Master´s Thesis
Master in Aeronautical Engineering

AUTO-based numerical study of three body problem orbits and trajectories with applications to the Lunar Deep Space Gateway

Author: José Manuel Montilla García

Tutor: Rafael Vázquez Valenzuela

Aerospace Engineering and Fluid Mechanics Department
Higher Technical School of Engineering
University of Seville

Seville, 2018

Master´s Thesis
Master in Aeronautical Engineering

AUTO-based numerical study of three body problem orbits and trajectories with applications to the Lunar Deep Space Gateway

Autor:
José Manuel Montilla García

Tutor:
Rafael Vázquez Valenzuela
Director of the Department of Aerospace Engineering

Aerospace Engineering and Fluid Mechanics Department
Higher Technical School of Engineering
University of Seville

Sevilla, 2018

Master´s Thesis: AUTO-based numerical study of three body problem orbits and trajectories with applications to the Lunar Deep Space Gateway

Autor: José Manuel Montilla García
Tutor: Rafael Vázquez Valenzuela

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Abstract

This work can serve the reader as an initiation in numerical continuation through the AUTO software, which is specialized in bifurcation analysis and has proved to be extremely versatile and efficient in calculating ordinary differential system solutions that would otherwise be much more difficult to obtain. A brief review is made about the concepts of non-linear dynamics that rest on the basis of the problem under study, the circular restricted three body problem. Then, the bases of the numerical continuation are presented, as well as the concrete strategies that will be used for the calculation of periodic solutions and their manifolds. After this, the reader finds a guide to the basic concepts that would allow him to use the AUTO software, as well as to understand the subsequent study that is carried out with it. Finally, AUTO is used to generate the families of periodic orbits that arise from the Lagrange points in the circular restricted three body problem, in addition to the associated stable and unstable manifolds when possible. This analysis is used to comment on the characteristics of the orbits considered to be the best candidates for the *Deep Space Gateway* project, which aims to use near rectilinear Halo orbits.

Resumen

Este trabajo puede servir al lector como iniciación en continuación numérica a través del software *AUTO*, especializado en análisis de bifurcaciones y que demuestra ser extremadamente versátil y eficiente en el cálculo de soluciones a sistemas diferenciales ordinarios que de otra forma serían mucho más difíciles de obtener. Se hace un breve repaso por los conceptos de dinámica no lineal que descansan en la base del tema principal bajo estudio, el problema circular restringido de los tres cuerpos. Después se plantean las bases de la continuación numérica, así como las estrategias concretas que se usarán para el cálculo de soluciones periódicas y sus variedades. Tras esto, el lector se encuentra con una guía de los conceptos básicos que le permitirían utilizar el software *AUTO*, así como comprender el posterior estudio que se realiza con este. Finalmente, se utiliza *AUTO* para calcular las familias de órbitas periódicas que nacen de los puntos de Lagrange en el problema circular restringido de los tres cuerpos, además de las variedades estables e inestables asociadas cuando procedan. Este análisis se aprovecha para comentar las características de las órbitas consideradas como mejores candidatas para el proyecto *Deep Space Gateway* (Portal de espacio profundo), que pretende utilizar órbitas Halo casi rectilíneas.

Índice Abreviado

Contents

Notation

$\mathbb{R}$	Real numbers set
$\mathbb{C}$	Complex number set
$\lvert\mathbf{v}\rvert$	Vector norm **v**
$\langle\mathbf{v},\mathbf{w}\rangle$	Scalar product between **v** and **w**
IFT	Implicit function theorem
CR3BP	Circular restricted three body problem
DSG	Deep space gateway
ODEs	Ordinary differential equations
ESA	European space agency
JAXA	Japan aerospace exploration agency
CSA	Canadian space agency
NRHOs	Near rectilinear halo orbits
$\mathscr{B}$	Banach space
HP	Hopf bifurcation
UZ	User requested output
$PAR(i)$	Parameter *i* inside the AUTO code

1 Introduction

It is trhough science that we prove, but through intuition that we discover.

Henri Poincaré

The three-body problem is a classical problem in physics and mechanics that, in contrast to the Keplerian two-body problem, doesn't have an analytical solution in general. The gravitational problem dates from 1687, when *Newton* published his "*Principia*" and set out the problem of the movements of three masses subjected to the influence of each other, he also tried to apply his results to study the motion of the moon under the influence of the Earth and the Sun. The longitude problem was of great importance in the 1720s and an accurate solution to the three body problem would have been an alternative answer to the marine chronometer. In 1747 *Jean le Rond d'Alembert* and *Alexis Clairaut* both submitted their analyses to the *Académie Royale des Sciences*, in which they used differential equations to be solved by successive approximations. The name "three-body problem" (*Problème des Trois Corps*) began to be used in the 1740s Paris due to this research. It was *Lagrange* in 1772 who demonstrated analytical solutions do exist if certain restrictions are imposed, as discused hereinafter. Finally, in 1887, *Heinrich Bruns* and *Henri Poincaré* showed that no analytical solutions given by algebraic expressions and integrals can exist in general, and even though the words *"chaotic behaviour"* weren't used in this context until the sixties, in [1] *Poincaré* does describe exactly this when considering an unstable solution in a two degrees of freedom Hamiltonian system:

> *If on tries to represent the figure formed by these two curves with an infinite number of intersections whereas each one corresponds with a double asymptotic solution, these intersections are forming a kind of lattice-work, a tissue, a network of infinite closely packed meshes. Each of the two curves must not cut itself but it must fold onto itself in a very complex way to be able to cut an infinite number of times through each mesh of the network.*
> *One will be struck by the complexity of this picture that I do not even dare to sketch. Nothing is more appropriate to give us an idea of the intricateness of the three-body problem and in general all problems of dynamics where one has not a uniform integral and where the Bohlin series are divergent.* (Henri Poincaré)

For the general case the system is 18th-order, and 10 analytical integrals can be written, corresponding to conservation of momentum (three integrals), energy (one integral), and motion of the system's mass center (six integrals). Together, this integrals and special case conditions allow for some solutions to be analytically studied (Lagrange's three-body Solutions [2]). In one of those solutions the masses lie at the vertices of a rotating equilateral triangle, with varying angular velocity and size (figure (1.1)). Another case is that of the three masses collinear with each other at any moment (figure (1.2)). Sub-solutions of the previous types can be found if the motion of the bodies is restricted to be circular, instead of a general conic (figures (1.3) and (1.4)).

Others special restricted forms of the problem have been studied, but so far the previous cases shown are the only ones with analytical solution. Here, the sub-problem to be addressed is that of one of the masses to be negligible for the motion of the other two, the restricted three-body problem. In particular, the case of study impose the two massive bodies to be in circular motion around their center of mass, and so it is called *Circular*

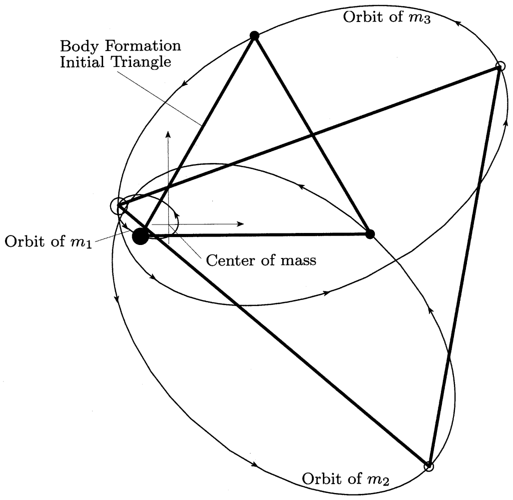


Figure 1.1 General equilateral triangle solution. From [2].

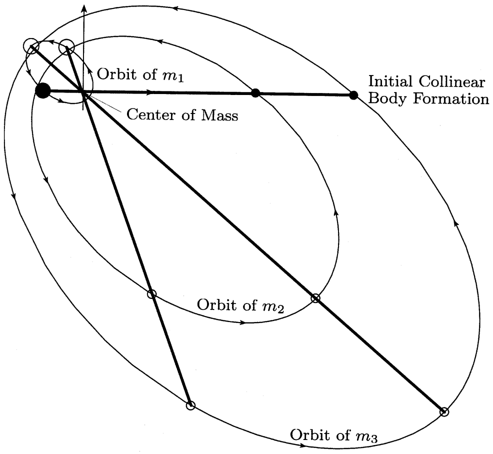


Figure 1.2 General invariant collinear solution. From [2].

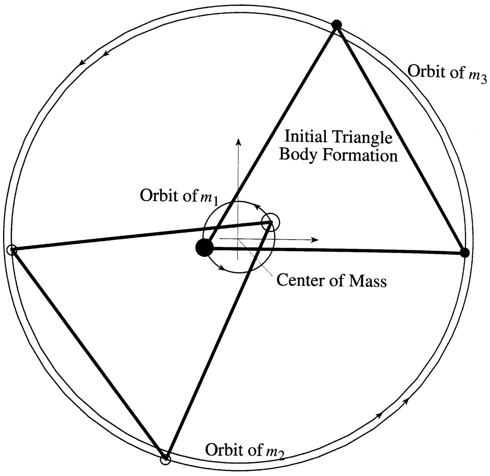
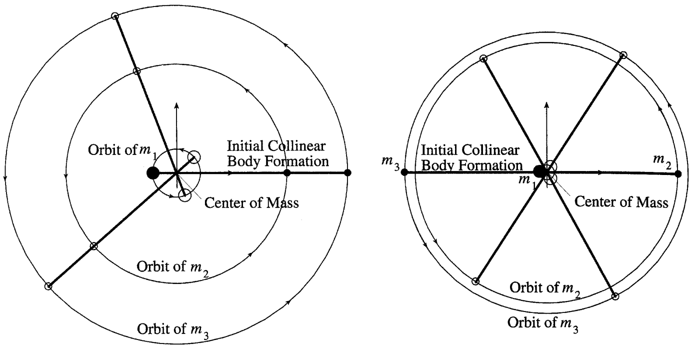


Figure 1.3 Equilateral triangle solution with circular orbits. From [2].



(a) m_3 on the far side of m_2

(b) m_3 on the far side of m_1

Figure 1.4 Collinear solutions with circular orbits. From [2].

restricted three-body problem, CR3BP from here on out. The problem itself has been extensively studied, and analytical solutions for stationary points can be deducted from the equations of motion (Lagrange Libration Points, a degenerate case of the previously shown circular solutions of the general problem viewed from a rotating frame), but the main interest is in the periodic orbits around two body systems as the Earth-Moon or Sun-Earth systems, in order to design missions that makes use of such practical orbits. Because of this, the main goal of this work is to find said periodic orbits and study the invariant manifolds associated, all of which can be applied to the design of missions, such as the *Deep Space Gateway* [3]. To do this we have to rely on numerical methods, and continuation, as we will see, has many advantages over other approaches. Continuation techniques rely on much of the work of *Henri Poincaré*, an so we will review its more relevant elements as a mean of historical introduction for the mathematical aspects treated here.

1.1 Poincaré's dynamical system theory

Henri Poincaré (1854-1912) was a famous mathematician that today is know as the last polymath [4], because he made significant contributions in multiple areas of mathematics and the physical sciences. His work in dynamical systems and topology is specialy important, and it will be the aspect to be commented in this chapter. His most important inventions [5] regarding dynamical systems are algebroid functions, index theory for plane ODEs, the Poincaré-Bendixon theorem, convergence of series solutions of ODEs, the use of the implicit function theorem, **bifurcation theory** (the Hopf bifurcation), asymptotic series, fixed point theorems for dynamical systems, homoclinic chaos...

Poincaré was educated mainly in geometry and analysis, but as has been stated, he didn't focus on one particular discipline. Still, his most important works are characterized by the interaction of analytical and geometrical thinking.

The Mémoire, 1881-82

It is concerned with two-dimensional problems. *The Mémoire* focus on autonomous second order equations, as many articles on ODEs at the time, but with an approach that differs broadly form others, to such point that today the philosophy of the study in nonlinear dynamics hasn't change much in it's generality and strategies. In here, *Poincaré* remarks the impossibility of integrating ODEs in general with known functions, and so divides the study of these systems in two parts:

1. Qualitative part, study of the geometry defined by the function

2. Quantitative part, numerical calculations

Here, *Poincaré* uses gnomic projection to analyse systems, which leads to the definition of the nowadays well-known *saddle*, *node*, *focus* and *centre*, the singularities of first type. This also develops in the basis of *index theory*, a tool for the study of cycles.

Another useful tool is the *'théorie of conséquents'*, what is now called the theory of *Poincaré maps*, this helps to understand high-dimensional ODEs and will be useful later.

The Prize Essay for Oscar II, 1888-89

This essay contains already fundamental theorems, and important results involve series expansions, **periodic solutions** and **bifurcations**. At the time, series expansions with respect to a small parameter were the main tool in celestial mechanics. *Poincaré* gave explicit criteria for the convergence and divergence of such series based on the implicit function theorem (IFT) and holomorphic expansion theorems. He also set-up a field of study which is centric in the present work, such us **bifurcation theory**. As it will be clear later on, bifurcations are the mathematical key to the Halo orbits that will be studied in some depth here.

Another basic result that develops in here, which in fact was a correction of the first version of the essay, is the *non-integrability of conservative systems*. At the time, it was thought that finding an integral of, for instance, the three-body problem, was a matter of analytical skill, and that all Hamiltonian systems were always integrable. In the first version he proved integrability of the CR3BP, identified an unstable periodic solution and approximated its stable and unstable manifolds by series expansions. *Poincaré* called these invariant manifolds *asymptotic surfaces*. He incorrectly concluded that the continuation of stable and unstable manifolds could be glued together to form a second first integral of the system. After finding about his mistake, he found out that this gluing was not possible in this particular case, and that there was in fact an infinite number of intersection between the manifolds (instead of merging).

Les Méthodes Nouvelles de la Mécanique Céleste, 1892-1899

This work contains the first general theory of dynamical systems describing both conservative and dissipative systems by analytical and geometrical methods. Despite the title, celestial mechanics is only used as a mean to illustrate the methods, and it is not the subject of study.

Poincaré used the IFT to demonstrate conditions for the convergence of series expansions in ODEs, with consequences for the bifurcation solutions, all off which was a new use for the IFT. He also introduces the notion of *bifurcation set*, which in Chap. 3 leads to a general discussion of what is now called the *Hopf bifurcation*. In this same chapter he implicitly points out the *"chaotic nature"* of the behaviour of an unstable periodic solution in a two degrees of freedom Hamiltonian system, as was previously shown.

1.2 Halo orbits and the Deep Space Gateway

Leaded by NASA and supported by the main space agencies in the world (ESA, JAXA, Roscosmos and CSA), the next step that will be taken in order to advance in the exploration of space starts by building a new space station in cislunar space[1], the *Deep Space Gateway* (DSG). The principal goal of this station is for the astronauts and the agencies to test the systems needed for missions in deep space (Mars, asteroids...), and this area of the Earth-Moon system offers a great opportunity to gain experience. It could also serve as a middle point between Earth and a Moon base. For this space station to be of practical use it needs to be placed in an orbit such that it's easily accessible and relatively cheap to maintain, and at the same time it has to serve as the *gateway* for the future missions of exploration in the Solar System. The study of periodical orbits in the CR3BP may shed some light on this issue.

Later it will be shown that many families of periodic orbits exist in the CR3BP, and one of these families is known for having a peculiar shape as seen from Earth. They are the **Halo orbits**, and as their name suggests they are seen as a halo around the Moon. There is also a great variety of halo orbits themselves, and they have been extensively studied in the context of space exploration [6]. Inside this group of orbits there is a subset that stands out for the purpose of having an inhabited station with deep space access [7], which are the *Near Rectilinear Halo Orbits*, or **NRHOs**. These orbits are at the edge of Earth's gravity well from an energetic point of view, which makes them ideal as stepping stone for deep space missions, and their stability properties are quite promising.

1.3 Objectives and scope of work

Continuation will serve here as a tool for studying these orbits and their dynamical properties in the CR3BP framework, in order to stablish the foundations for later studies with more detailed models. Although a complete mathematical justification of continuation isn't the scope of this work, some of the bases it relies on have been plainly established. This work tries to serve as a how-to basic guide in the ways the AUTO software could be used in the context of computing the dynamical structures around the CR3BP with continuation. It

[1] It is the volume within the Moon's orbit, beyond cislunar space lies translunar space

shows the basic utilisation of AUTO with the *Python* interface, the calculation of fixed points, cycles and finally the manifolds, as well as how to extract the relevant information of them from the output files that AUTO provides.

1.4 Structure of the document

Aside from this introduction, this work is composed of 4 more different chapters that address relevant aspects of the topics regarding the CR3BP. Chapter 2 serves as a revision of nonlinear dynamics theory that helps to understand concepts used later on, it also explains very few important concepts regarding bifurcation theory. Chapter 3 serves as a mathematical overview of continuation as a tool, it also explains some of the internal workings of the software AUTO, in order to help understand what happens inside the black box that this software will ultimately be for us. This chapter also exemplifies the strategies that will be used later to compute cycles, and specially manifolds, from a numerical point of view. Chapter 4 serves as a starting point for the reader to familiarise himself with the AUTO environment and capabilities, it tries to be the minimum amount of information needed for someone that wants to use the software in the context of this work and doesn't want to deal with the complete official manual. Chapter 5 is the study concerning the CR3BP using AUTO as main computational tool, it is fairly detailed in how the calculations have been done and it also analyses the viability of the NRHO as the future destination of the DSG, only in the context of the idealised circular restricted problem at hand.

2 Nonlinear dynamics and bifurcation theory

This chapter tries to stablish the main ideas that are used in the study of non-linear dynamical systems, and that will be useful in the later continuation study of the CR3BP. It will follow some of the topics discussed in [8] and [9].

2.1 Dynamical systems

The idea of dynamical systems arise from the mathematical conceptualization of a deterministic process. This means that given an initial state and the laws governing its evolution, the future states are completely defined. Because of this, the concept of dynamical system includes a set of its possible states (state space) and a law of the evolution in time. This way, if a point x that is part of the state space (or *phase space*) X is given, it is enough to describe the system in its actual "position" and the ones following it.

In a general way, the evolution of the system comes from an *evolution operator*, which for a given $t \in T$ can be defined as a map φ^t in the phase space X that:

$$\varphi^t : X \longrightarrow X$$

It transforms an initial state $x_0 \in X$ in some state $x_t \in X$ at time t:

$$x_t = \varphi^t x_0$$

The map φ^t might be known explicitly, but in most cases it is defined only indirectly. When the governing behaviour of the system does not change in time it is said that the system is autonomous, which can be expressed as:

$$\varphi^{t+s} = \varphi^t(\varphi^s x)$$

2.1.1 Orbits and phase portraits

Orbits are geometrical objects associated with a dynamical system, and phase portraits are a composition of these orbits. For a given x_0, the associated orbit (or trajectory) is:

$$Or(x_0) = \{x \in X : x = \varphi^t x_0, \textit{ for all } t \in T \textit{ such that } \varphi^t x_0 \textit{ is defined } \}$$

A point $x_0 \in X$ is called an *equilibrium* (fixed point) if $\varphi^t x_0 = x_0$ for all $t \in T$, thus, an evolution operator maps an equilibrium onto itself. The term equilibrium is usually reserved for continuous-time dynamical systems, while fixed point is more commonly used in discrete-time systems.

