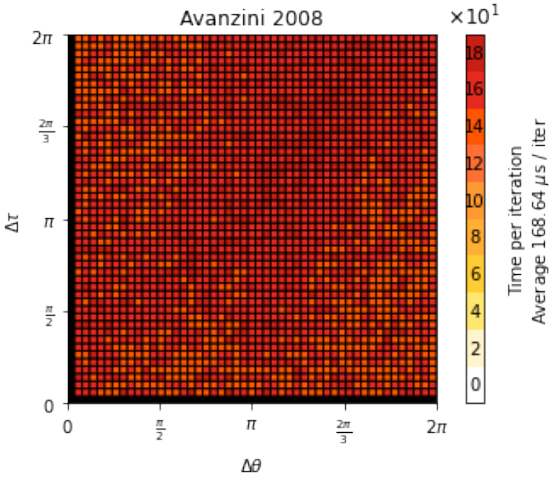


**Fig. 5.11:** Avanzini's time per iteration.

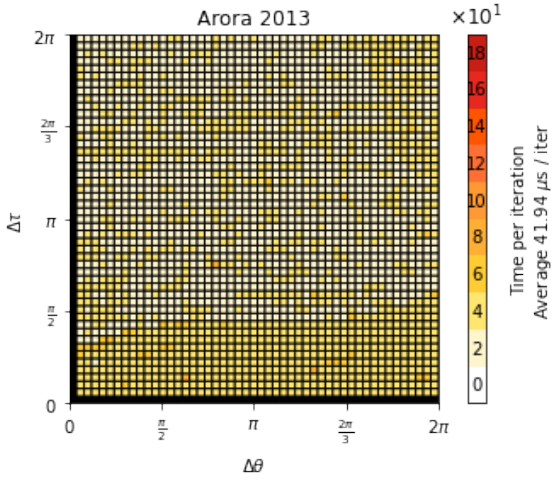


**Fig. 5.12:** Arora' time per iteration.

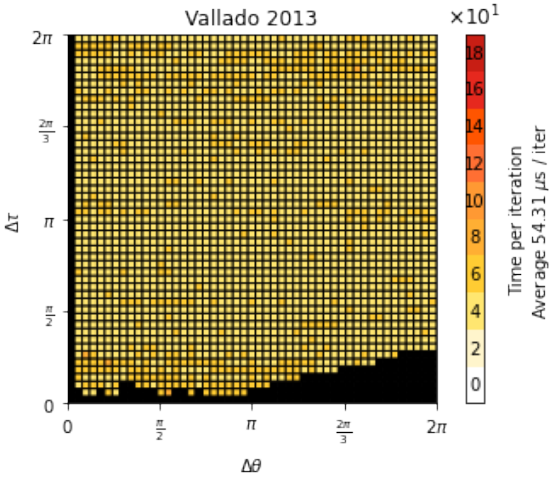


**Fig. 5.13:** Vallado's time per iteration.

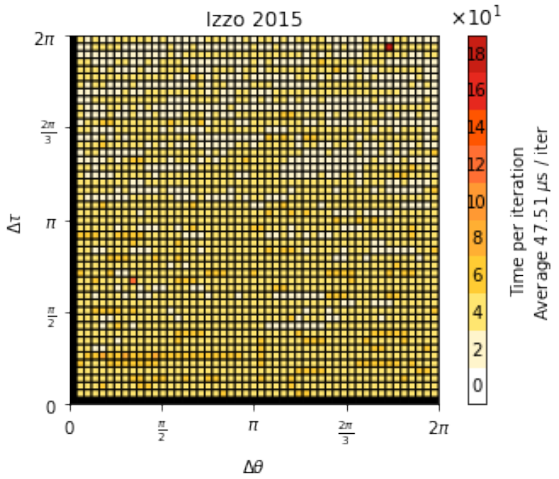


**Fig. 5.14:** Izzo's time per iteration.

### 5.1.3 Total computation time

Finally, the total computation time employed by the algorithm for computing the whole problem is presented. Because the amount of iterations required and the time per iterations are known, it is possible to compute the percentage of the time spent during the iteration workload. The idea is that the main workload of a solver should be located within the iteration process itself and not in the initial guess computation, the construction of the velocity vectors or function calls. Again, it must be pointed out that background processes running on local machine can affect the computation time.