A cycle is a periodic orbit, so if L_0 is the orbit, then any point $x_0 \in L_0$ will satisfy $\varphi^{t+T_0} x_0 = \varphi^t x_0$ for some $T_0 > 0$ at any $t \in T$. If there are no other cycles in the neighbourhood then it is called a limit cycle. Finally, a phase portrait is a partitioning of the state space into orbits, revealing the behaviour of the system. An example of it can be seen in figure (2.1) for a continuous-time dynamical system.

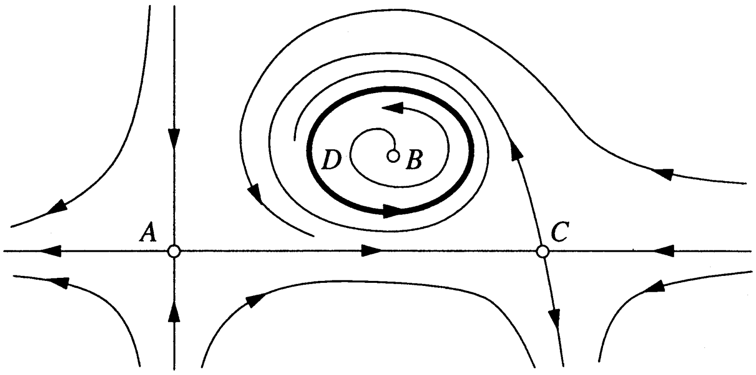


Figure 2.1 This image from [9] shows some of the most important features of a phase portrait: fixed points (A, B, C), cycles (D) and the different types of behaviour near fixed points.

2.1.2 Invariant sets

An invariant set of a dynamical system $\{T, X, \varphi^t\}$ is a subset $S \subset X$ such that $x_0 \in S$ implies $\varphi^t x_0 \in S$ for all $t \in T$. Orbits $Or(x_0)$ are invariant sets, and we could consider closed invariant sets in X, such as equilibria and cycles. The next more complex invariant sets are *invariant manifolds*, finite-dimensional hypersurfaces in some space $\mathbb{R}^K$, like the invariant torus of figure (2.2).

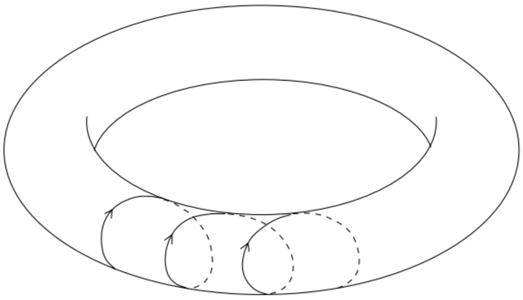
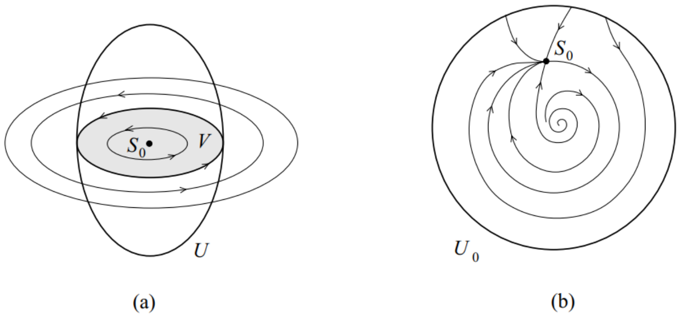


Figure 2.2 This image from [8] shows an invariant two-dimensional torus $\mathbb{T}^2$ of a continuous-time dynamical system in $\mathbb{R}^3$.

An invariant set S_0 is called stable if for any sufficiently small neighbourhood $U \supset S_0$ there exists a neighbourhood $V \supset S_0$ such that $\varphi^t x \in U$ for all $x \in V$ and all $t > 0$; or if there exist a neighbourhood $U_0 \subset S_0$ such that $\varphi^t x \to S_0$ for all $x \in U_0$, as $t \to +\infty$. The first type of stability is called Lyapunov stability, and the second asymptotic stability (figure (2.3) (a) and (b) respectively).



(a)

(b)

Figure 2.3 (a) Lyapunov stability versus (b) asymptotic stability. From [8].

2.1.3 Differential equations

Differential equations are the most common way to define a continuous-time dynamical system. For a state space $X = \mathbb{R}^n$ with coordinates $(x_1, x_2, \ldots, x_n)$ the laws of evolution are given implicitly in terms of velocities $\dot{x}_i$ as functions of the coordinates:

$$\dot{x}_i = f_i(x_1, x_2, \ldots, x_n),\ i = 1, 2, \ldots, n$$

or in the vector form

$$\dot{x} = f(x) \tag{2.1}$$

where the vector-valued function $f : \mathbb{R}^n \to \mathbb{R}^n$ is sufficiently differentiable (smooth). Equation (2.1) is a system of *n autonomous ordinary differential equations*, ODEs for short. For an isolated energy-conserving mechanical system with s degrees of freedom, the equations of motion can be determined by 2s Hamiltonian equations:

$$\dot{q}_i = \frac{\partial H}{\partial p_i}, \dot{p}_i = -\frac{\partial H}{\partial q_i}$$

for $i = 1, 2, \ldots, s$. The scalar function $H = H(q,p)$ is the Hamilton function.

The conditions for existence, uniqueness and smooth dependence of the function $x = x(t, x_0)$, $x : \mathbb{R}^1 \times \mathbb{R}^n \to \mathbb{R}^n$ that is solution to equation (2.1) are expressed with detail in [8]. The function of time $x = x(t, x_0)$ is called *solution starting at* x_0. It defines a *solution curve* $Cr(x_0)$ (time dependent), and an *orbit*, which is the projection onto the state space of the solution curve. Here, the evolution operator φ^t can be defined as

$$\varphi^t x_0 = x(t, x_0)$$

in an interval of t. Dynamical systems theory tries to analyse the behaviour of a dynamical system defined by ODEs. This can be done by simply computing many orbits numerically, however, this approach isn't very practical, and it is possible to predict some features of the phase portrait without actually having to solve the system. The first thing that can be done is to study the number and positions of equilibria by finding the solutions to

$$f(x) = 0 \tag{2.2}$$

It is also possible to study the stability of an equilibrium, and sufficient conditions are given by the following theorem (Lyapunov [1892]):

Consider a dynamical system defined by

$$\dot{x} = f(x), \quad x \in \mathbb{R}^n,$$

where f is smooth. Suppose that it has an equilibrium x^0 (i.e., $f(x^0) = 0$), and denote by A the Jacobian matrix of f(x) evaluated at the equilibrium, $A = f_x(x^0)$. Then x^0 is stable if all eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ of A satisfy Re $\lambda < 0$.

Where the eigenvalues are the roots of the characteristic equation ($det(A - \lambda I) = 0$). More complex properties such as cycles aren't so easy to study by just solving algebraic equations, but as we'll see later, there is a way with continuation.

2.2 Poincaré maps

It is not uncommon that discrete-time dynamical systems (maps) appear in the study of continuous-time systems defined by ODEs [10]. These maps arising from ODEs are called *Poincaré maps*.

2.2.1 Poincaré maps and stability of cycles

Consider a continuous dynamical system defined by

$$\dot{x} = f(x), \quad x \in \mathbb{R}^n$$

with smooth f. Given a periodic orbit L_0 and a point $x_0 \in L_0$ we introduce a *cross-section* Σ to the cycle at this point, see figure (2.4).

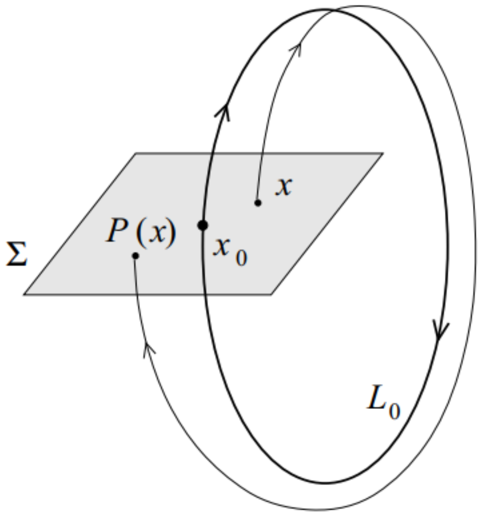


Figure 2.4 The Poincaré map associated with a cycle. From [8].

The cross-section Σ is a smooth hypersurface of dimension $n-1$, intersecting L_0 at an nonzero angle. Thus, the simplest choice of Σ is an hyperplane orthogonal to the cycle L_0 at x_0. Because L_0 is a cycle, and by definition of Σ, it starts at a point in Σ and returns to Σ at the same point. On the other hand, orbits near the cycle L_0 that start at a point $x \in \Sigma$ sufficiently close to x_0 also returns to Σ at some point $\tilde{x} \in \Sigma$ near x_0. Moreover, nearby orbits will also intersect Σ transversally. Thus, a map $P : \Sigma \to \Sigma$,

$$x \to \tilde{x} = P(x),$$

is constructed. The map P is called a Poincaré map associated with the cycle L_0. Let us introduce local coordinates $\xi = (\xi_1, \xi_2, \ldots, \xi_{n-1})$ on Σ such that $\xi = 0$ corresponds to x_0. Then the Poincaré map will be characterized by a locally defined map $P : \mathbb{R}^{n-1} \to \mathbb{R}^{n-1}$, which transforms ξ (corresponding to x) into $\tilde{\xi}$ (corresponding to $\tilde{x}$),

$$P(\xi) = \tilde{\xi}$$

.

Thus, the origin $\xi = 0$ is a fixed point of the map $P : P(0) = 0$. This way, studying the stability of the fixed point $\xi_0 = 0$ is the same as studying the stability of the cycle L_0. Then, the cycle is stable if all eigenvalues (multipliers) $\mu_1, \mu_2, \ldots, \mu_{n-1}$ of the $(n-1) \times (n-1)$ Jacobian matrix of P,

$$A = \frac{dP}{d\xi}\bigg|_{\xi=0}$$

are located *inside* the unit circle $|\mu| = 1$. It is also possible to probe that the result of the stability analysis is independent of the chosen cross-section Σ (see Lemma 1.3 of [8]).

There exists a relationship between the multipliers of a cycle and the differential equations (2.1), and for that it is necessary to construct the dynamical system for the perturbed motion around the cycle.

2.2.2 Variational equations of the cycle

Let $x^0(t)$ denote a periodic solution of (2.1), $x^0(t+T_0) = x^0(t)$, for the cycle L_0. Now consider a solution in the form

$$x(t) = x^0(t) + u(t)$$

with $u(t)$ a deviation from the periodic solution. Then, by linearising around $x^0(t)$

$$\dot{u}(t) = \dot{x}(t) - \dot{x}^0(t) = f(x^0(t) + u(t)) - f(x^0(t)) = A(t)u(t) + O(\|u(t)\|^2)$$

Truncating the $O(\|u(t)\|^2)$ terms results in the linear T_0-periodic system

$$\dot{u} = A(t)u, \quad u \in \mathbb{R}^n, \tag{2.3}$$

where $A(t) = f_x(x^0(t))$, such that $A(t+T_0) = A(t)$. The system (2.3) is called the variational equation about the cycle L_0. This is the linear part of the system governing the evolution of perturbations near the cycle and, naturally, the stability of the cycle depends on the properties of the variational equations.

Now consider the matrix initial-value problem

$$\dot{Y} = A(t)Y, \quad Y(0) = I_n,$$

where I_n is the unit $n \times n$ matrix. Its solution $Y(t)$ is called the *fundamental matrix of solutions* of the system, and at $t = T_0$ is the *monodromy matrix* of the cycle:

$$M = Y(T_0).$$

The determinant of the monodromy matrix can be calculated using the following *Liouville formula* in terms of $A(t)$:

$$\det M = \exp\left\{\int_0^{T_0} \operatorname{tr} A(t)dt\right\}.$$

The monodromy matrix is nonsingular. Any solution $u(t)$ to (2.3) satisfies

$$u(T_0) = Mu(0). \tag{2.4}$$

The eigenvalues of the monodromy matrix M are called the *Floquet multipliers* of the cycle

$$1, \mu_1, \mu_2, \ldots, \mu_{n-1},$$

where there is always a $+1$ and are the same as the multipliers of the Poincaré map associated with the cycle L_0. We also define the *exponents* associated as the λ_i such that $\mu_i = e^{\lambda_i T_0}$. Due to (2.4) any multiplier μ satisfies $v(T) = \mu v(0)$ with $v(0) \neq 0$ or, equivalently, it is solution component of the following BVP [11] on the unit interval:

$$\begin{cases} \dot{v} = TA(t)v, \\ v(1) = \mu v(0), \\ |v| = 1. \end{cases}$$

Assuming $\mu > 0$ we can write $\mu = e^{\lambda}$ and $v(t) = e^{\lambda t}w(t)$, so that w satisfies a periodic BVP such as

$$\begin{cases} \dot{w} = TA(t)w - \lambda w, \\ w(1) = w(0), \\ |w(0)| = 1, \end{cases}$$

where $w(t)$ is the *Floquet eigenfunction* corresponding to the multiplier μ. This will be useful when computing the unstable manifolds of the cycle.

2.3 Notes on bifurcation

The AUTO software is specialised in bifurcation analysis as we will see later on, but there is a special type of bifurcation that will be very useful when computing cycles through continuation, and these are the **Hopf bifurcations**. Suppose a three dimensional system has a stable fixed point, this can only mean that the eigenvalues of the Jacobian evaluated at said point are in the left half-plane Re $\lambda_i < 0$. For the Hopf bifurcation to exists there has to be at least one pair of complex conjugated eigenvalues so that its stability properties depend on a parameter μ, see figure (2.5). The Hopf bifurcation occurs when the eigenvalues cross the imaginary axis so that the fixed point losses its stability. The system is first damped (small disturbances decay over time), and as μ varies the decay becomes slower until it finally changes to a growth at a critical value μ_c. There are various types of Hopf bifurcation, which are exemplified in [9], but all of them share the crossing of imaginary axis property.

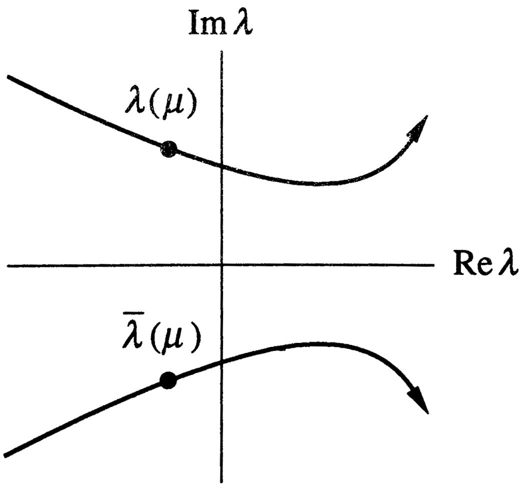


Figure 2.5 Hopf bifurcation representation.

The interesting property about this bifurcation is the fact that there is a value of μ at which the system becomes neutrally stable at the fixed point. At this value of μ the aforementioned eigenvalues are purely imaginary, which means that very close to the fixed point it is possible to linearise the system and express its solution as a sinusoidal motion, in other words, a cycle. This is a property that will be take advantage of when computing cycles in the CR3BP, as we will artificially transform the conservative system in a damped one, and compute the different cycles when the parameter generates the Hopf bifurcation, as AUTO is specially designed to detect them.

There are other bifurcations that will be important in our case of study, such as the families of orbits that growth from bifurcations, which AUTO can detect through monitoring the dimension of the Jacobian of the system.

3 Numerical continuation methods

This chapter will follow the topics discused in Chap. 1 and 9 of [12] to illustrate the internal workings of the software AUTO in the continuation of solutions and the computation of periodic orbits, as well as the more complex computation of invariant manifolds that will be shown later.

3.1 Theoretical support of continuation

When it comes to the study of stationary or periodic solutions of non-linear systems, numerical methods stand over analytical approaches because of its versatility, but numerical integration isn't always the way to go. Instead, numerical continuation presents itself as a very powerful and efficient tool.

The theoretical background that supports continuation techniques in software packages like AUTO are detailled in [12], but the basic pillars of it will be succinctly shown in this section.

First of all, let's define a Banach space $\mathscr{B}$ as a complete, normed vector space, here assumed to be $\mathbb{R}^n$. For $\mathbf{x}_0 \in \mathscr{B}$, we denote by $S_\rho(\mathbf{x}_0)$ the closed ball of radius ρ centered at $\mathbf{x}_0$, that is,

$$S_\rho(\mathbf{x}_0) = \{\mathbf{x} \in \mathscr{B} \mid \|\mathbf{x} - \mathbf{x}_0\| \leq \rho\}$$

Then, the existence and uniqueness of solutions is obtained by using two theorems: *Contraction Theorem* and *Implicit Function Theorem*, enunciated below.

Theorem 1: Contraction Theorem *Consider a continuous function $F : \mathscr{B} \to \mathscr{B}$ on a Banach space $\mathscr{B}$ and suppose that for some $\boldsymbol{x}_0 \in \mathscr{B}$, $\rho > 0$, and some K_0 with $0 \leq K_0 \leq 1$, we have*

$$\|F(\mathbf{u}) - F(\mathbf{v})\| \leq K_0\|\mathbf{u} - \mathbf{v}\|, \forall \mathbf{u}, \mathbf{v} \in S_\rho(\mathbf{x}_0),$$
$$\|F(\mathbf{x}_0) - \mathbf{x}_0\| \leq (1 - K_0)\rho.$$

Then the equation

$$\mathbf{x} = F(\mathbf{x}), \quad \mathbf{x} \in \mathscr{B},$$

has one and only one solution $\boldsymbol{x}_* \in S_\rho(\boldsymbol{x}_0)$, and $\boldsymbol{x}_*$ is the limit of the sequence

$$\mathbf{x}_{k+1} = F(\mathbf{x}_k), \quad k = 0, 1, 2, \ldots.$$

Theorem 2: Implicit Function Theorem *Let $\boldsymbol{G} : \mathscr{B} \times \mathbb{R}^m \to \mathscr{B}$ satisfy:*

- *$\boldsymbol{G}(\boldsymbol{u}_0, \boldsymbol{\lambda}_0) = \mathbf{0}$ for $\boldsymbol{u}_0 \in \mathscr{B}$ and $\boldsymbol{\lambda}_0 \in \mathbb{R}^m$*
- *$\boldsymbol{G}_u(\boldsymbol{u}_0, \boldsymbol{\lambda}_0)$ is not singular with bounded inverse,*

$$\|\mathbf{G}_u(\mathbf{u}_0, \boldsymbol{\lambda}_0)^{-}1\| \leq M$$

for some $M > 0$;

- *$\boldsymbol{G}$ and $\boldsymbol{G_u}$ are Lipschitz continuous, that is, for all $\boldsymbol{u},\boldsymbol{v} \in S_\rho(\boldsymbol{u}_0)$, and for all $\boldsymbol{\lambda},\boldsymbol{\mu} \in S_\rho(\boldsymbol{\lambda}_0)$ the following inequalities hold for some $K_L > 0$:*

$$\|\mathbf{G}(\mathbf{u},\boldsymbol{\lambda}) - \mathbf{G}(\mathbf{v},\boldsymbol{\mu})\| \leq K_L(\|\mathbf{u}-\mathbf{v}\| + \|\boldsymbol{\lambda}-\boldsymbol{\mu}\|),$$

$$\|\mathbf{G}_u(\mathbf{u},\boldsymbol{\lambda}) - \mathbf{G}_u(\mathbf{v},\boldsymbol{\mu})\| \leq K_L(\|\mathbf{u}-\mathbf{v}\| + \|\boldsymbol{\lambda}-\boldsymbol{\mu}\|),$$

Then there exist δ, with $0 < \delta \leq \rho$, and a unique function $\boldsymbol{u}(\boldsymbol{\lambda})$ that is continuous on $S_\delta(\boldsymbol{\lambda}_0)$, with $u(\boldsymbol{\lambda}_0) = \boldsymbol{u}_0$, such that

$$\mathbf{G}(\mathbf{u}(\boldsymbol{\lambda}),\boldsymbol{\lambda}) = 0, \forall \boldsymbol{\lambda} \in S_\delta(\boldsymbol{\lambda}_0).$$

If $\mathbf{G}(\mathbf{u},\boldsymbol{\lambda}_0) = 0$ and if $\mathbf{G_u}(\mathbf{u}_0,\boldsymbol{\lambda}_0)$ is invertible with bounded inverse, then $\mathbf{u}_0$ is called an *isolated solution* of $\mathbf{G}(\mathbf{u},\boldsymbol{\lambda}_0) = 0$. Hence, the IFT states that isolation (plus Lipschitz countinuity assumptions) implies the existence of a locally unique *solution family* (or *solution branch*) $\mathbf{u} = \mathbf{u}(\boldsymbol{\lambda})$, with $\mathbf{u}(\boldsymbol{\lambda}_0) = \mathbf{u}_0$. It is also possible to prove that the solution family $\mathbf{u}(\boldsymbol{\lambda})$ is continuously differentiable [12].

3.2 Continuation of solutions

It is very common to face problems dependent of some parameters, or at least one. Because of this, and without loss of generality, we can consider the continuation of a solution for the one-parameter equation

$$\mathbf{G}(\mathbf{u},\lambda) = \mathbf{0}, \quad \mathbf{u}, \mathbf{G}(\cdot,\cdot) \in \mathbb{R}^n, \quad \lambda \in \mathbb{R}.$$

With $\mathbf{x} \equiv (\mathbf{u},\lambda)$ the equation can be rewritten as

$$\mathbf{G}(\mathbf{x}) = \mathbf{0}, \quad \mathbf{u}, \mathbf{G} : \mathbb{R}^{n+1} \to \mathbb{R}.$$

Let's first see what are the conditions for a solution $\mathbf{x}_0$ to be regular, meaning that continues to exists in its vicinity as a unique one-dimensional *solution family* (or *solution branch*). Given the $n \times (n+1)$ matrix $\mathbf{G}^0_\mathbf{x} \equiv \mathbf{G}_\mathbf{x}(\mathbf{x}_0)$, then $\mathbf{x}_0$ solution of $\mathbf{G}(\mathbf{x}) = \mathbf{0}$ is *regular* if $\mathbf{G}^0_\mathbf{x}$ has maximal rank, which is n. With the parameter formulation we can differentiate two cases for $\mathbf{G}^0_\mathbf{x}$ having maximum rank:

$$\text{Rank}(\mathbf{G}^0_\mathbf{x}) = \text{Rank}(\mathbf{G}^0_\mathbf{u}|\mathbf{G}^0_\lambda) = n \Leftrightarrow \begin{cases} (i)\ \mathbf{G}^0_\mathbf{u} \text{ is non singular,} \\ \text{or} \\ (ii) \begin{cases} \dim \mathscr{N}(\mathbf{G}^0_\mathbf{u}) = 1, \\ \text{and} \\ \mathbf{G}^0_\lambda \notin \mathscr{R}(\mathbf{G}^0_\mathbf{u}) \end{cases} \end{cases}.$$

Where $\mathscr{N}(\mathbf{G}^0_\mathbf{u})$ is the *null space* of $\mathbf{G}^0_\mathbf{u}$ and $\mathscr{R}(\mathbf{G}^0_\mathbf{u})$ is the *range* of $\mathbf{G}^0_\mathbf{u}$.

In the first case the IFT provides us proof of existence of the solution $\mathbf{u} = \mathbf{u}(\lambda)$ near x_0. In the second case it is possible to interchange columns in the Jacobian $\mathbf{G}^0_\mathbf{x}$ to see that the solution can be parametrized by one of the components of $\mathbf{u}$. Notice that this second case is that of a simple fold, which is exemplified in figure (3.1).

3.2.1 Parameter continuation

In this first case we'll assume that the solution is solely define by the parameter λ and there are no folds (first case). Suppose we have a solution $(\mathbf{u}_0,\lambda_0)$ of

$$\mathbf{G}(\mathbf{u},\lambda) = \mathbf{0}$$

as well as the direction vector $\dot{\mathbf{u}}_0 = d\mathbf{u}/d\lambda$, and we want to compute the solution $\mathbf{u}_1$ at $\lambda_1 = \lambda_0 + \Delta\lambda$; as illustrated in figure (3.2). Then, we just apply the Newton's method

$$\begin{cases} \mathbf{G_u}(\mathbf{u}_1^{(\nu)},\lambda_1)\Delta\mathbf{u}_1^\nu = -\mathbf{G}(\mathbf{u}_1^\nu,\lambda_1), \\ \qquad \mathbf{u}_1^{\nu+1} = \mathbf{u}_1^\nu + \Delta\mathbf{u}_1^\nu, \qquad \nu = 0, 1, 2, \ldots. \end{cases}$$

Where the initial approximation comes from

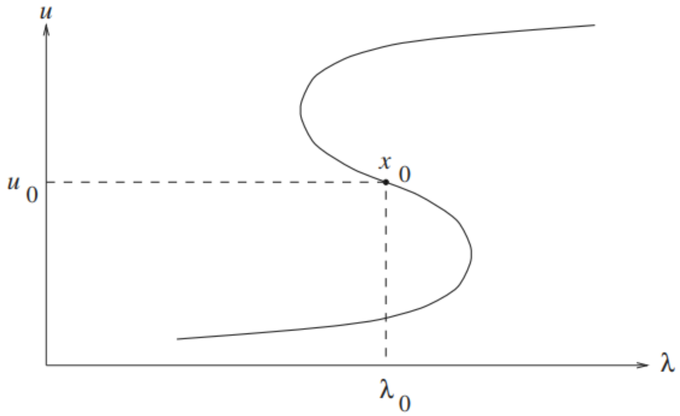


Figure 3.1 A solution branch with two folds. From [12].

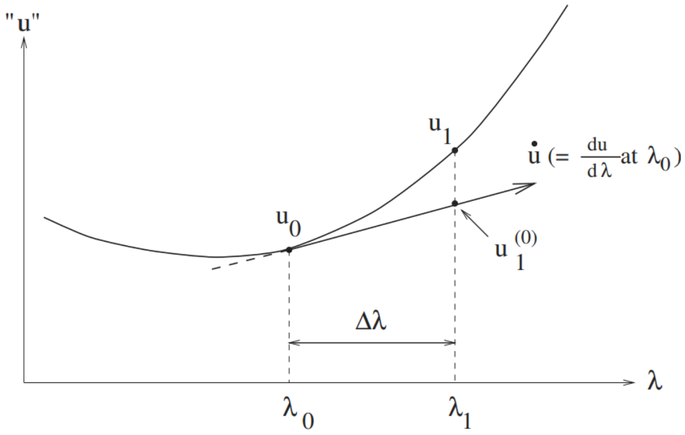


Figure 3.2 Interpretation of parameter continuation. From [12].

$$\mathbf{u}_1^0 = \mathbf{u}_0 + \Delta\lambda \dot{\mathbf{u}}_0.$$

The convergence of the iteration is guaranteed by the convergence theory for Newton's method if $\mathbf{G}_\mathbf{u}(\mathbf{u}_1,\lambda_1)$ is nonsingular. After that, the new direction vector $\dot{\mathbf{u}}_1$ can be computed by solving

$$\mathbf{G}_\mathbf{u}(\mathbf{u}_1,\lambda_1)\dot{\mathbf{u}}_1 = -\mathbf{G}_\lambda(\mathbf{u}_1,\lambda_1),$$

Which comes from differentiating $\mathbf{G}(\mathbf{u}(\lambda),\lambda) = \mathbf{0}$ with respect to λ at $\lambda = \lambda_1$.

3.2.2 Keller's pseudo-arclength continuation

The previous was valid if the solution doesn't have folds, so a new approach is required in general. To overcome this problem AUTO uses *Keller's Pseudo-Arclength Continuation*. Suppose we have a solution $(\mathbf{u}_0,\lambda_0)$ of $\mathbf{G}(\mathbf{u},\lambda)$, as well as the direction vectors $(\dot{\mathbf{u}}_0,\dot{\lambda}_0)$, where now the derivative is with respect to Δs at $\Delta s = 0$. Pseudo-arclength solves the following equations for $(\mathbf{u}_1,\lambda_1)$:

$$\begin{cases} \mathbf{G}(\mathbf{u}_1,\lambda_1) = \mathbf{0}, \\ (\mathbf{u}_1 - \mathbf{u}_0)^* \dot{\mathbf{u}}_0 + (\lambda_1 - \lambda_0)\dot{\lambda}_0 - \Delta s = 0 \end{cases} \tag{3.1}$$

This can be geometrically interpreted as seen in figure (3.3), which means that the projection of $(\dot{\mathbf{u}}_0,\dot{\lambda}_0)$ over $(\mathbf{u}_1 - \mathbf{u}_0,\lambda_1 - \lambda_0)$ is Δs. The equations (3.1) can be solved using Newton's method with the next iterative system:

$$\begin{pmatrix} (\mathbf{G}_{\mathbf{u}}^{1})^{(\nu)} & (\mathbf{G}_{\lambda}^{1})^{(\nu)} \\ \dot{\mathbf{u}}_0 * & \dot{\lambda}_0 \end{pmatrix} \begin{pmatrix} \Delta\mathbf{u}_1^{(\nu)} \\ \Delta\lambda_1^{(\nu)} \end{pmatrix} = \begin{pmatrix} \mathbf{G}(\mathbf{u}_1^{(\nu)},\lambda_1^{\nu}) \\ (\mathbf{u}_1^{(\nu)} - \mathbf{u}_0)^* \dot{\mathbf{u}}_0 + (\lambda_1^{(\nu)} - \lambda_0)\dot{\lambda}_0 - \Delta s \end{pmatrix},$$

With the direction vector defined as:

$$\begin{pmatrix} \mathbf{G}_{\mathbf{u}}^{1} & \mathbf{G}_{\lambda}^{1} \\ \dot{\mathbf{u}}_0^* & \dot{\lambda}_0 \end{pmatrix} \begin{pmatrix} \dot{\mathbf{u}}_1 \\ \dot{\lambda}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix},$$

After its computation the direction vector must be rescaled, so that its norm continues to be 1. This method can be efficiently solved if correctly implemented as shown in [12].

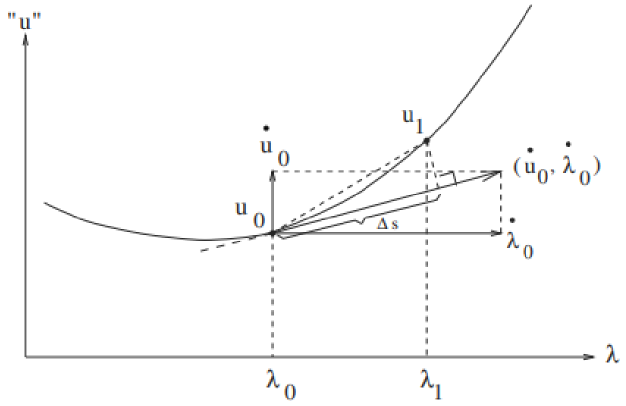


Figure 3.3 Graphical interpretation of pseudo-arclength method. From [12].

3.3 Boundary value problem

In a general way, AUTO solves the BVP of the following shape:

$$\mathbf{u}'(t) - \mathbf{f}(\mathbf{u}(t),\boldsymbol{\mu},\lambda) = \mathbf{0}, \quad t \in [0,1],$$

where "$'$" is a derivative with respect to time, and

$$\mathbf{u}(\cdot),\mathbf{f}(\cdot) \in \mathbb{R}^n, \quad \lambda \in \mathbb{R}, \quad \boldsymbol{\mu} \in \mathbb{R}^{n_\mu}$$

and subject to the next boundary conditions

$$\mathbf{b}(\mathbf{u}(0),\mathbf{u}(1),\boldsymbol{\mu},\lambda) = \mathbf{0}, \quad \mathbf{b}(\cdot) \in \mathbb{R}^{n_b},$$

and integral conditions if necessary for the problem

$$\int_0^1 \mathbf{q}(\mathbf{u}(s),\boldsymbol{\mu},\lambda)ds = \mathbf{0}, \quad \mathbf{q}(\cdot) \in \mathbb{R}^{n_q}$$

The goal is to solve the BVP for $\mathbf{u}(\cdot)$ and $\boldsymbol{\mu}$. For the problem to be well posed the next condition has to be true:

$$n_\mu = n_b + n_q - n \geq 0.$$

In this case the parameter λ is the continuation parameter in which the solution $(\mathbf{u},\boldsymbol{\mu})$ is continued. AUTO solves BVP using the method of *orthogonal collocation with piecewise polynomials*, as it is very accurate and allows adaptive mesh-selection. Here the general idea behind this method will be shown, for a more detailed explanation see [13] and [14].

First, a mesh is introduced between $t = 0$ and $t = 1$ (AUTO can change this mesh through the iterations if necessary):

$$\{0 = t_0 < t_1 \cdots < t_N = 1\},$$

with

$$h_j = t_j - t_{j-1}, \quad (1 \le j \le N).$$

Next, the space of (vector-valued) piecewise polynomials $\mathscr{P}_h^m$ is defined as:

$$\mathscr{P}_h^m = \{\mathbf{p}_h \in C[0,1] \mid \mathbf{p}_h|_{[t_{j-1},t_j]} \in \mathscr{P}^m\},$$

where $\mathscr{P}^m$ is the space of (vector-valued) polynomials of degree $m \le 3$. The collocation method consists in finding $\mathbf{p}_h \in \mathscr{P}_h^m$ and $\boldsymbol{\mu} \in \mathbb{R}^{n_\mu}$, such that the collocation equations are satisfied:

$$\mathbf{p}_h'(z_{j,i}) = \mathbf{f}(\mathbf{p}_h(z_{j,i}),\boldsymbol{\mu},\lambda), \quad j = 1,\cdots,N, \quad i = 1,\cdots,m,$$

and such that $\mathbf{p}_h$ satisfies the boundary and integral conditions. The collocation points $z_{j,1}$ in each subinterval $[t_{j-1},t_j]$ are the roots of the mth-degree orthogonal polynomial (Gauss points). For a graphical interpretation of it see the figure (3.4).

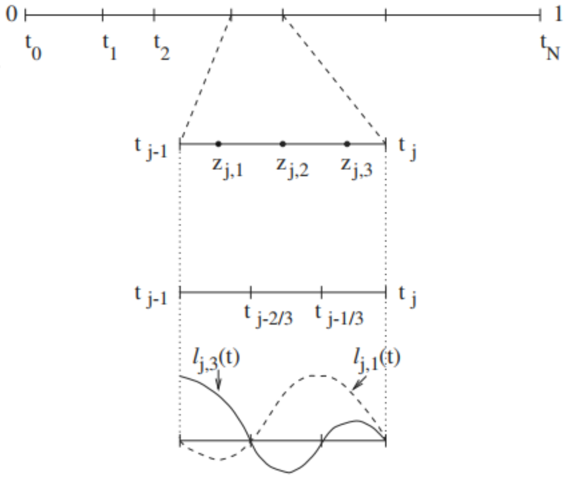


Figure 3.4 Here are shown the collocation points and the 'extended-mesh points' for the case m=3. Also, two of the local Lagrange basis polynomials are shown. From [12].

The implementation is done with Lagrange basis polynomials for each subinterval $[t_{j-1},t_j]$. We define

$$\{\ell_{j,i}(t)\}, \quad j = 1,\ldots,N, \quad i = 0,1,\ldots,m,$$

by

$$\ell_{j,i}(t) = \prod_{k=0,k\neq i}^{m} \frac{t - t_{j-\frac{k}{m}}}{t_{j-\frac{i}{m}} - t_{j-\frac{k}{m}}}$$

where

$$t_{j-\frac{i}{m}} = t_j - \frac{i}{m} h_j$$

Then, the local polynomials can be written as

$$\mathbf{p}_j(t) = \sum_{i=0}^{m} \ell_{j,i}(t)\mathbf{u}_{j-\frac{i}{m}}.$$

Then the collocation equations are

$$\mathbf{p}_j'(z_{j,i}) = \mathbf{f}(\mathbf{p}_j(z_{j,i}),\boldsymbol{\mu},\lambda), \quad i = 1,\dots,m, j = 1,\dots,N,$$

the discrete boundary conditions are

$$b_i(\mathbf{u}_0,\mathbf{u}_N,\boldsymbol{\mu},\lambda) = 0, i = 1,\dots,n_b,$$

and the integrals constraints can be discretized as

$$\sum_{j=1}^{N}\sum_{i=0}^{m} w_{j,i} q_k(\mathbf{u}_{j-\frac{i}{m}},\boldsymbol{\mu},\lambda) = 0, k = 1,\dots,n_q,$$

where the $w_{j,i}$ are the Lagrange quadrature coefficients. Given the solution of the previously computed point on the solution branch, $(\mathbf{u}_0,\boldsymbol{\mu}_0,\lambda_0)$ and the direction vector $(\dot{\mathbf{u}}_0,\dot{\boldsymbol{\mu}}_0,\dot{\lambda}_0)$, the discretized pseudo-arclength equation is

$$\sum_{j=1}^{N}\sum_{i=0}^{m} w_{j,i}[\mathbf{u}_{j-\frac{i}{m}} - (\mathbf{u}_0)_{j-\frac{i}{m}}]^*(\dot{\mathbf{u}}_0)_{j-\frac{i}{m}} + (\boldsymbol{\mu}-\boldsymbol{\mu}_0)^*\dot{\boldsymbol{\mu}}_0 + (\lambda-\lambda_0)\dot{\lambda}_0 - \Delta s = 0.$$

The shape of the system is illustrated in figure (3.5), where the number of collocation equations, mnN, continuity equations, $(N-1)n$ and constraints, $n_b + n_q$ $(= n + n_\mu)$, match the total number of degrees of freedom $(m+1)nN + n_\mu$ (with λ fixed).