Gauss' solver exhibits a high total computation time, see figure 5.15 even if its time per iteration is low. This means that the algorithm requires lots of iterations to reach an accurate solution. The improved version devised by Battin performs much better, as depicted in figure 5.16. However, these two algorithms are far from being fast when computing the solution.

Gooding's solver performs much better in total time than Gauss or Battin algorithms. The figure in 5.17 is proportional to previous results for the number of iterations and time per each one of those. In addition, a bottlenecks is identified for transfer times near $\tau = \pi/2$.

On the other hand, Avanzini's solver has a huge time cost as seen in figure 5.18. When comparing this result with previous figures in 5.4 and 5.11, an implementation issue might be the reason behind this. Because the Kepler's equation becomes complex when written as function of the transverse eccentricity, a simplification making use of function calls was made. The excessive calls to these function are probably the reason behind this total computation time results. The usage of a high-order method might be also a good strategy for reducing the computation time.

Arora's solver presents a very low computation time when compared to the rest of the solvers. Not only that, a uniform value is found for every combination of the transfer angle and the canonical time of flight, being not possible to identify any bottlenecks on figure 5.19.

Vallado's algorithm, due to bisection method again, is seen to be detrimental for this algorithm, see the results in figure 5.20. However, as pointed by its author, it converges to the majority of the cases. A possible way to improve this algorithm would be the introduction of a better initial guess or mixed numerical solver.

Finally, Izzo's algorithm is the one requiring the lowest amount of time to converge to the solution. The corner case for this solver was previously identified during the number of iterations but no effect is seen in figure 5.21.