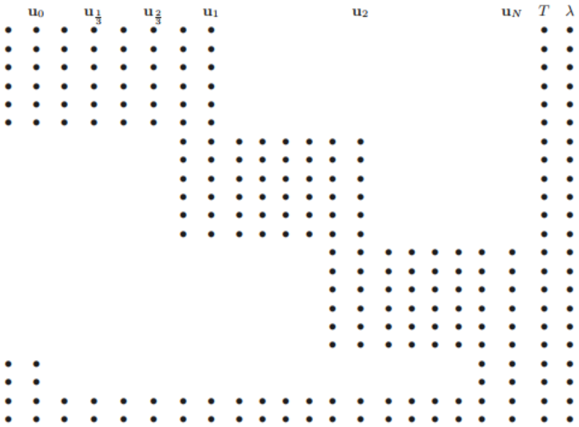


Figure 3.5 Structure of the system for the case of $n = 2$ differential equations with number of mesh intervals $N = 3$, number o collocation points per mesh interval $m = 3$, the number of boundary conditions $n_b = 2$, and the number of integral constraints $n_q = 1$, being the last row the pseudo-archlength equation. From [12].

AUTO solves these systems with an efficient method that includes the condensation of parameters by Gauss elimination done in parallel.

3.4 Computing cycles

Computing periodic solutions can be done using a boundary value approach. Consider the first order system

$$\mathbf{u}'(t) = \mathbf{f}(\mathbf{u}(t),\lambda), \quad \mathbf{u}(\cdot),\mathbf{f}(\cdot) \in \mathbb{R}^n, \quad \lambda \in \mathbb{R}.$$

In order to make the period of the solution 1 we apply the next transformation $t \to \frac{t}{T}$, so the equation becomes

$$\mathbf{u}'(t) = T\mathbf{f}(\mathbf{u}(t),\lambda), \quad \mathbf{u}(\cdot),\mathbf{f}(\cdot) \in \mathbb{R}^n, \quad T,\lambda \in \mathbb{R}. \tag{3.2}$$

This way, the solutions we seek are those in which

$$\mathbf{u}(0) = \mathbf{u}(1). \tag{3.3}$$

The period T is one of the unknowns of the system. Suppose that we have already calculated a solution $(\mathbf{u}_{k-1}(\cdot), T_{k-1}, \lambda_{k-1})$ and we want to compute the next $(\mathbf{u}_k(\cdot), T_k, \lambda_k)$. Then, if it were only for the last two equations, a translated solution from the last one is also a solution: $\mathbf{u}_k(t) = \mathbf{u}_{k-1}(t+\sigma)$. Thus, a *phase condition* is needed to arrive at the next solution. One option is to impose a displacement using the Poincaré orthogonality condition (see figure (3.6)):

$$(\mathbf{u}_k(0) - \mathbf{u}_{k-1}(0))^* \mathbf{u}'_{k-1}(0) = 0,$$

though a numerically more suitable phase condition is

$$\int_0^1 \mathbf{u}_k(t)^* \mathbf{u}'_{k-1}(t) dt = 0. \tag{3.4}$$

This is deducted in [12] and it's called the integral phase condition, which is what AUTO has implemented.

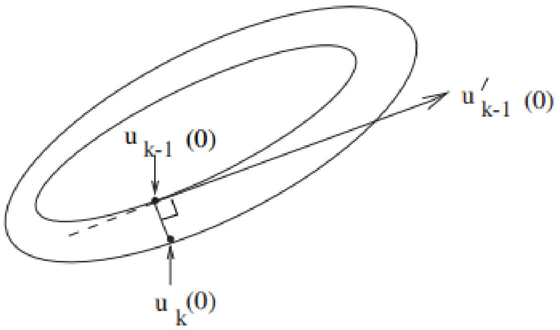


Figure 3.6 Graphical interpretation of the Poincaré phase condition. From [12].

The continuation of a family of periodic solutions is also done with the pseudo-arclength method, as this allows calculation past folds. It also has important applications to the computation of periodic solutions to conservative systems, as it allows calculations of a 'vertical family' of periodic solutions. For periodic solutions the pseudo-arclength equation is:

$$\int_0^1 (\mathbf{u}_k(t) - \mathbf{u}_{k-1}(t))^* \dot{\mathbf{u}}_{k-1}(t) dt + (T_k - T_{k-1})^* \dot{T}_{k-1} + (\lambda_k - \lambda_{k-1}) \dot{\lambda}_{k-1} = \Delta s. \tag{3.5}$$

Equations (3.2)-(3.5) are the ones AUTO solves for the continuation of periodic solutions in general. In addition, during the continuation, the **Flouquet multipliers** of the cycles are monitored by computing a special decomposition of the monodromy matrix that arises as a by-product of the decomposition of the Jacobian of the collocation system.

3.4.1 Cycles in conservative systems, unfolding parameter

The approach shown until now for the continuation of periodic orbits in dynamical systems is valid in general, but in the case of conservative systems with one conserved quantity the strategy followed uses a parameter with a special meaning.

Rather than perform the continuation using the conserved quantity, which would be a reduction method, product of the cylinder theorem, the alternative is about increasing the dimension of the system, adding an *unfolding parameter*. The additional term is defined in a way that the periodicity of the solution can only exist when the parameter is 0, so that looking for the cycle is equivalent to looking for a "vertical" Hopf bifurcation.

The system with the unfolding term becomes:

$$\mathbf{u}'(t) = T\mathbf{f}(\mathbf{u}(t), \lambda) + \alpha \mathbf{d}(\mathbf{u}(t)), \quad \alpha \in \mathbb{R},$$

where α is the unfolding parameter, which acts as a dissipation (positive or negative), and that will be $\alpha = 0$ in the periodic solution. The system can now be expressed as:

$$\mathbf{u}'(t) = T\mathbf{f}(\mathbf{u}(t),\alpha).$$

All the previous additional equations remains the same. With this set-up, the family of periodic solutions will be that in which $\alpha = 0$ (to numerical precision), so that the bifurcation diagram in α looks like the figure (3.7).

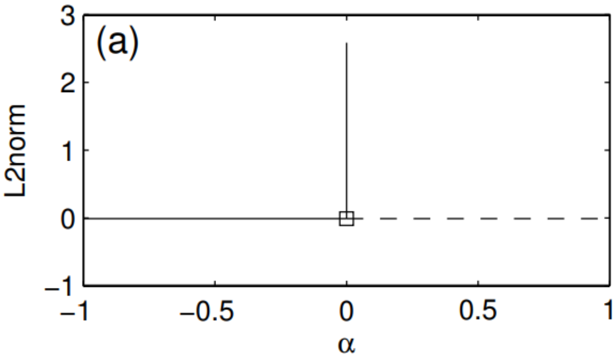


Figure 3.7 Bifurcation diagram in α with the vertical branch of periodic solutions. From [12].

3.5 Computing manifolds

The method followed here is the one we will use in AUTO for the computation of manifolds in *three* dimensional systems (as the CR3BP) even though it could be done for any kind of system with periodic solutions.

We will first assume that the periodic orbit has a single, real Floquet multiplier outside the unit circle, so that it gives rise to a *two-dimensional unstable manifold* of the cycle in phase space. In this case, the eigenfunction associated to this multiplier gives a linear approximation to the manifold close to the periodic orbit. As stated in chapter 2 this eigenfunction can be computed as a solution of the BVP

$$\begin{cases} \mathbf{w}'(t) = T\mathbf{f}_{\mathbf{u}}(\mathbf{u}(t),0)\mathbf{w}(t) - \lambda\mathbf{w}(t), \\ \mathbf{w}(0) = s\mathbf{w}(1), \\ |\mathbf{w}(0)| = \sqrt{\rho}, \end{cases}$$

where $s = +1$ if the multiplier is positive and $s = -1$ for a negative one. The Floquet multiplier is then se^{λ}. We have impose the norm on the normalized eigenfunction at $t = 0$ to be $\sqrt{\rho}$, where typically $\rho = 1$. Only when there is one real and greater than one in absolute value Floquet multiplier then this problems results in an unique unstable eigenfunction $\mathbf{w}(t)$. This algorithm applies equally well to stable manifolds, in which case there would only be one real Floquet multiplier with absolute value less than one. We will also restrict to the case $s = 1$, so that the Floquet multiplier is positive, and the corresponding manifold is orientable rather than twisted. A liner approximation to the unstable manifold at time zero is given by

$$\mathbf{r}(0) = \mathbf{u}(0) + \varepsilon\mathbf{w}(0) \tag{3.6}$$

for ε small. Although this procedure would indeed result in the needed eigenfunction, the actual strategy followed is a bit different. Given the periodic orbit $\mathbf{u}_0(t)$, from which we want to compute the unstable manifold, the system that is solved is the one given by

$$\begin{aligned}
&\mathbf{u}'(t) = T\mathbf{f}(\mathbf{u}(t),\alpha),\\
&\mathbf{u}(0) = \mathbf{u}(1),\\
&\int_0^1 \mathbf{u}(t)^*\mathbf{u}_0'(t)dt = 0.\\
&\mathbf{w}'(t) = T\mathbf{f}_{\mathbf{u}}(\mathbf{u}(t),0)\mathbf{w}(t) - \lambda\mathbf{w}(t),\\
&\mathbf{w}(0) = \mathbf{w}(1),\\
&|\mathbf{w}(0)| = \sqrt{\rho},\\
&\int_0^1 (\boldsymbol{\zeta}(t) - \boldsymbol{\zeta}_0(t))^*\dot{\boldsymbol{\zeta}}_0(t)dt + (\mathbf{p}-\mathbf{p}_0)^*\dot{\mathbf{p}}_0 = \Delta s, \quad \boldsymbol{\zeta}(t) = (\mathbf{u}(t),\mathbf{w}(t)), \quad \mathbf{p} = (\alpha,\lambda,\rho),
\end{aligned}$$

where $\mathbf{w}(t)$ and ρ are initialized to zero. This way, the cycle corresponds to a bifurcation, where branch switching gives the non-zero eigenfunction $\mathbf{w}(t)$. This method, although elaborate, allows for the calculation of the eigenfunction in sensitive cases, as for example when the Floquet multipliers are either very large or small. Now that we have the eigenfunction, the computation of the manifold is as proceed [15].

The method consists basically in integrating orbits that lie in the manifold until a certain condition is matched. We start with the conditions in the linear approximation of the manifold (see equation (3.6)), and perform continuation using the integration time as a free parameter, then stop the continuation when the desired condition is achieved. To go down all the manifold it is important that the range of the parameter ε corresponds to a *fundamental domain* $[\varepsilon_1, \varepsilon_2)$, meaning that the orbit that starts at $\mathbf{r}(0) = \mathbf{u}(0) + \varepsilon_1\mathbf{w}(0)$ closely passes the line given by $\mathbf{r}(0) = \mathbf{u}(0) + \varepsilon_2\mathbf{w}(0)$ for the first time. Given a certain ε_1, that has to be sufficiently small so that the manifold is correctly approximated, we can compute the corresponding linear approximation of ε_2 using $\varepsilon_2 = se^{\lambda}\varepsilon_1$. A possible stopping criteria could be that the orbit ends in a chosen section Σ. The part of the manifold to be approximated is then given by the set of orbits:

$$\{\mathbf{r}(t)\,|\,\mathbf{u}(0) + \varepsilon\mathbf{w}(0) \text{ and } \mathbf{r}(1) \in \Sigma, \text{ for } \varepsilon_1 \leqslant \varepsilon \leqslant \varepsilon_2\}$$

Where time has been rescaled so that the entire finite integration interval becomes the unit interval. The boundary value problem for $\mathbf{r}(t)$ is given by:

$$\begin{aligned}
&\mathbf{r}'(t) = T_r\mathbf{f}(\mathbf{r}(t),0),\\
&\mathbf{r}(0) = \mathbf{u}(0) + \varepsilon\mathbf{w}(0),\\
&\mathbf{r}(1)_x = x_\Sigma,\\
&\int_0^1 (\mathbf{r}(t) - \mathbf{r}_0(t))^*\dot{\mathbf{r}}_0(t)dt + (\varepsilon - \varepsilon_0)^*\dot{\varepsilon}_0 + (T_r - T_{r0})\dot{T}_{r0} = \Delta s,
\end{aligned}$$

where $\alpha = 0$ and Σ denotes the section $x = x_\Sigma$. The last requires a starting orbit to be computed using time integration.

4 AUTO

This chapter tries to exemplify the most important features of AUTO regarding non-linear dynamics and shows how to use it with the *Python interface*, so that using this minimun information the reader should be able to work with the software. It is a selection of the most relevant parts of the AUTO manual [16] regarding the subject of this work, so a more detailed description of the software capabilities (quite extensive) can be found in there if necessary.

4.1 Capabilities of the software

AUTO can perform bifurcation analysis of algebraic systems of the form

$$f(u,p) = 0, \qquad f(\cdot,\cdot), u \in \mathbb{R}^n.$$

Although the main algorithms in AUTO are aimed at the continuation of solutions of ODE's of the form

$$u'(t) = f(u(t),p), \qquad f(\cdot,\cdot), u(\cdot) \in \mathbb{R}^n,$$

subject to boundary and integral constrains as it has already be explained. Regarding algebraic systems AUTO can:

- Compute solution of families.
- Locate branch points and automatically compute bifurcating families.
- Locate Hopf bifurcation points and detect its properties.
- Locate folds.
- Find extrema in an objective function and continue it in more parameters.

Regarding the study of ordinary differential equations the list is longer, but here there is a brief overview:

- Compute families of stable and unstable periodic solutions and compute the Floquet multipliers that determines stability along these families. The starting data to compute this orbits can be generated automatically at Hopf bifurcation points.
- Compute folds, period-doubling bifurcations, and bifurcations to tori, in two parameters, detecting 1:1, 1:2, 1:3 and 1:4 resonances.
- Follow curves of homoclinic orbits and detect and continue various codimension-2 bifurcations.
- Locate extrema along a family of periodic solutions and continue them in more parameters.

4.2 User-supplied elements

4.2.1 Files

AUTO needs *two* file types for a problem to be defined, one is the *equations file*, and the other is the *constants file*.

The equations file is either a $xxx.f90$, $xxx.f$ or $xxx.c$ file, depending on the programming language utilised (here xxx stands for the name of the problem). The termination ($.f$) corresponds to fixed-form Fortran, ($.f90$) is in free-form Fortran and ($.c$) is written in C. All of them are valid but the format ($.f90$) is the one used in this work.

The file $xxx.f90$ contains the Fortran routines that will be explained later, so that if any of them is irrelevant for the problem then it doesn't have to be completed. Many examples can be found in the folder $auto/07p/demos$. For a new problem, the simplest way to create its equation file is to copy the one corresponding to a similar problem and modify the appropriate lines of code.

The constants file is a $c.xxx$ file, where xxx doesn't necessarily have to coincide with the name of the equations file, because the constants file used can be specified when a problem is executed, though it has to be contained in the same folder as the $xxx.f90$ file. The meaning of many of these constants will be explained in section (4.2.3).

4.2.2 Routines

The purpose of each of the user-supplied routines in the $xxx.f90$ file is described bellow, although a good way to familiarise yourself with them is to check the code of the demo problem $cusp$, as it is fully documented.

- $FUNC$: Defines the function $f(u,p)$ of the problem.
- $STPNT$: It defines the starting solution (u,p), which should not be a branch point. This routine is only called when the constant $IRS = 0$, which is usually the case of the first run of the problem, otherwise it sets the label of the solution where the computation is to be restarted.
- $BCND$: Defines the boundary conditions, see demo exp or kar.
- $ICND$: Defines the integral conditions, see demo int or lin.
- $FOPT$: Defines the objective functional if needed, see demo opt or ops.
- $PVLS$: This routine is for "solution measures", meaning that it is used when the user wants to specify a given output function of the problem variables, such as norm of the solution, minimum, value at a boundary...

4.2.3 Constants and parameters

As explained before, the constants that define the problem are expected to appear in the $c.xxx$ file, but this file is not strictly necessary when using the Python interface, because you can define all the constants inside the scripts when calling for the problem to run.

In the case that the file is used, the format in it is free, with $constant = value$ separated by comas or spaces. An example with *default* values is in figure (4.1), though in real constant files you only need to specify the values that differ from these.

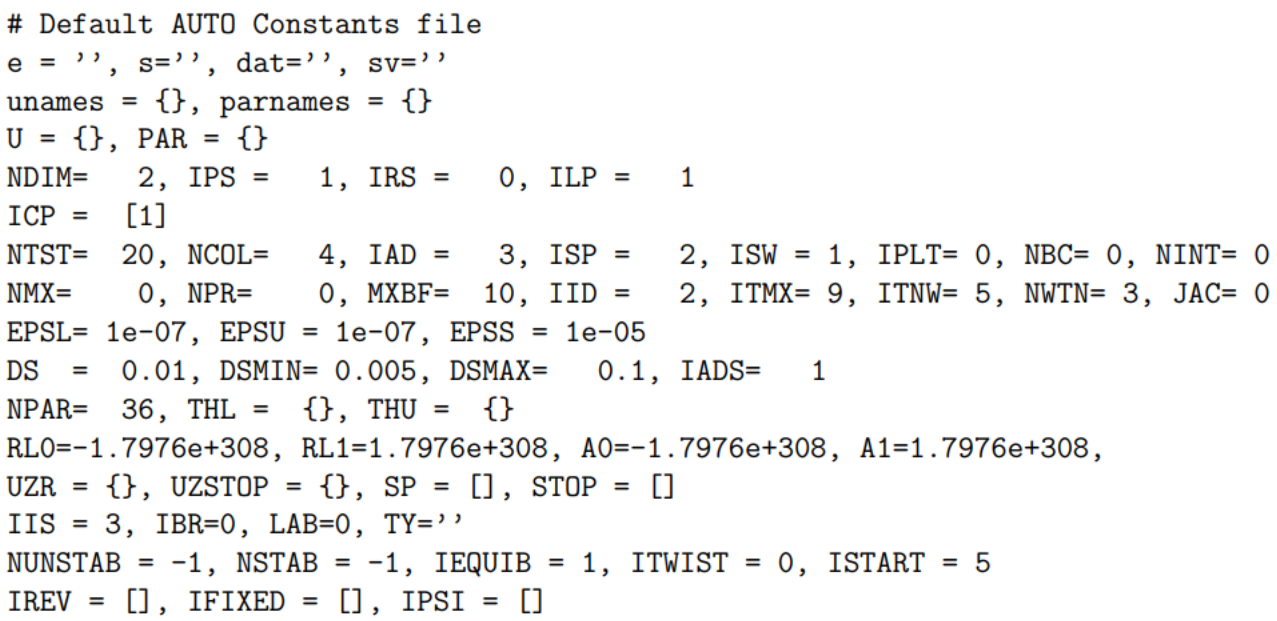
A brief overview of some of the constants is next.

Problem constants

- $NDIM$: Dimension of the system of equations.
- NBC: Number of boundary conditions.
- $NINT$: Number of integral conditions.
- $NPAR$: Maximum number of parameters that can be used in all user-supplied routines.
- JAC: Used to indicate whether derivatives are supplied by the user or to be obtained by differencing, such that when $JAR = 0$ no derivatives are provided by the user.