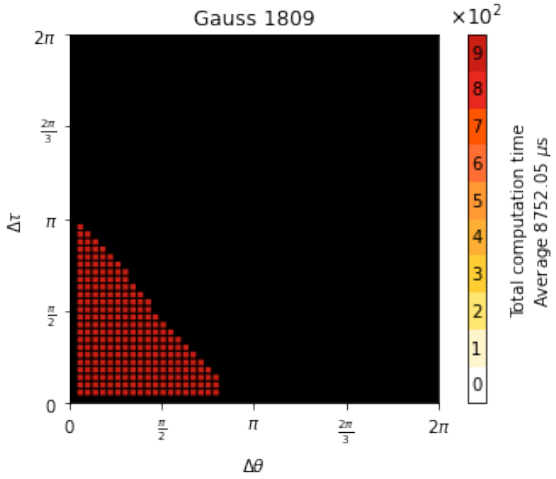


**Fig. 5.15:** Gauss' total computation time.

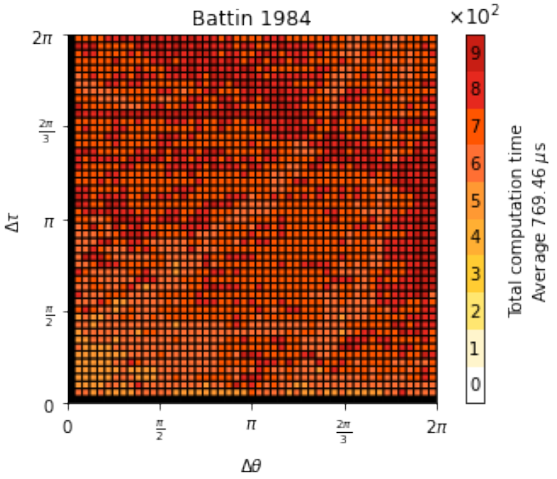


**Fig. 5.16:** Battin's total computation time.

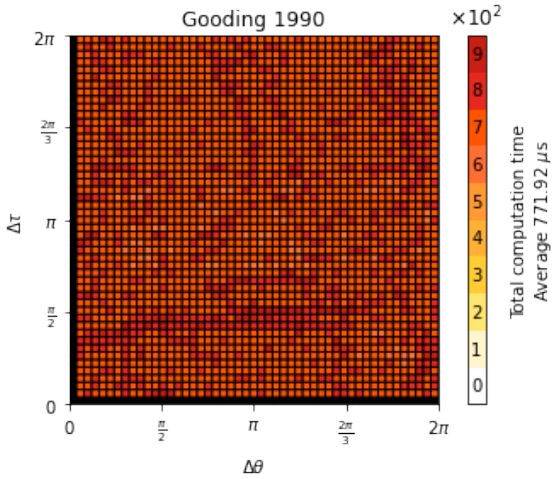


**Fig. 5.17:** Gooding' total computation time.

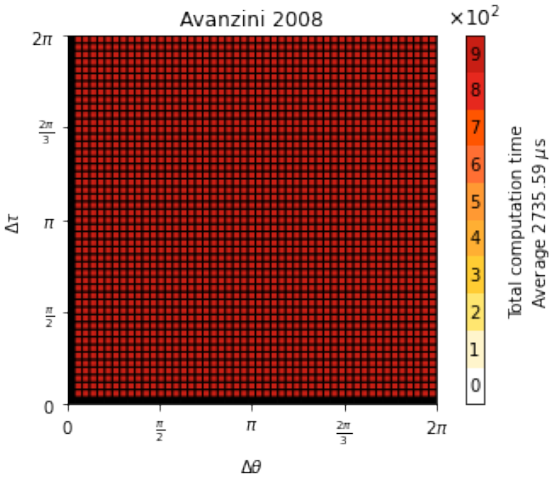


**Fig. 5.18:** Avanzini's total computation time.

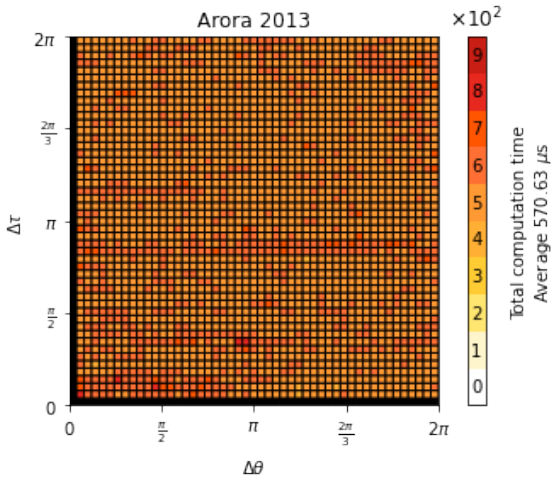


**Fig. 5.19:** Arora' total computation time.

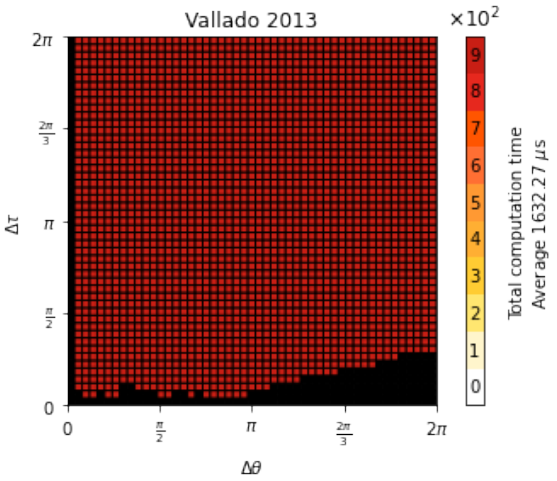


**Fig. 5.20:** Vallado's total computation time.

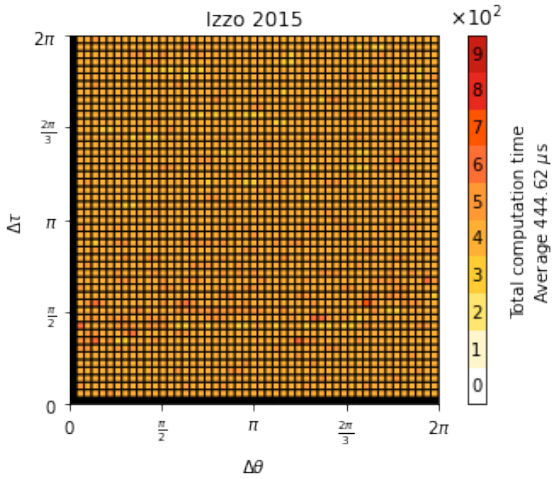


**Fig. 5.21:** Izzo's total computation time.

## 5.2 Discussion of the results

In table 5.1, the fundamental performance data has been collected so reader can better study it. All the metrics are shown, including the mean number of iterations, the time per each one of those, the total computation time and finally the iteration workload. This last parameter has been computed considering expression 5.2:

$$\text{Iteration workload} = \frac{\text{Mean number of iterations} \times \text{Time per iteration}}{\text{Total computation time}} \tag{5.2}$$

being the result expressed as a percentage.