Discretization constants

- $NTST$: The number of mesh intervals used for discretization, it remains fixed for any particular run and it's recommended to be as small as possible to maintain convergence.
- $NCOL$: The number of Gauss collocation points per mesh interval, ($2 \leqslant NCOL \leqslant 7$). $NCOL = 4$ is a recommended value in most cases.
- IAD: It controls the mesh adaptation, $IAD = 0$ for fixed mesh and $IAD > 0$ to adapt mesh every IAD steps along the family.



```
# Default AUTO Constants file
e = '', s='', dat='', sv=''
unames = {}, parnames = {}
U = {}, PAR = {}
NDIM=   2, IPS =   1, IRS =   0, ILP =   1
ICP =  [1]
NTST=  20, NCOL=   4, IAD =   3, ISP =   2, ISW = 1, IPLT= 0, NBC= 0, NINT= 0
NMX=    0, NPR=    0, MXBF=  10, IID =   2, ITMX= 9, ITNW= 5, NWTN= 3, JAC= 0
EPSL= 1e-07, EPSU = 1e-07, EPSS = 1e-05
DS  =  0.01, DSMIN= 0.005, DSMAX=   0.1, IADS=   1
NPAR=  36, THL =  {}, THU =  {}
RL0=-1.7976e+308, RL1=1.7976e+308, A0=-1.7976e+308, A1=1.7976e+308,
UZR = {}, UZSTOP = {}, SP = [], STOP = []
IIS = 3, IBR=0, LAB=0, TY=''
NUNSTAB = -1, NSTAB = -1, IEQUIB = 1, ITWIST = 0, ISTART = 5
IREV = [], IFIXED = [], IPSI = []
```

Figure 4.1 Default values for the AUTO constants file.

Tolerances

- $EPSL$: Relative convergence criterion for equation parameters in the Newton/Chord method.
- $EPSU$: Relative convergence criterion for solution components in the Newton/Chord method.
- $EPSS$: Relative arc-length convergence criterion for the detection of special solutions.
- $ITMX$: Maximum number of iterations allowed in the accurate location of special solutions.
- $NWTN$: After $NWTN$ Newton iterations the Jacobian is frozen, because AUTO uses full Newton for the first $NWTN$ iterations and the Chord method for iterations $NWTN+1$ to $ITNW$, which is the maximum number of combined Newton-Chord iterations.

Continuation step size

- DS: It defines the pseudo-arclength step size to be used for the first attempted step along any family.
- $DSMIN$: Minimum allowable absolute value of the pseudo-arclength step size.
- $DSMAX$: Maximum allowable absolute value of the pseudo-arclength step size.
- $IADS$: It controls the frequency of the step size adaptation. Adapts the step size every $IADS$ steps.
- THL: Defines the parameters whose weights is to be modified in the pseudo-arclength continuation, so that not all the parameters change in the same way (or cannot change at all).

Limit the continuation

- $STOP$: This adds stopping conditions, for example stopping at the third Hopf bifurcation.
- NMX: The maximum number of steps taken along any family.
- $RL0$: The lower bound on the principal continuation parameter.
- $RL1$: The upper bound on the principal continuation parameter.
- $A0$: The lower bound on the principal solution measure.
- $A1$: The upper bound on the principal solution measures.

Others stopping options will be shown later on.

Free parameters

The array *ICP* designates the free parameters of the problem. The parameters that appears first is the *principal continuation parameter*. As an example, consider the case of the continuation of a solution to the system $f(u,p) = 0$, where there is only one free parameter. In this case $ICP = [1]$, so the variable $PAR(1)$ is designated the free parameter. Consider now the continuation of a family of periodic orbit, where there is also a free parameter. In this case it is usual to consider $ICP = [1,\ 11]$, as in AUTO the parameter $PAR(11)$ is reserved for the period of the solution. If the period is not specified in a periodic solution continuation problem, AUTO will automatically add it, but it will not appear in the screen output or output-file (*fort.7*). Other cases are specified in the AUTO manual.

Computation constants

- *ILP*: Controls the detection of folds. $ILP = 1$ for the detection of folds, recommended when fold continuation is intended.
- *ISP*: Controls the detection of Hopf bifurcations, branch points, period-doubling bifurcations and torus bifurcations. Look up the manual for more details.
- *ISW*: Controls the branch switching for the case of differential equations.
- *s*: Sets the name of the solution file from which the computation is to be restarted.
- *IRS*: This constant sets the label of the solution where the computation is to be restarted. As stated before, it is $IRS = 0$ for the first run, and $IRS > 0$ to restart the computation at the previously computed solution with label *IRS*.
- *IPS*: This is one of the most important constants to be set correctly, as it defines the problem type. $IPS = 0$ for algebraic bifurcation problems, $IPS = 1$ for stationary solutions of ODEs with detection of Hopf bifurcations, $IPS = 2$ for computation of periodic solutions (starting data can be a Hopf bifurcation), and $IPS = 4$ for boundary value problem are the most used. Many more problem types are considered.

Output control

- *NPR*: This constant is used to write *fort.8* plotting and restarting data every *NPR* steps.
- *IBR*: This constant specifies the initial branch number *BR* that is used. If $IBR = 0$ then it is determined automatically.
- *LAB*: This constant specifies the initial label number *LAB* that is used.
- *UZR*: This constant allows the setting of parameter values at which labeled plotting and restart information is to be written in *fort.8* output-file. It also allows the computation to terminate at such point (with a $-$ before the value of the variable).
- *UZSTOP*: To terminate the computation if any solution point that is specified is encountered at a given parameter (works the same way that *UZR* with $-$ before the values).

Many constants don't appear in this document for the sake of brevity, and the ones that appear aren't fully described, for a more detailed description of the many options that AUTO offers the manual [16] is highly recommended.

4.3 Output files

When an AUTO program is run it writes a standard output in the same window, where only special labelled solutions are shown. Some of this solutions types are in the figure (4.3). The numbers corresponding to each type are used internally in the *fort.7* and *fort.8* output-files described bellow.

The *fort.7* output-file contains the bifurcation diagram, its format is the same as the standard output but it is more extensive, as every solution has a line printed here.

The *fort.8* output-file contains graphic and restart data for labelled solutions. Here the information is more extensive than that in *fort.7*.

The *fort.9* is for diagnostic messages, convergence history, eigenvalues and **Floquet multipliers** of periodic solutions. The amount of diagnostic data can be controlled using the *IID* AUTO constant.

4.4 Using the Python commands

There are several user interfaces to use AUTO, some of them are graphical (GUI94, PLAUT04), and others are code based. For this work, the code based Python user interface has been chosen for its versatility and because it is relatively easy to use (compared with the others code based user interface, as the Unix commands or the PLAUT-commands). It also allows the user to create *scripts* with the instructions to perform a series of runs, which will come very handy.

In order to use the AUTO command line user interface (CLUI), which is the object oriented Python based command language for AUTO, the user has to call for it in the Unix shell by typing *auto* in the directory where the equations file is contained (for a new problem call the CLUI in an empty directory). If successfully executed, the prompt will appear as *AUTO>*.

To begin with this brief CLUI tutorial lets start with how to use any of the demo problems that AUTO provides. First, by typing *mkdir newfolder* AUTO creates a folder called *newfolder*, then *cd newfolder* changes the directory to that folder. Note that those commands are similar to the Unix commands of the same function. Now by typing *copydemo(’ab’)* AUTO copies the demo files of the problem *ab*, in which the equations that model an $A \rightarrow B$ reaction are implemented. By typing *ls* the copied files are shown in the terminal. There, the *ab.f90* file and a series of constant files are present, in addition to other file types that will be explained later. It is possible to see the content of those files in the terminal by typing *cat filename*, where *filename* includes the termination, although a text editor is recommended for this.

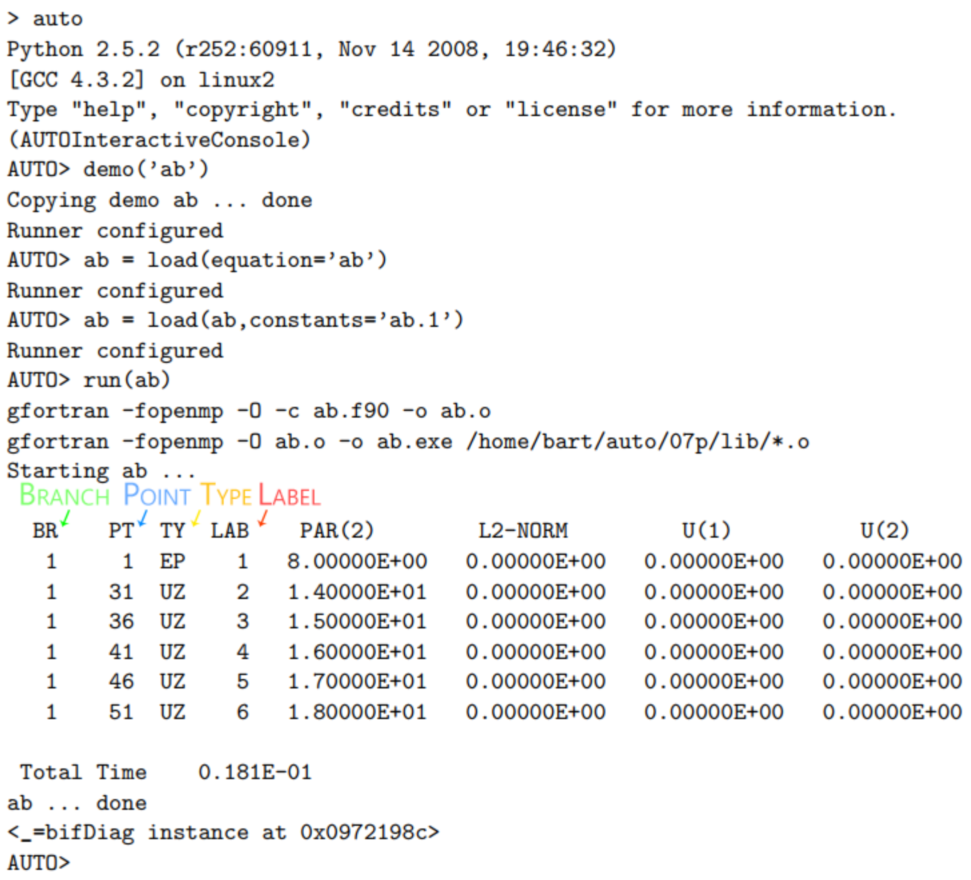
Now lets proceed on how to run a problem in general, given the equations file and the constants file. One possible method is to type the following commands one after the other in the CLUI:

```
ab = load(equation=’ab’)
ab = load(ab, constants=’ab.1’)
run(ab)
```

The first command creates a Python object named *ab* with the equations file of the problem associated to it. The capacity to use variables in this way is something that differentiates the CLUI form the other user interfaces. The second command adds the constants file that will be used in this run, which in this case is the *c.ab.1* file. Note that the command used is the same. This two commands could be replaced with a sole *ab = load(equation=’ab’, constants=’ab.1’)*, which creates the very same object. Finally the command *run(ab)* executes the problem that has been defined in the object *ab*. Executing these three commands results in what is shown in figure (4.2). The *run* command returns a *bifurcation diagram structure*. In this case we can see that there is only one branch in the solution family (*BR* is always 1), and that all output solutions are user requested except the first one, which is the starting point given in the *SPTNT* routine. The point types are in the figure (4.3). Lastly, the problem was of the type *IPS = 1*, and so the given solutions are the stationary solutions of the ODE system for the continuation of a certain parameter. This two commands, *load* and *run*, are the most common, but if the intermediate variable *ab* is not needed then all of it can be done with *abSolution = run(e=’ab’, c=’ab.1’)*, where the resulting bifurcation diagram object has been stored in a variable for later use, and the reserved words *equation* and *constants* have been contracted. The result is the same that in the figure (4.2).

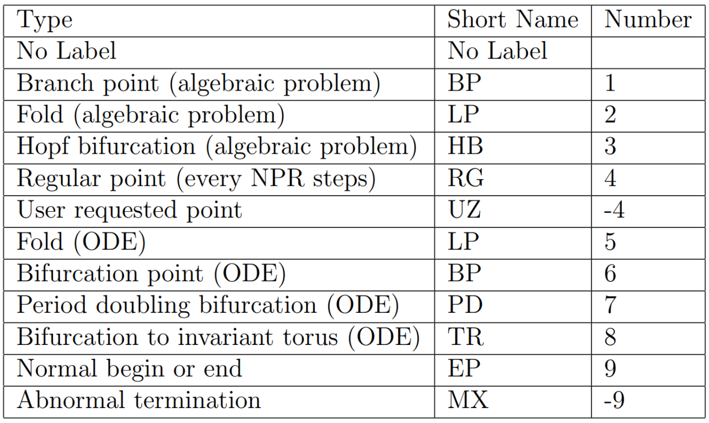
The next important feature of the CLUI is the *scripting* capability, which is now explained. It is basically a feature that allows the user to place all the commands that need to be executed in an *.auto* file, and then executing that file for those commands to be run all at once. Consider the example of a file named *abDemo.auto* with the following code:

```
copydemo(’ab’)
ab = load(equation=’ab’)
ab = load(ab, constants=’ab.1’)
run(ab)
```



```
> auto
Python 2.5.2 (r252:60911, Nov 14 2008, 19:46:32)
[GCC 4.3.2] on linux2
Type "help", "copyright", "credits" or "license" for more information.
(AUTOInteractiveConsole)
AUTO> demo('ab')
Copying demo ab ... done
Runner configured
AUTO> ab = load(equation='ab')
Runner configured
AUTO> ab = load(ab,constants='ab.1')
Runner configured
AUTO> run(ab)
gfortran -fopenmp -O -c ab.f90 -o ab.o
gfortran -fopenmp -O ab.o -o ab.exe /home/bart/auto/07p/lib/*.o
Starting ab ...

  BR    PT  TY  LAB   PAR(2)        L2-NORM        U(1)           U(2)
   1     1  EP    1  8.00000E+00   0.00000E+00   0.00000E+00   0.00000E+00
   1    31  UZ    2  1.40000E+01   0.00000E+00   0.00000E+00   0.00000E+00
   1    36  UZ    3  1.50000E+01   0.00000E+00   0.00000E+00   0.00000E+00
   1    41  UZ    4  1.60000E+01   0.00000E+00   0.00000E+00   0.00000E+00
   1    46  UZ    5  1.70000E+01   0.00000E+00   0.00000E+00   0.00000E+00
   1    51  UZ    6  1.80000E+01   0.00000E+00   0.00000E+00   0.00000E+00

 Total Time    0.181E-01
ab ... done
<_=bifDiag instance at 0x0972198c>
AUTO>
```

Figure 4.2 Result of runing demo program with *constans='ab.1'*. Image from [16].



Type	Short Name	Number
No Label	No Label	
Branch point (algebraic problem)	BP	1
Fold (algebraic problem)	LP	2
Hopf bifurcation (algebraic problem)	HB	3
Regular point (every NPR steps)	RG	4
User requested point	UZ	-4
Fold (ODE)	LP	5
Bifurcation point (ODE)	BP	6
Period doubling bifurcation (ODE)	PD	7
Bifurcation to invariant torus (ODE)	TR	8
Normal begin or end	EP	9
Abnormal termination	MX	-9

Figure 4.3 Point types in a bifurcation diagram structure. Image from [16].

Then, by typing *auto abDemo.auto* in the Unix shell AUTO will execute the code in it as if it where in the CLUI. The same result would be obtained by typing *auto('abDemo.auto')* in the CLUI itself.

As a final comment, the file *autorc* contains commands to control the behaviour of the *PyPlaut* plotting options (some of them will be used), and the *plaut04.rc* controls the PLAUT04 user interface, which won't be of use in this work.

4.4.1 Other commands

- *save*: Save data files. By typing *save(x,'xxx')* AUTO saves the bifurcation diagram *x* to the files *b.xxx*, *s.xxx* and *d.xxx*. The file *b.xxx* contains the bifurcation diagram, *s.xxx* contains the solution and *d.xxx* contains the diagnostics. If *x* is a solution or does not contain any bifurcation diagram or diagnostic data, then only the file *s.xxx* is saved to.

- *append*: Append data files. By typing *append(x,'xxx')* AUTO appends the bifurcation diagram *x* to existing data files *b.xxx*, *s.xxx* and *d.xxx*.
- *plot*: Plotting data. *plot(x)* runs the graphic program *PyPlaut* for the graphical inspection of bifurcation diagram or solution data in *x*.
- *clean*: Clean the current directory of *fort.* and *.exe* files.
- *floquet*: Print the Floquet multipliers. *floquet(x)* lists the Floquet multipliers in the diagnostics of the bifurcation diagram object *x*, *floquet()* lists the content of the *fort.9* file and *floquet('xxx')* lists the Floquet multipliers in the file *d.xxx*.

4.5 Problem example

The AUTO manual has many examples for the user to grasp the software potential. In this work we are interested mainly in the way periodic solutions are calculated, and so a first example of the method we will be shown in this section. This way, the later procedure to compute the cycles in the CR3BP will be very straightforward to understand, though the computation of the manifolds will be explained on the fly.