| Method | Performance data | | | |
|---|---|---|---|---|
| | **Mean iterations** | **Time per iteration** | **Total computation time** | **Iteration workload** |
| Gauss 1809 | 10.96 | 652.24 $\mu$s/iter | 8752.05 $\mu$s | 81.68 % |
| Battin 1984 | 3.82 | 96.34 $\mu$s/iter | 769.49 $\mu$s | 47.83 % |
| Gooding 1990 | 2.67 | 27.92 $\mu$s/iter | 771.92 $\mu$s | 9.66 % |
| Avanzini 2008 | 4.65 | 168.64 $\mu$s/iter | 2735.59 $\mu$s | 28.67 % |
| Arora 2013 | 2.14 | 41.94 $\mu$s/iter | 570.63 $\mu$s | 15.73 % |
| Vallado 2013 | 24.32 | 54.31 $\mu$s/iter | 1632.27 $\mu$s | 80.92 % |
| Izzo 2015 | 2.33 | 47.51 $\mu$s/iter | 444.62 $\mu$s | 24.90 % |

Table 5.1: Critical performance data for each solver.

The iteration workload indicates how much of the total computation time is consumed by a solver when performing the iterative process. One might think that a high iteration workload is desired, which indicates that the root solver performs the majority of the work. However, this may also mean that the initial guess is poor, so an excessive amount of iterations are being performed.

For the case of Gauss 1809, this algorithm shows the greatest iteration workload meaning that its focuses on the computation of the independent variable. However, this algorithm is known not to converge for the majority of the cases and shows an excessive amount of total computation time.

Regarding Battin and Vaughan 1984, around half of the time is devoted to the computation of the free-parameter. The solver also has a lower computation time if compared to Gauss one. Therefore, the method devised by Battin and Vaughan truly improves the one by Gauss.

Gooding 1990 solver has a similar total computation time as Battin's algorithm. However, the iteration workload is seen to be the lowest one, showing up that the majority of the time is being consumed by the initial guess. Nevertheless, Gooding's solver converges with great accuracy to the final solution.

On the other hand, the relation between iteration workload and computation time for Avanzini 2008 is poor. This algorithm lacks of an extensive initial guess but requires lots of function calls to have the Kepler's equation as direct function of the free-parameter. These function calls are shown to produce a bottleneck in the performance of this solver. The use of a high-order numerical method would probably reduce the amount of required iterations.

Arora and Russell 2013 shows the lowest computation time per iteration and the iteration workload shows a low value due to the short amount of iterations required by the solver. Arora's solver employed an exhaustive initial guess, devoting the majority of the time to this procedure. Neverhteless, the solver is seen to have a high performance, knowing that it converges for all cases and includes the multi-revolution scenario.

Izzo 2015 has the lowest computation time while showing a relatively high itera-

tion workload. The arbitrary initial guess based on a linear approximation in combination with Householder's method makes this a high performance solver who's implementation improves the one devised by Gooding's one.

Finally, for sorting of the solvers from a performance point of view, the following criteria has been established by order of priority:

1. Convergence of a solver: the greater of the covered cases, the better.

2. Total computation time: the lower, the better.

3. Iteration workload: the greater, the better.

which leads to the following performance ranking: Izzo 2015, Arora and Russell 2013, Gooding 1990, Battin and Vaughan 1984, Avanzini 2008, Vallado 2013, Gauss 1809.

A key concept arose from the performance comparison. A strong initial guess is critical for reducing the number of iterations but if too complex, the total computation time will increase. The idea is to have a simple initial guess routine so the whole iteration workload can be performed by the root solver. From the results presented in this work, high-order root solvers are preferred, although those ones based on Bolzano's theorem might be used for critical regions in the time of flight curves.