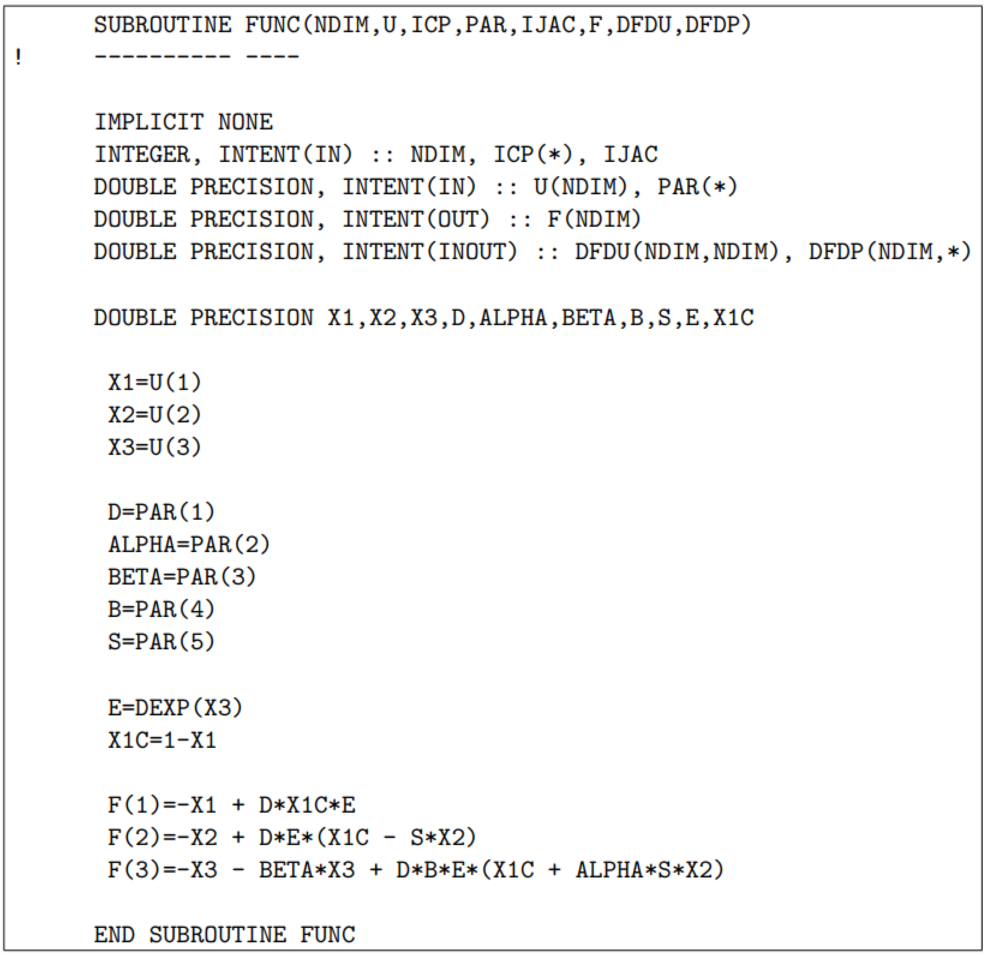
4.5.1 The $A \rightarrow B \rightarrow C$ reaction

The dynamic of the reaction is given by the system:

$$\begin{aligned} u_1' &= -u_1 + p_1(1-u_1)e^{u_3}, \\ u_2' &= -u_2 + p_1 e^{u_3}(1-u_1-p_5u_2), \\ u_3' &= -u_3 + p_3u_3 + p_1p_4r^{u_3}(1-u_1+p_2p_5u_2), \end{aligned} \tag{4.1}$$

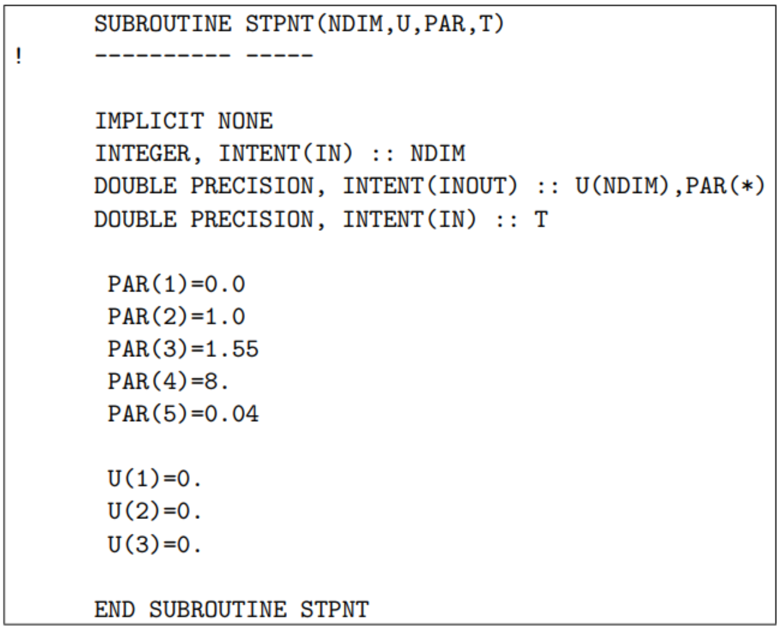
where $p_2 = 1$, $p_3 = 1.55$, $p_4 = 8$ and $p_5 = 0.05$. The free parameter is then p_1. The implementation of the equations in the *FUNC* subroutine can be seen in the figure (4.4), this scheme is valid for the implementation of mostly any problem, aside from the name of the parameters and the number of equations. The *STPNT* subroutine is shown in figure (4.5), where an equilibrium of the equations for $p_1 = 0$ has been implemented as the starting point. Notice that the rest of the routines haven't been completed as they don't serve any purpose in this problem

The first step to compute cycles with AUTO is to detect stationary solutions from which they might emerge. For this the *c.abc.1* constants file, in figure (4.6), is to be used. The most important things to notice in here are that $IPS = 1$, so the problem is defined to compute a family of stationary solutions, $IRS = 0$, so that the starting point is the one defined in *STPNT*, $ICP = [1]$, so that the free parameter is p_1, and finally that $UZR = \{1 : 0.4\}$ with $UZSTOP = ['UZ1']$, which states that the computation ends once the free parameter $p_1 = 0.4$. With all of this, by executing *abc = run(e='abc', c='abc.1')* the screen output is the one in figure (4.7). There we can see that aside from the starting point and the user requested one, there are 4 special solutions in between with the type *HB*, which corresponds to **Hopf bifurcation** points in the bifurcation diagram. This means that some eigenvalues of the Jacobian cross the imaginary axis at those points, which is the same as saying that those points have eigenvalues with zero real part, and that *very close* to them analytical periodic solutions do exist. This is the main foundation the computation of cycles in the AUTO environment relies on. With *plot(abc, stability=True, grid=True)* the bifurcation diagram can be represented using *PyPlaut*, this is in figures (4.8) and (4.9). Figure (4.8) represents the evolution of the *L2-Norm* of the stationary solution ($L2_{NORM} = \sqrt{u_1^2+u_2^2+u_3^2}$) with the change of p_1. Notice that the change between solid and dashed line represents the change in stability of those solutions (solid lines are stable solutions). Figure (4.9) on the other hand is the representation of those solutions in the phase space. Using the command *eigenvalues(abc)* it is possible to see that, out of the three eigenvalues of the system, there is a pair of complex conjugate eigenvalues whose real part change sign at those exact points, which indicates that a sole family of periodic solutions emanate from the *HP* as a linear combination of the corresponding eigenvectors and $e^{\lambda t}$.



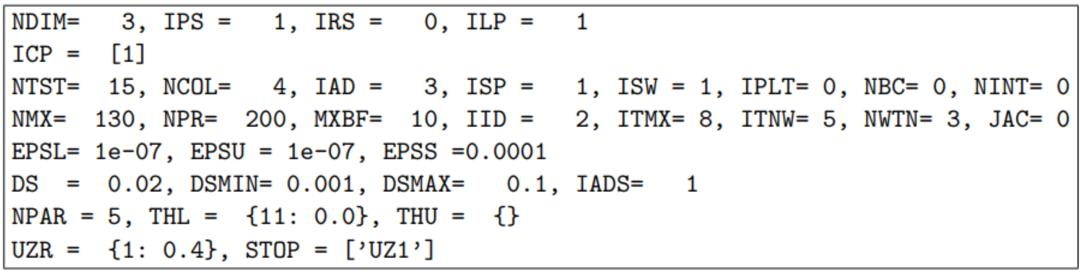
```
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
!     ---------- ----

      IMPLICIT NONE
      INTEGER, INTENT(IN) :: NDIM, ICP(*), IJAC
      DOUBLE PRECISION, INTENT(IN) :: U(NDIM), PAR(*)
      DOUBLE PRECISION, INTENT(OUT) :: F(NDIM)
      DOUBLE PRECISION, INTENT(INOUT) :: DFDU(NDIM,NDIM), DFDP(NDIM,*)

      DOUBLE PRECISION X1,X2,X3,D,ALPHA,BETA,B,S,E,X1C

       X1=U(1)
       X2=U(2)
       X3=U(3)

       D=PAR(1)
       ALPHA=PAR(2)
       BETA=PAR(3)
       B=PAR(4)
       S=PAR(5)

       E=DEXP(X3)
       X1C=1-X1

       F(1)=-X1 + D*X1C*E
       F(2)=-X2 + D*E*(X1C - S*X2)
       F(3)=-X3 - BETA*X3 + D*B*E*(X1C + ALPHA*S*X2)

      END SUBROUTINE FUNC
```

Figure 4.4 *FUNC* subroutine in the *abc* demo problem. Image from [16].



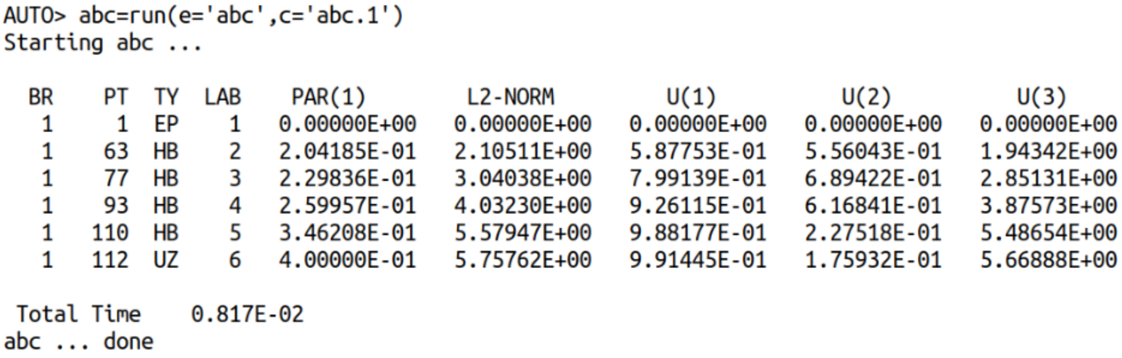
```
      SUBROUTINE STPNT(NDIM,U,PAR,T)
!     ---------- -----

      IMPLICIT NONE
      INTEGER, INTENT(IN) :: NDIM
      DOUBLE PRECISION, INTENT(INOUT) :: U(NDIM),PAR(*)
      DOUBLE PRECISION, INTENT(IN) :: T

       PAR(1)=0.0
       PAR(2)=1.0
       PAR(3)=1.55
       PAR(4)=8.
       PAR(5)=0.04

       U(1)=0.
       U(2)=0.
       U(3)=0.

      END SUBROUTINE STPNT
```

Figure 4.5 *STPNT* subroutine in the *abc* demo problem. Image from [16].



```
NDIM=   3, IPS =   1, IRS =   0, ILP =   1
ICP =  [1]
NTST=  15, NCOL=   4, IAD =   3, ISP =   1, ISW = 1, IPLT= 0, NBC= 0, NINT= 0
NMX=  130, NPR=  200, MXBF=  10, IID =   2, ITMX= 8, ITNW= 5, NWTN= 3, JAC= 0
EPSL= 1e-07, EPSU = 1e-07, EPSS =0.0001
DS  =  0.02, DSMIN= 0.001, DSMAX=   0.1, IADS=   1
NPAR = 5, THL =  {11: 0.0}, THU =  {}
UZR =  {1: 0.4}, STOP = ['UZ1']
```

Figure 4.6 Content of the constants file for the first run of the problem. Image from [16].



```
AUTO> abc=run(e='abc',c='abc.1')
Starting abc ...

  BR    PT  TY  LAB    PAR(1)        L2-NORM         U(1)          U(2)          U(3)
   1     1  EP    1   0.00000E+00   0.00000E+00   0.00000E+00   0.00000E+00   0.00000E+00
   1    63  HB    2   2.04185E-01   2.10511E+00   5.87753E-01   5.56043E-01   1.94342E+00
   1    77  HB    3   2.29836E-01   3.04038E+00   7.99139E-01   6.89422E-01   2.85131E+00
   1    93  HB    4   2.59957E-01   4.03230E+00   9.26115E-01   6.16841E-01   3.87573E+00
   1   110  HB    5   3.46208E-01   5.57947E+00   9.88177E-01   2.27518E-01   5.48654E+00
   1   112  UZ    6   4.00000E-01   5.75762E+00   9.91445E-01   1.75932E-01   5.66888E+00

 Total Time    0.817E-02
abc ... done
```

Figure 4.7 Result of running stationary solution family search in the $A \to B \to C$ problem.

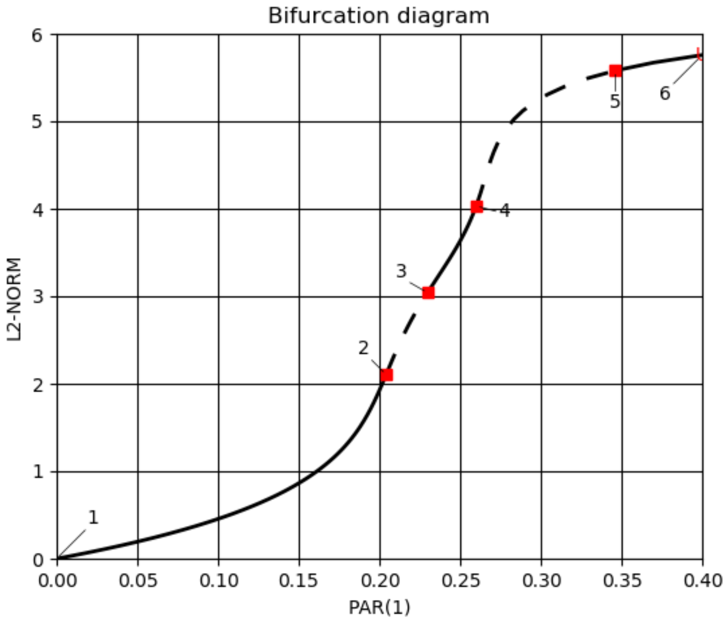


Figure 4.8 *L2-NORM* of the stationary solution family in $A \to B \to C$ problem.

The next run use the *c.abc.2* constants file, in figure (4.10), where $IPS = 2$, as now the problem is a search of a family of periodic solutions, $ICP = [1\,,\,11]$, as now the period is also a free parameter and $UZR = \{1 : 0.25\}$ so the search of each solution family ends when $p_1 = 0.25$. For this run the next python script will be used:

```
abc = run(e=’abc’, c=’abc.1’)
for solution in abc(’HB’)
    abc = abc + run(solution, c=’abc.2’)
```

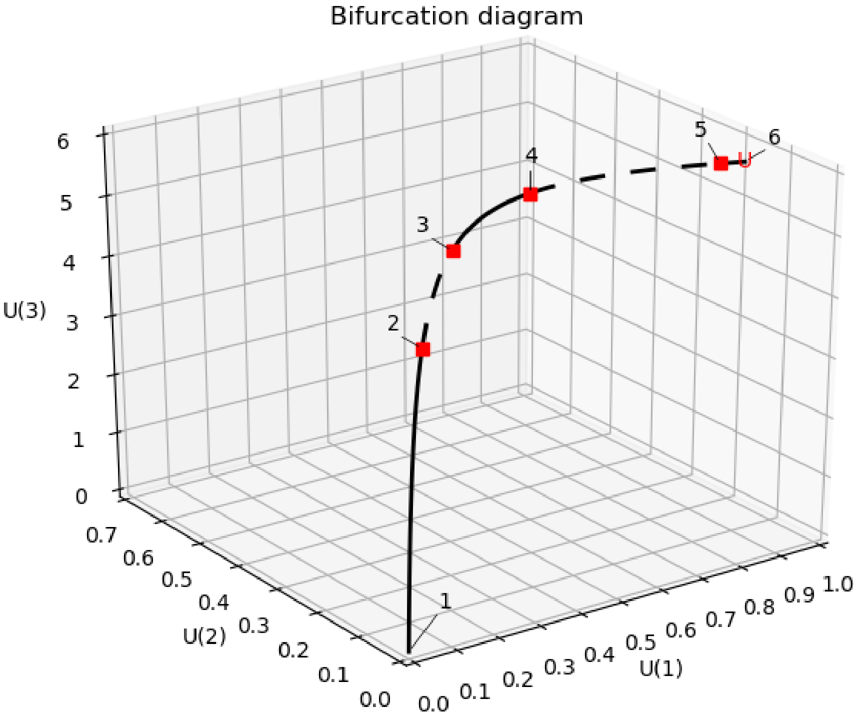
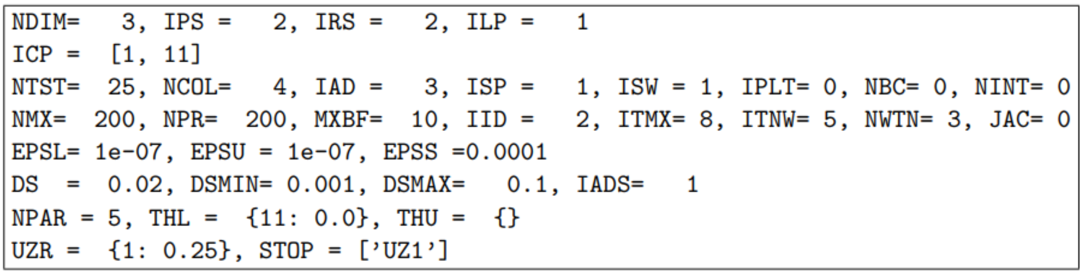


Figure 4.9 Stationary solution family of $A \rightarrow B \rightarrow C$ problem in the phase space.



```
NDIM=   3, IPS =   2, IRS =   2, ILP =   1
ICP =  [1, 11]
NTST=  25, NCOL=   4, IAD =   3, ISP =   1, ISW = 1, IPLT= 0, NBC= 0, NINT= 0
NMX=  200, NPR=  200, MXBF=  10, IID =   2, ITMX= 8, ITNW= 5, NWTN= 3, JAC= 0
EPSL= 1e-07, EPSU = 1e-07, EPSS =0.0001
DS  =  0.02, DSMIN= 0.001, DSMAX=   0.1, IADS=   1
NPAR = 5, THL =  {11: 0.0}, THU =  {}
UZR =  {1: 0.25}, STOP = ['UZ1']
```

Figure 4.10 Content of the constants file for the second to last run of the problem. Image from [16].

The *abc('HB')* is a list of the Hopf bifurcation solutions in the python object *abc*, so the *for* loop goes through all those solutions. Inside the loop the line *abc* = *abc* + *run(solution, c='abc.2')* executes a run in which the *solution* acts as initial point (the Hopf bifurcations) from which the periodic solutions are continued, and then concatenates the result with *abc*. The results of this calculations can be seen in the bifurcation diagram of figure (4.11) and the solutions in (4.12). In this case the *L2-Norm* of the periodic solution families corresponds to the maximum norm of the states through the orbits themselves. The orbits marked as L in (4.11) correspond to folds. In figure (4.13) the period of the solution families are represented, notice that even at the start of each of them the period isn't zero. Finally, as to clarify how the periodic solutions emerge form the Hopf bifurcations, a detail of this is represented in figure (4.14), where the family of periodic solutions that comes from $HB1$ is pictured from the start.

This whole problem is a good example of the method that will follow, and will help to clarify some of the details involved. The CR3BP is studied now with AUTO.

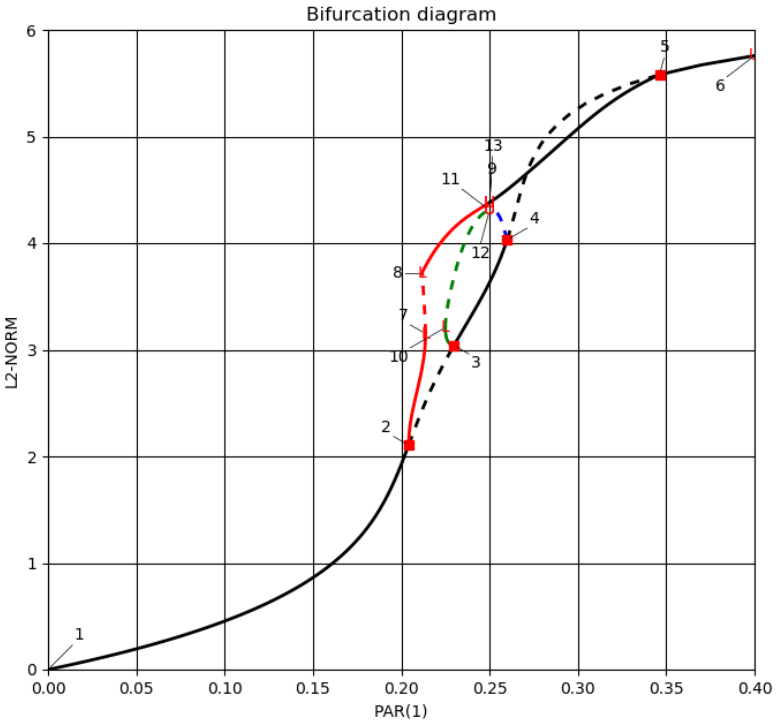


Figure 4.11 *L2-NORM* of the periodic solution families in $A \to B \to C$ problem.