# 6 Conclusion

In section 5.1, the performance comparison for the selected solvers was presented, achieving the main goal of this work. However, different performance tools were developed for previous purpose and will be useful for other comparison works. In this last chapter, some ideas about future work and extension of this one are exposed together with a final conclusion.

## 6.1 Future work

The results presented in this work might be enhanced and expanded in a future by considering the addition of more Lambert's problem solvers and metrics. These would allow to carry out more complex performance analysis. Some ideas are listed down:

- **Error color palette.** Define a set of colors associated to particular errors such us exceeding maximum number of iterations or wrong algorithm input. Those would allow to properly identify which problem was found in the black regions of the non-dimensional time versus the transfer angle contour plots.
- **Multi-revolution analysis.** Expand the analysis to the multi-revolution scenario, as this work only considered the direct-arc transfer problem.
- **Implement more solvers.** Implement more Lambert's problem solvers under the same library and programming language so the performance comparison is fair. The *lamberthub* provides all the required tools for carrying out this task.
- **Memory usage analysis.** A comparison using the metric of memory usage might be also interesting to establish the minimum requirements for a CPU to make use a solver.
- **Parallelization analysis.** This analysis would be devoted to the investigation of how suitable is for a solver to be implemented using high performance computing techniques based on GPU parallelization.

## 6.2 Final conclusion

This work presented a performance comparison between modern Lambert's problem solvers, considering the direct-arc transfer scenario and not the multi-revolution one.

A review of Lambert's problem was presented in chapter 2. The problem was presented from both a geometrical and analytical point of view using as example the first solution to the problem devised by Lagrange.

Chapter 3 was devoted to modern solvers. The different steps involved during the computation of the problem's solution were presented together with a solvers classification based on the free-parameter. Among the selected solvers, Gauss 1809, Battin and Vaughan 1984, Gooding 1990, Avanzini 2008, Arora and Russell 2013, Vallado 2013 and Izzo 2015 were the chosen ones. Only the direct-arc transfer and not the multi-revolution scenario was considered in this analysis.

Then, in chapter 4, different metrics such us the required number of iterations, time per iteration and total computation time for a particular combination of non-dimensional time and transfer angle were used to test the accuracy and robustness of each solver. Contour plots were generated showing these results. Finally, the performance was based considering first the convergence of a solver, the total computation time and the iteration workload. With all these conditions, previous solvers were sorted in performance as Izzo 2015, Arora and Russell 2013, Gooding 1990, Battin and Vaughan 1984, Avanzini 2008, Vallado 2013, Gauss 1809.

# Bibliography


[1] J. H. Lambert, *Insigniores orbitae cometarum proprietates*. 1761.

[2] J. L. Lagrange, *Méchanique analitique*. Vve Desaint, 1788.

[3] C. F. Gauss, *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*. 1809.

[4] S. Herrick and A. Liu, "Two body orbit determination from two positions and time of flight," *Appendix A, Aeronutronic*, vol. 365, 1959.

[5] A. D. Wheelon, "Free flight of a ballistic missile," *ARS journal*, vol. 29, no. 12, pp. 915–926, 1959.

[6] R. Gunkel, D. Lascody, and D. Merrilees, "Impulsive midcourse correction of an interplanetary transfer," in *Xth International Astronautical Congress London 1959/X. Internationaler Astronautischer Kongress/Xe Congrès International d'Astronautique*, Springer, 1960, pp. 650–670.

[7] P. Escobal, "Methods of orbit determination.," *Methods of orbit determination*, 1965.

[8] W. Hilton, "I. theory of flight of artificial earth satellites. pe el'yasberg,-ii. motion of an artificial satellite about its center of mass. vv beletskii. israel program for scientific translations, translated by z. lerman from the russian.," *The Aeronautical Journal*, vol. 72, no. 690, pp. 529–529, 1968.

[9] E. Lancaster and R. Estes, "A universal solution of lambert's problem," *NASA TM X-65306*, 1970.

[10] F. R. Moulton, *An introduction to celestial mechanics*. Courier Corporation, 1970.

[11] R. R. Bate, D. D. Mueller, J. E. White, and W. W. Saylor, *Fundamentals of astrodynamics*. Courier Dover Publications, 1971.

[12] R. T. Savely, B. F. Cockrell, and S. Pines, *Apollo Experience Report: Onboard Navigational and Alignment Software*. National Aeronautics and Space Administration, 1972.

[13] C. Simó, "Solución al problema de lambert mediante regularización," *Collectanea Mathematica, 1973, vol. 24, núm. 3, p. 231-248*, 1973.

[14] D. J. Jezewski, "K/s two-point-boundary-value problems," *Celestial mechanics*, vol. 14, no. 1, pp. 105–111, 1976.

[15] J. Kriz, "A uniform solution of the lambert problem," *Celestial mechanics*, vol. 14, no. 4, pp. 509–513, 1976.

[16] F.-T. Sun, "A global analysis of two-terminal trajectories," *Acta Astronautica*, vol. 4, no. 5-6, pp. 469–493, 1977.

[17] J. E. Prussing, "Geometrical interpretation of the angles cy and b in lambert's problem," *Journal of Guidance and Control*, vol. 2, no. 5, pp. 442–443, 1979.

[18] R. M. Vaughan, "An improvement of gauss' method for solving lambert's problem," Ph.D. dissertation, Massachusetts Institute of Technology, 1983.

[19] R. H. Battin and R. M. Vaughan, "An elegant lambert algorithm," *Journal of Guidance, Control, and Dynamics*, vol. 7, no. 6, pp. 662–670, 1984.

[20] R. Gooding, "A procedure for the solution of lambert's orbital boundary-value problem," *Celestial Mechanics and Dynamical Astronomy*, vol. 48, no. 2, pp. 145–165, 1990.

[21] S. L. Nelson and P. Zarchan, "Alternative approach to the solution of lambert's problem," *Journal of Guidance, Control, and Dynamics*, vol. 15, no. 4, pp. 1003–1009, 1992.

[22] J. Thorne, "Series reversion/inversion of lambert's time function," in *Astrodynamics Conference*, 1995, p. 2886.

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

[24] A. Klumpp, "Performance comparison of lambert and kepler algorithms," *Interoffice Memorandum, JPL*, 1999.

[25] D. Teets and K. Whitehead, "The discovery of ceres: How gauss became famous," *Mathematics Magazine*, vol. 72, no. 2, pp. 83–93, 1999.

[26] J. E. Prussing, "A class of optimal two-impulse rendezvous using multiple-revolution lambert solutions," *The Journal of the Astronautical Sciences*, vol. 48, no. 2, pp. 131–148, 2000.

[27] A. Celletti, "The levi-civita, ks and radial-inversion regularizing transformations," in *Singularities in Gravitational Systems: Applications to Chaotic Transport in the Solar System*, D. Benest and C. Froeschlé, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 25–48, ISBN: 978-3-540-48009-9. DOI: 10.1007/3-540-48009-9_2. [Online]. Available: https://doi.org/10.1007/3-540-48009-9_2.

[28] H. Shen and P. Tsiotras, "Using battin's method to obtain multiple-revolution lambert's solutions," *Advances in the Astronautical Sciences*, vol. 116, pp. 1067–1084, 2003.

[29] J. D. Thorne, "Lambert's theorem—a complete series solution," *The Journal of the Astronautical Sciences*, vol. 52, no. 4, pp. 441–454, 2004.

[30] G. Avanzini, "A simple lambert algorithm," *Journal of guidance, control, and dynamics*, vol. 31, no. 6, pp. 1587–1594, 2008.

[31] C. Ericson, *Floating-point tolerances revisited*, Oct. 2008. [Online]. Available: https://realtimecollisiondetection.net/blog/?p=89.

[32] Q. He, J. Li, and C. Han, "Multiple-revolution solutions of the transverse-eccentricity-based lambert problem," *Journal of guidance, control, and dynamics*, vol. 33, no. 1, pp. 265–269, 2010.

[33] W. E. Wiesel, *Modern Astrodynamics*. Aphelion Press Beavercreek, OH, 2010, vol. 2652.