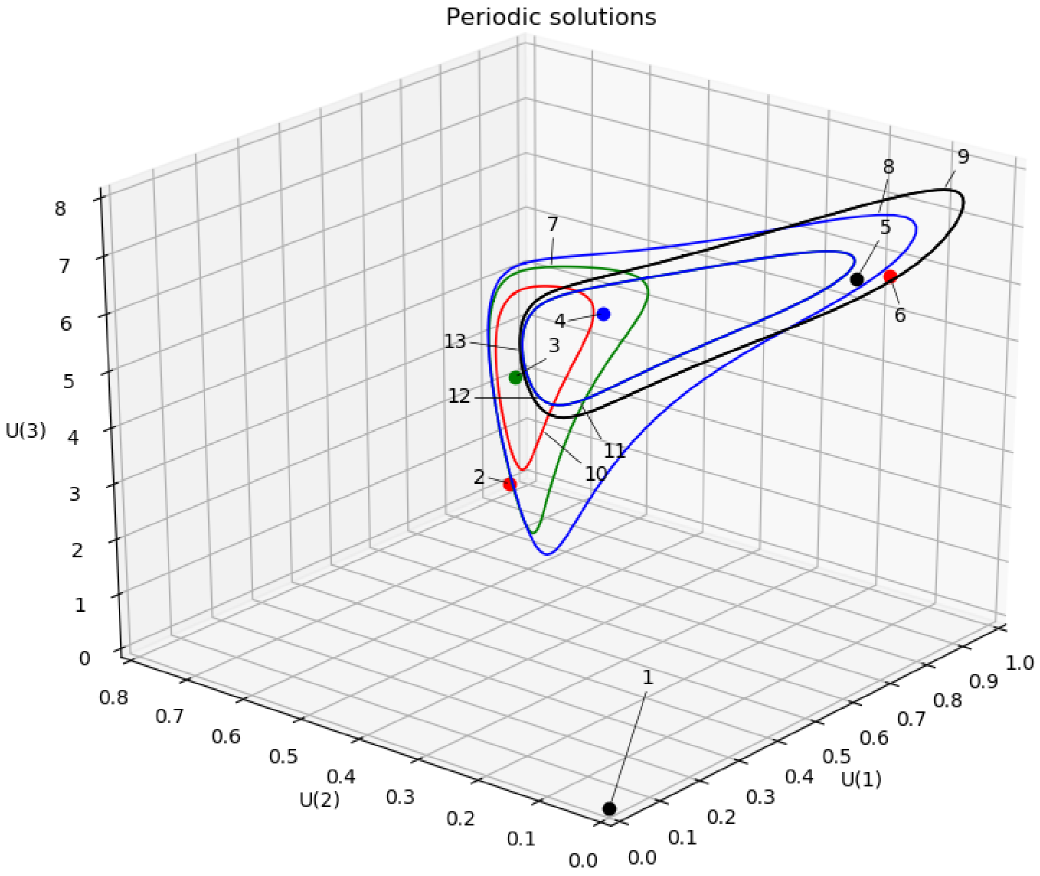


Figure 4.12 Some of the periodic solutions in $A \to B \to C$ problem.

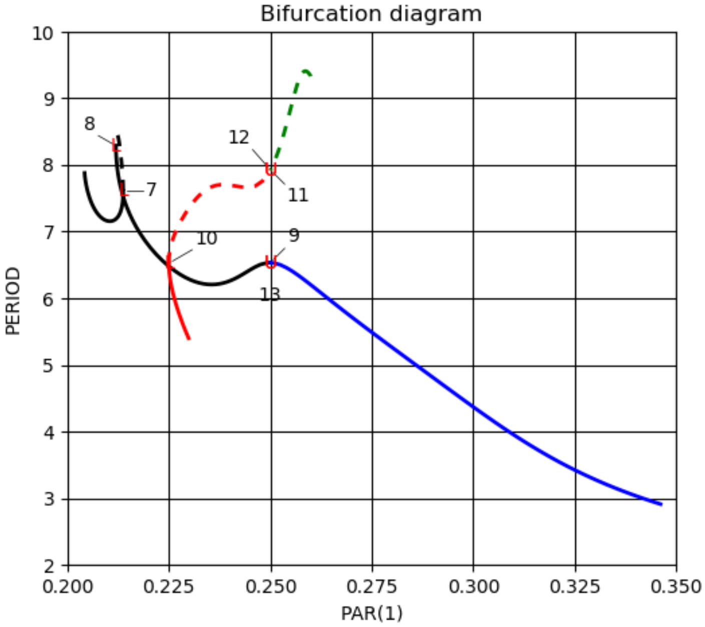


Figure 4.13 Period of the periodic solution families in $A \to B \to C$ problem.

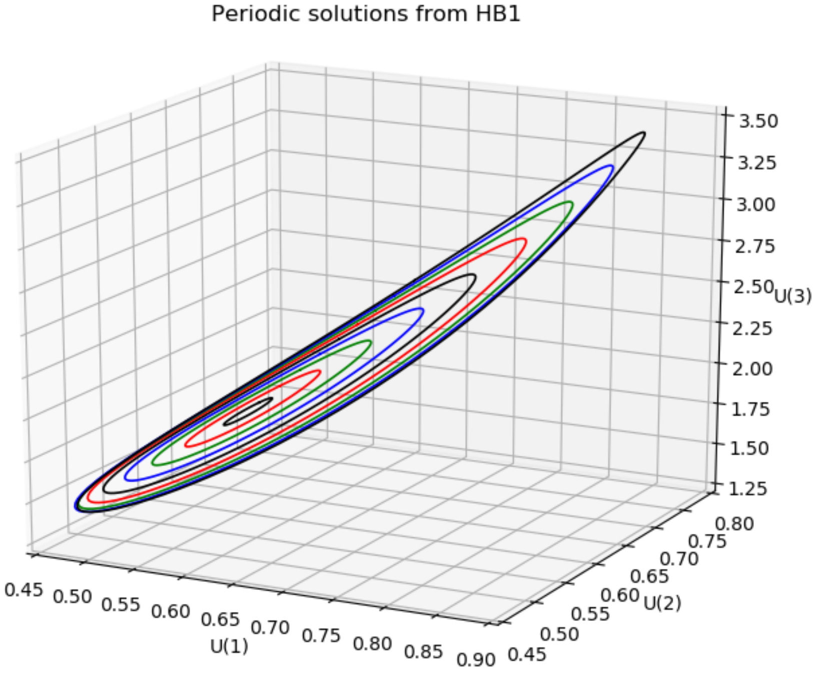


Figure 4.14 Detail of the $HB1$ periodic solution family as it emerges from its Hopf bifurcation. The parameter p_1 changes from 0.204 (in the HB) to 0.207.

5 AUTO-based numerical study of the CR3BP

In this chapter the tool AUTO is used to study the types of periodic orbits around the Earth-Moon system as well as the invariant manifolds associated.

5.1 Problem definition

As stated in the introduction, the CR3BP describes the motion of a negligible mass in three dimensional space as an effect of the gravitational attraction of two heavy bodies which orbit their common center of mass in a perfect circular motion. The scheme of this situation for a given position of the mass under study is in figure (5.1), where the parameter μ represents the ratio of the mass of the smaller primary to the total mass of the system ($\mu = m_2/(m_1 + m_2)$). The problem will be studied in the non-dimensional form, so that the distance between the two main bodies equals to 1 at all times (the time is also adimensionalized with de angular velocity).

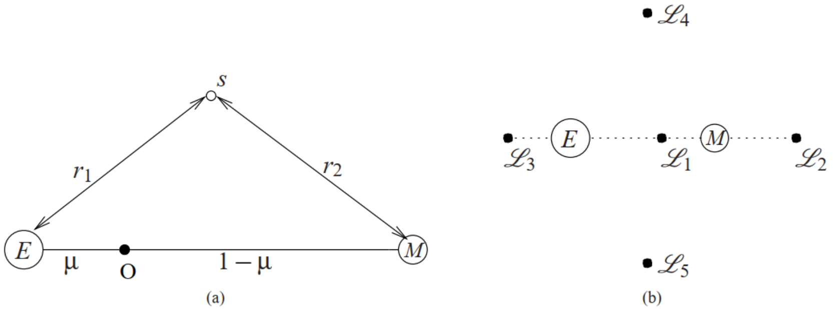


Figure 5.1 CR3BP scheme (a) and nomenclature of the five libration points (b). Image from [15].

Given the frame of reference centred in the barycentre of the two massive bodies that rotates with the same speed as them (a constant of the problem), the adimensionalized equations of motion are well known [2]:

$$
\begin{aligned}
x'' &= 2y' + x - (1-\mu)(x+\mu)r_1^{-3} - \mu(x-1+\mu)r_2^{-3},\\
y'' &= -2x' + y - (1-\mu)yr_1^{-3} - \mu y r_2^{-3},\\
z'' &= -(1-\mu)zr_1^{-3} - \mu z r_2^{-3},
\end{aligned}
\tag{5.1}
$$

where

$$
\begin{aligned}
r_1 &= \sqrt{(x+\mu)^2+y^2+z^3}, \\
r_2 &= \sqrt{(x-1+\mu)^2+y^2+z^{”}}.
\end{aligned}
\tag{5.2}
$$

The equation system (5.1) has one integral of motion, specifically the energy or *Jacobi constant*:

$$
E = \frac{x'^2+y'^2+z'^2}{2} - U(x,y,z) - \frac{\mu(1-\mu)}{2}, \tag{5.3}
$$

where $U(x,y,z)$ is the potential energy, combination of gravity and the rotation of the frame of reference:

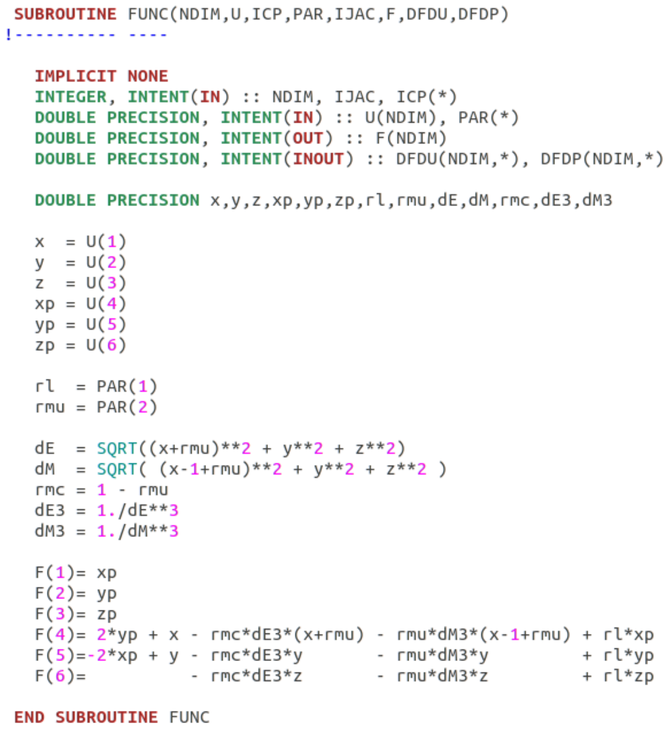
$$
U(x,y,z) = \frac{1}{2}(x^2+y^2) + \frac{1-\mu}{r_1} + \frac{\mu}{r_2}.
$$

As we are most interested in the case of the Earth-Moon system, throughout much of this document we will consider $\mu = 0.01215$.

5.2 Setting up the AUTO problem

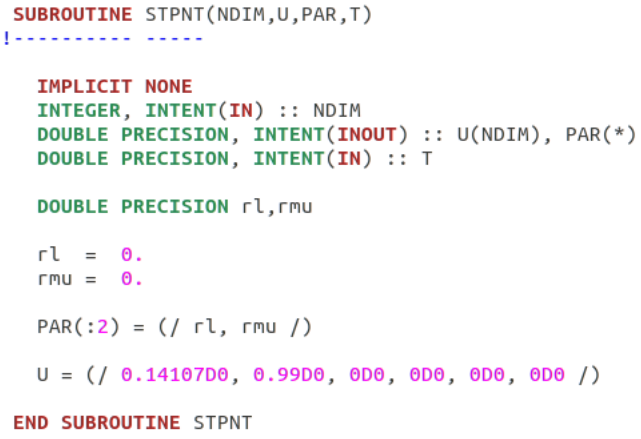
The *.f90* file that will be use in general is the corresponding to the AUTO demo *r3b*, which is explained now. The *FUNC* subroutine is in figure (5.2), where the equations of motion are coded in free-form Fortran. Notice that there are two explicit parameters in the equations, one is μ, and the other is multiplying the velocities in the acceleration equations ($F(4)$, $F(5)$ and $F(6)$ in the code), which is what we called unfolding parameter in section (3.4.1). That last parameter will be the one used to continue the periodic orbits.

Next, the *STPTN* subroutine is in figure (5.3). Here the state variables and the parameters are set up for the first run that will be executed, which will be done to compute the stationary solutions with μ. Notice that for the selected value of μ the chosen state is that in which the position is on the unit circle in the *xy* plane and the velocities are zero. That is the degenerate case in which there is no secondary body in the system, where any position in the unit circle is a stationary solution of (5.1).



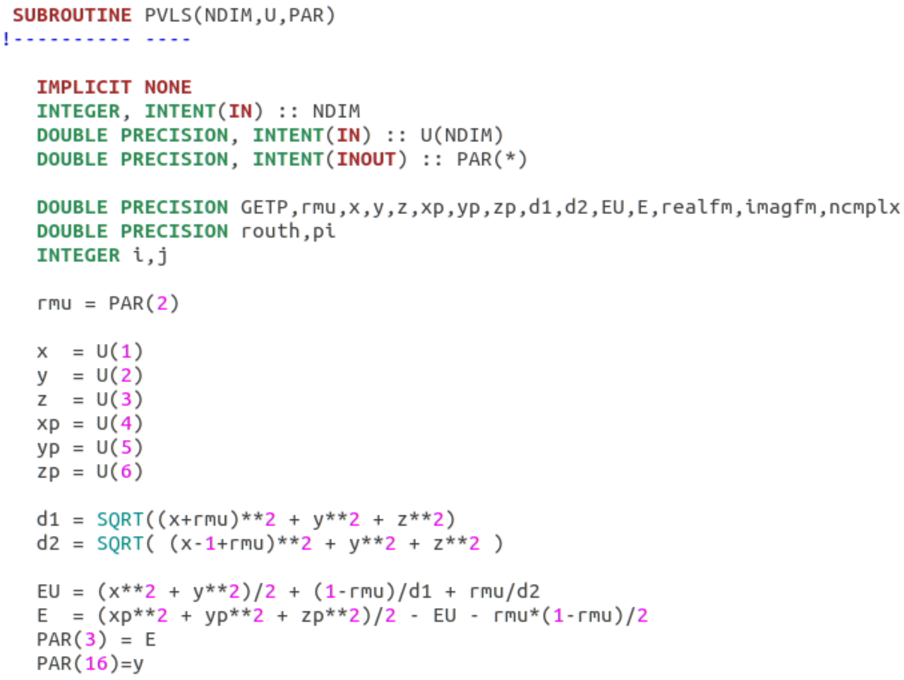
```fortran
SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)
!---------- ----

  IMPLICIT NONE
  INTEGER, INTENT(IN) :: NDIM, IJAC, ICP(*)
  DOUBLE PRECISION, INTENT(IN) :: U(NDIM), PAR(*)
  DOUBLE PRECISION, INTENT(OUT) :: F(NDIM)
  DOUBLE PRECISION, INTENT(INOUT) :: DFDU(NDIM,*), DFDP(NDIM,*)

  DOUBLE PRECISION x,y,z,xp,yp,zp,rl,rmu,dE,dM,rmc,dE3,dM3

  x  = U(1)
  y  = U(2)
  z  = U(3)
  xp = U(4)
  yp = U(5)
  zp = U(6)

  rl  = PAR(1)
  rmu = PAR(2)

  dE  = SQRT((x+rmu)**2 + y**2 + z**2)
  dM  = SQRT( (x-1+rmu)**2 + y**2 + z**2 )
  rmc = 1 - rmu
  dE3 = 1./dE**3
  dM3 = 1./dM**3

  F(1)= xp
  F(2)= yp
  F(3)= zp
  F(4)= 2*yp + x - rmc*dE3*(x+rmu) - rmu*dM3*(x-1+rmu) + rl*xp
  F(5)=-2*xp + y - rmc*dE3*y        - rmu*dM3*y         + rl*yp
  F(6)=          - rmc*dE3*z        - rmu*dM3*z         + rl*zp

END SUBROUTINE FUNC
```

Figure 5.2 *FUNC* subroutine in the CR3BP.



```
 SUBROUTINE STPNT(NDIM,U,PAR,T)
!---------- -----

   IMPLICIT NONE
   INTEGER, INTENT(IN) :: NDIM
   DOUBLE PRECISION, INTENT(INOUT) :: U(NDIM), PAR(*)
   DOUBLE PRECISION, INTENT(IN) :: T

   DOUBLE PRECISION rl,rmu

   rl  =  0.
   rmu =  0.

   PAR(:2) = (/ rl, rmu /)

   U = (/ 0.14107D0, 0.99D0, 0D0, 0D0, 0D0, 0D0 /)

 END SUBROUTINE STPNT
```

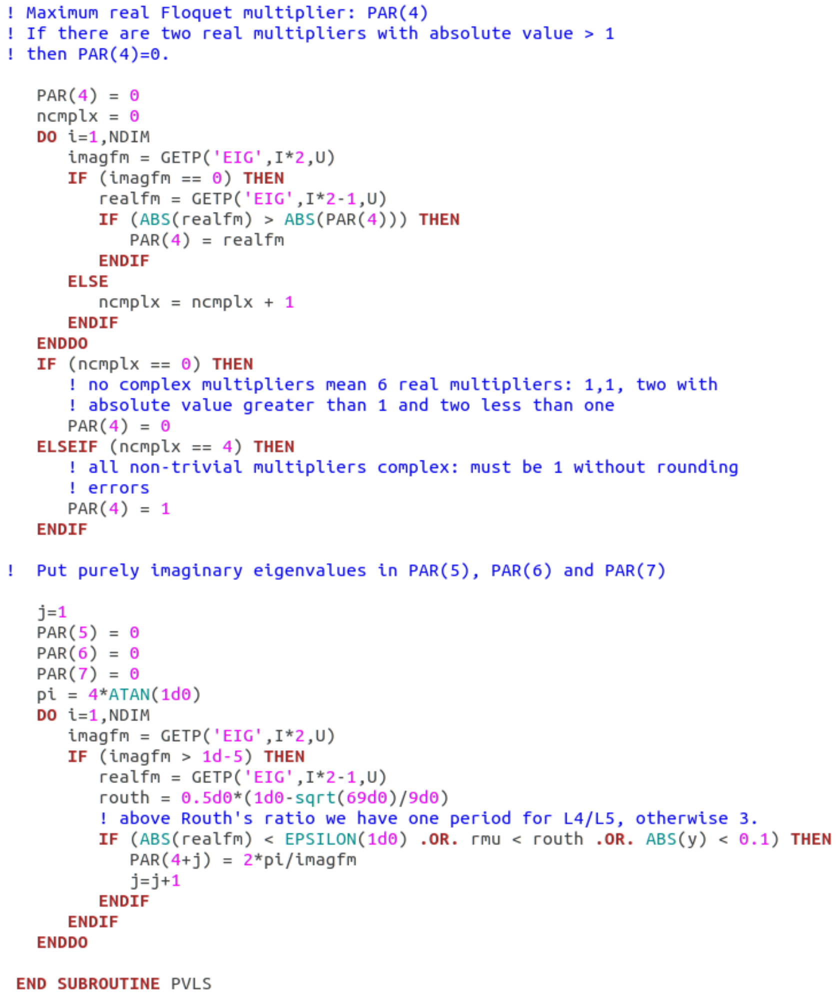
Figure 5.3 *STPNT* subroutine in the CR3BP.



```
 SUBROUTINE PVLS(NDIM,U,PAR)
!---------- ----

   IMPLICIT NONE
   INTEGER, INTENT(IN) :: NDIM
   DOUBLE PRECISION, INTENT(IN) :: U(NDIM)
   DOUBLE PRECISION, INTENT(INOUT) :: PAR(*)

   DOUBLE PRECISION GETP,rmu,x,y,z,xp,yp,zp,d1,d2,EU,E,realfm,imagfm,ncmplx
   DOUBLE PRECISION routh,pi
   INTEGER i,j

   rmu = PAR(2)

   x  = U(1)
   y  = U(2)
   z  = U(3)
   xp = U(4)
   yp = U(5)
   zp = U(6)

   d1 = SQRT((x+rmu)**2 + y**2 + z**2)
   d2 = SQRT( (x-1+rmu)**2 + y**2 + z**2 )

   EU = (x**2 + y**2)/2 + (1-rmu)/d1 + rmu/d2
   E  = (xp**2 + yp**2 + zp**2)/2 - EU - rmu*(1-rmu)/2
   PAR(3) = E
   PAR(16)=y
```