[34] P. Arlulkar and S. Naik, "Solution based on dynamical approach for multiple-revolution lambert problem," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 3, pp. 920–923, 2011.

[35] G. J. Der, "The superior lambert algorithm," *AMOS, Maui, Hawaii*, 2011.

[36] J. Ahn and S.-I. Lee, "Lambert algorithm using analytic gradients," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 6, pp. 1751–1761, 2013.

[37] N. Arora and R. P. Russell, "A fast and robust multiple revolution lambert algorithm using a cosine transformation," *Paper AAS*, vol. 13, p. 728, 2013.

[38] D. A. Vallado, *Fundamentals of astrodynamics and applications*. Springer Science & Business Media, 2013, vol. 12.

[39] S. E. Wailliez, "On lambert's problem and the elliptic time of flight equation: A simple semi-analytical inversion method," *Advances in Space Research*, vol. 53, no. 5, pp. 890–898, 2014.

[40] C. Wen, Y. Zhao, and P. Shi, "Derivative analysis and algorithm modification of transverse-eccentricity-based lambert problem," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 4, pp. 1195–1201, 2014.

[41] R. Allen and R. Wenzel, "Tan root for battin's lambert recipe," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 9, pp. 1826–1827, 2015.

[42] D. Izzo, "Revisiting lambert's problem," *Celestial Mechanics and Dynamical Astronomy*, vol. 121, no. 1, pp. 1–15, 2015.

[43] J. D. Thorne, "Convergence behavior of series solutions of the lambert problem," *Journal of Guidance, Control, and Dynamics*, vol. 38, no. 9, pp. 1821–1826, 2015.

[44] D. d. l. Torre Sangrà and E. Fantino, "Review of lambert's problem," in *ISSFD 2015: 25th International Symposium on Space Flight Dynamics, 19-23 October, Munich, Germany*, 2015, pp. 1–15.

[45] R. Jiang, T. Chao, S. Wang, and M. Yang, "Improved semi-major axis iterated method for lambert's problem," in *2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC)*, IEEE, 2016, pp. 1423–1428.

[46] D. De La Torre, R. Flores, and E. Fantino, "On the solution of lambert's problem by regularization," *Acta Astronautica*, vol. 153, pp. 26–38, 2018.

[47] A. Albouy, "Lambert's theorem: Geometry or dynamics?" *Celestial Mechanics and Dynamical Astronomy*, vol. 131, no. 9, Aug. 2019. DOI: 10.1007/s10569-019-9916-2. [Online]. Available: https://doi.org/10.1007/s10569-019-9916-2.

[48] H. Curtis, *Orbital Mechanics for Engineering Students: Revised Reprint*. Butterworth-Heinemann, 2020.

[49] D. De la Torre, "Optimization of interplanetary trajectories with gravity assist," Ph.D. dissertation, 2020.

[50] *Perseverance*, https://mars.nasa.gov/mars2020/mission/overview/, (Accessed on 08/31/2021), 2020.

[51] *Airbus - new space*, https://www.airbus.com/public-affairs/brussels/our-topics/space/new-space.html, (Accessed on 09/03/2021), 2021.

[52] D. Beďatš, "Gauss' calculation of ceres' orbit," 2021.

[53] B. Dawson, *Comparing floating point numbers, 2012 edition*, Jan. 2021. [Online]. Available: https://randomascii.wordpress.com/2012/02/25/comparing-floating-point-numbers-2012-edition/.

[54] *Hope*, https://en.wikipedia.org/wiki/Emirates_Mars_Mission, (Accessed on 08/31/2021), 2021.

[55] J. Martínez, *Lamberthub*, version 0.1, Jun. 2021. DOI: 10.5281/zenodo.4971895. [Online]. Available: https://doi.org/10.5281/zenodo.4971895.

[56] Martínez and Sanjurjo, "A critical review on the state-of-the-art or lambert's problem solvers," *International Conference on Astrodynamics and Techniques*, 2021.

[57] *Tianwen-1*, https://en.wikipedia.org/wiki/Tianwen-1, (Accessed on 08/31/2021), 2021.

[58] M. Alhulayil, A. B. Younes, and J. D. Turner, "Fast p-iteration for lambert's problem,"