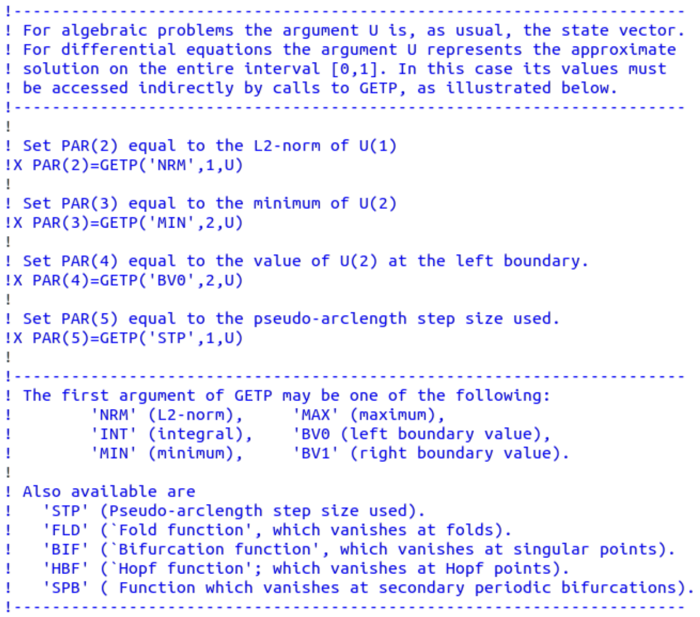
Figure 5.4 *PVLS* subroutine in the CR3BP, part 1.

Finally, the *PVLS* is in figures (5.4) and (5.5). The first part calculates the energy of the solution state U that comes as an input using equation (5.3). Notice that two parameters are created, one stores the energy of the solution and the other the value of the position in y. The $PAR(16)$ will be a solution measure. The second part of the routine has two distinguishable functions, one is to calculate the maximum real Floquet multiplier and to store it in $PAR(4)$, this will be useful in discerning the stability of the periodic orbits. If $PAR(4) = 0$ then it means that two Floquet multipliers are outside of the unit circle, so no stability can be expected from that solution. To achieve this it is necessary to use the function *GETP*, a more detailed description of its usage is in figure (5.6). The second function retrieves the imaginary parts of the eigenvalues which only have imaginary part, and stores the corresponding period ($2\pi/w$, where w is the imaginary part) in $PAR(5)$, $PAR(6)$ and $PAR(7)$. This will only be used in the stationary solution analysis at the beginning. For this equations file no other subroutines will be used.



```
! Maximum real Floquet multiplier: PAR(4)
! If there are two real multipliers with absolute value > 1
! then PAR(4)=0.

   PAR(4) = 0
   ncmplx = 0
   DO i=1,NDIM
      imagfm = GETP('EIG',I*2,U)
      IF (imagfm == 0) THEN
         realfm = GETP('EIG',I*2-1,U)
         IF (ABS(realfm) > ABS(PAR(4))) THEN
            PAR(4) = realfm
         ENDIF
      ELSE
         ncmplx = ncmplx + 1
      ENDIF
   ENDDO
   IF (ncmplx == 0) THEN
      ! no complex multipliers mean 6 real multipliers: 1,1, two with
      ! absolute value greater than 1 and two less than one
      PAR(4) = 0
   ELSEIF (ncmplx == 4) THEN
      ! all non-trivial multipliers complex: must be 1 without rounding
      ! errors
      PAR(4) = 1
   ENDIF

!  Put purely imaginary eigenvalues in PAR(5), PAR(6) and PAR(7)

   j=1
   PAR(5) = 0
   PAR(6) = 0
   PAR(7) = 0
   pi = 4*ATAN(1d0)
   DO i=1,NDIM
      imagfm = GETP('EIG',I*2,U)
      IF (imagfm > 1d-5) THEN
         realfm = GETP('EIG',I*2-1,U)
         routh = 0.5d0*(1d0-sqrt(69d0)/9d0)
         ! above Routh's ratio we have one period for L4/L5, otherwise 3.
         IF (ABS(realfm) < EPSILON(1d0) .OR. rmu < routh .OR. ABS(y) < 0.1) THEN
            PAR(4+j) = 2*pi/imagfm
            j=j+1
         ENDIF
      ENDIF
   ENDDO

 END SUBROUTINE PVLS
```

Figure 5.5 *PVLS* subroutine in the CR3BP, part 2.



```
!----------------------------------------------------------------------
! For algebraic problems the argument U is, as usual, the state vector.
! For differential equations the argument U represents the approximate
! solution on the entire interval [0,1]. In this case its values must
! be accessed indirectly by calls to GETP, as illustrated below.
!----------------------------------------------------------------------
!
! Set PAR(2) equal to the L2-norm of U(1)
!X PAR(2)=GETP('NRM',1,U)
!
! Set PAR(3) equal to the minimum of U(2)
!X PAR(3)=GETP('MIN',2,U)
!
! Set PAR(4) equal to the value of U(2) at the left boundary.
!X PAR(4)=GETP('BV0',2,U)
!
! Set PAR(5) equal to the pseudo-arclength step size used.
!X PAR(5)=GETP('STP',1,U)
!
!----------------------------------------------------------------------
! The first argument of GETP may be one of the following:
!        'NRM' (L2-norm),     'MAX' (maximum),
!        'INT' (integral),    'BV0 (left boundary value),
!        'MIN' (minimum),     'BV1' (right boundary value).
!
! Also available are
!   'STP' (Pseudo-arclength step size used).
!   'FLD' (`Fold function', which vanishes at folds).
!   'BIF' (`Bifurcation function', which vanishes at singular points).
!   'HBF' (`Hopf function'; which vanishes at Hopf points).
!   'SPB' ( Function which vanishes at secondary periodic bifurcations).
!----------------------------------------------------------------------
```

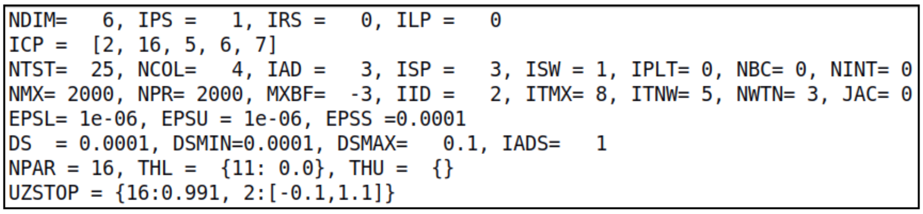
Figure 5.6 Description of GETP function extracted from *cusp* demo problem.

5.3 Stationary solution analysis

First, and as stated in the example of section (4.5), we need the stationary solutions from which the periodic orbits emerge. As it is well known, the CR3BP has 5 stationary solutions or libration points (figure (5.1) (b)). Three of the libration points, denoted L1, L2 and L3, are collinear with the primary bodies. Each of the other two libration points, L4 and L5, forms an equilateral triangle with the primaries.

The procedure will be to start at the stationary solution in the $STPNT$ subroutine and let AUTO search from there until it finds the desired points at $\mu = 0.01215$. To do that, the constants file in figure (5.7) is the one utilized. The most relevant constants values to notice from there are:

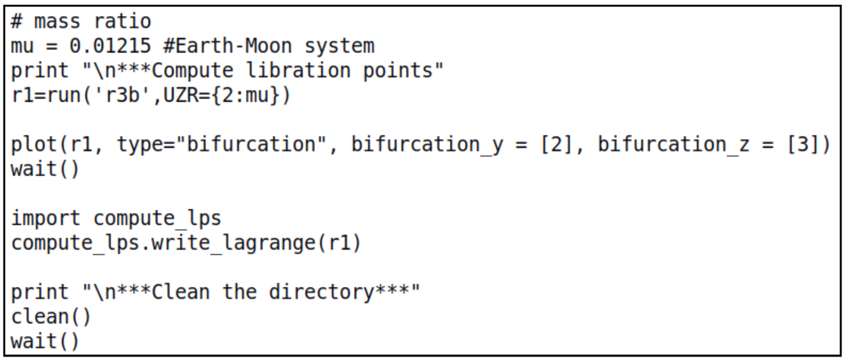
- $NDIM = 6$, the dimension of the problem is now 6 once expressed as an ODE.
- $IPS = 1$ and $IRS = 0$, as we are looking for stationary solutions starting from the point in $STPNT$.
- $ICP = [2,16,5,6,7]$, the main continuation parameter is $PAR(2)$ (μ in the code of *r3b.f90*), but $PAR(16)$, $PAR(5)$, $PAR(6)$ and $PAR(7)$ will be solution measures. By knowing $PAR(16)$ (the y coordinate of the point) we can know if it is a collinear or triangular solution, but more importantly, it allows to continue the solution in the unit circle when $\mu = 0$. On the other hand, knowing the periods associated will tell us the number and type of orbits that can emanate from them.
- $NPAR = 16$, this is necessary in order to use parameters until $PAR(16)$.
- $UZSTOP = \{16 : 0.991,\ 2 : [-0.1,1.1]\}$, the conditions for the continuation to stop in each branch.



```
NDIM=   6, IPS =   1, IRS =   0, ILP =   0
ICP =  [2, 16, 5, 6, 7]
NTST=  25, NCOL=   4, IAD =   3, ISP =   3, ISW = 1, IPLT= 0, NBC= 0, NINT= 0
NMX= 2000, NPR= 2000, MXBF=  -3, IID =   2, ITMX= 8, ITNW= 5, NWTN= 3, JAC= 0
EPSL= 1e-06, EPSU = 1e-06, EPSS =0.0001
DS  = 0.0001, DSMIN=0.0001, DSMAX=   0.1, IADS=   1
NPAR = 16, THL =  {11: 0.0}, THU =  {}
UZSTOP = {16:0.991, 2:[-0.1,1.1]}
```

Figure 5.7 Constants file used for the analysis of stationary solutions in the CR3BP.

Maybe more important than the constants that are in the file, the lack of any reference to the value of μ at which the output is needed for the problem to continue stands out. That will be solved when the program is run.



```
# mass ratio
mu = 0.01215 #Earth-Moon system
print "\n***Compute libration points"
r1=run('r3b',UZR={2:mu})

plot(r1, type="bifurcation", bifurcation_y = [2], bifurcation_z = [3])
wait()

import compute_lps
compute_lps.write_lagrange(r1)

print "\n***Clean the directory***"
clean()
wait()
```

Figure 5.8 Script that computes the libration points for the μ in the Earth-Moon system, as well as the number of periodic orbits that emanate from them thorough the periods.

Then, to execute the search, the AUTO script in figure (5.8) has been used. Let's focus first in the lines before the *import*. A variable with the value of μ for the Earth-Moon system is created, then a message is printed on the screen to indicate that the computation of the libration points begins. Then, by using *r1 = run('r3b', UZR={2:mu})*, AUTO automatically sets the problem with the files called *r3b* (equations and constants files), and adds the constant *UZR={2:mu}* to the problem, which is meant to force an output for the value of μ previously given. This few lines are enough to make a complete analysis of stationary solutions in the CR3BP with AUTO, and it is a very complete analysis as the figure (5.9) shows, the result of the plot command. The screen output is in figure (5.10), where we can see AUTO has unveil 4 branches in the solution families. The first branch is the one of the starting point, the unit circle for $\mu = 0$, as the values of $PAR(2)$ seems to indicate (zero to numerical precision). The 3 other branches are accessed through the branching points with labels 2, 3 and 4. The second branch corresponds to the triangular points with positive y coordinate, as the $PAR(16)$ reveals, with the third branch being the triangular points with negative y coordinate. The last branch is that of the collinear points, blue in figure (5.9). Next thing to notice is that branch 4 has 3 user requested outputs with $PAR(2) = 0.01215$ to numerical precision. These are the three collinear points for the value of μ requested, which we will call "$UZ7$", "$UZ8$" and "$UZ6$" (UZ because they are user requested and the number comes from their order of appearance, they are $L1$, $L2$ and $L3$ respectively). The triangular points are also in their corresponding branches with type "$UZ2$" and "$UZ4$" ($L4$ and $L5$). These labels are needed for the next step. Finally, as expected from this particular problem, the triangular points seem to have the three values of non-zero periods for $\mu = 0.01215$, while the collinear points have two. All of it tells us the number of periodic orbits that emerge from them, that is, 3 periodic orbits from $L4$ and $L5$ respectively, and 2 from $L1$, $L2$ and $L3$.

We now continue with the script. Next line imports a python script with a function inside called *write_lagrange()*, which can be seen in figure (5.11). This function prints on the screen information regarding the purely imaginary eigenvalues of the libration points, it is in figure (5.12). Although this last step is not strictly necessary for the next one (calculation of periodic orbits), it summarizes the information gathered about the periodic orbits that will be computed on forward, as for example the long period orbits that emanate from $L4$ and $L5$.

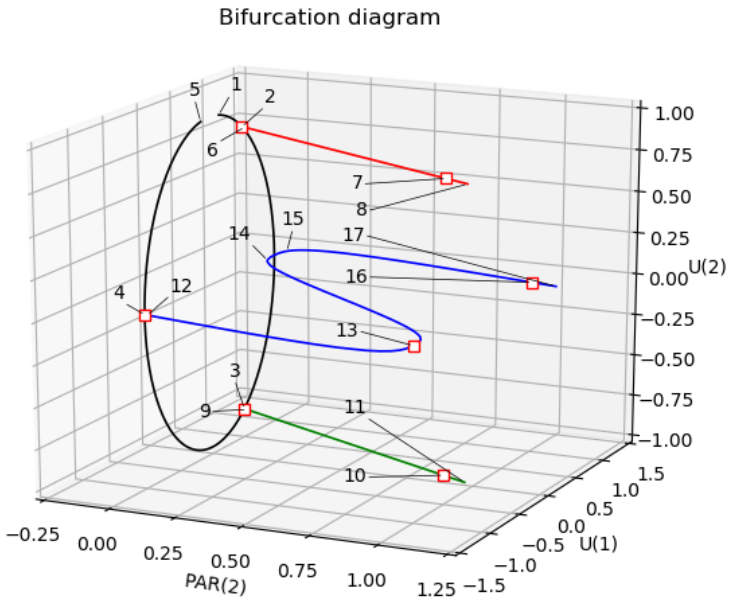
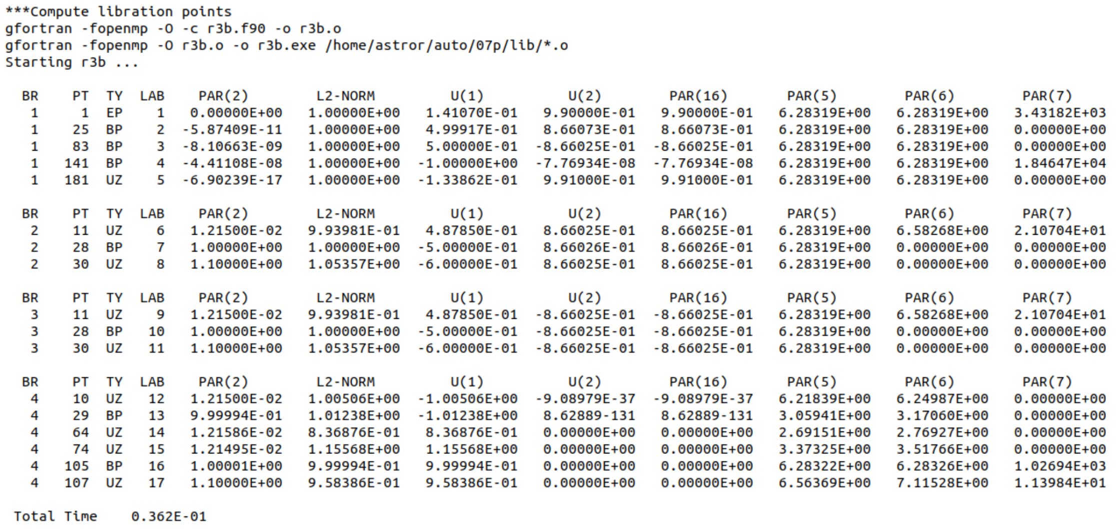
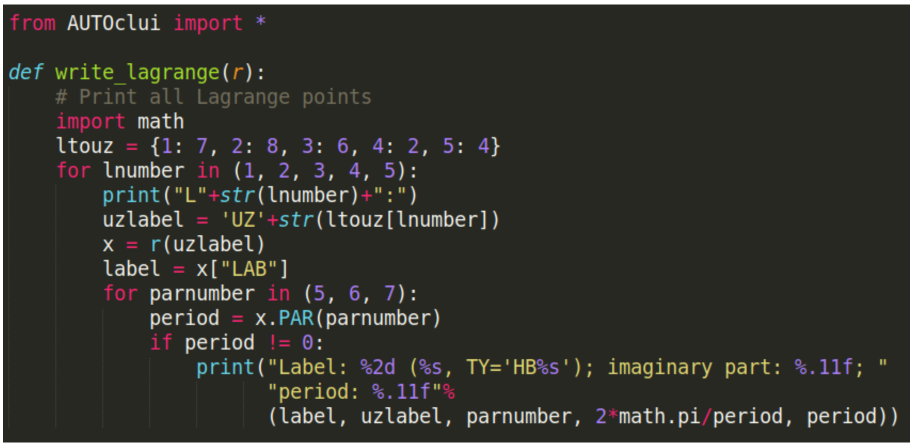


Figure 5.9 Bifurcation diagram of the stationary solutions in the CR3BP.



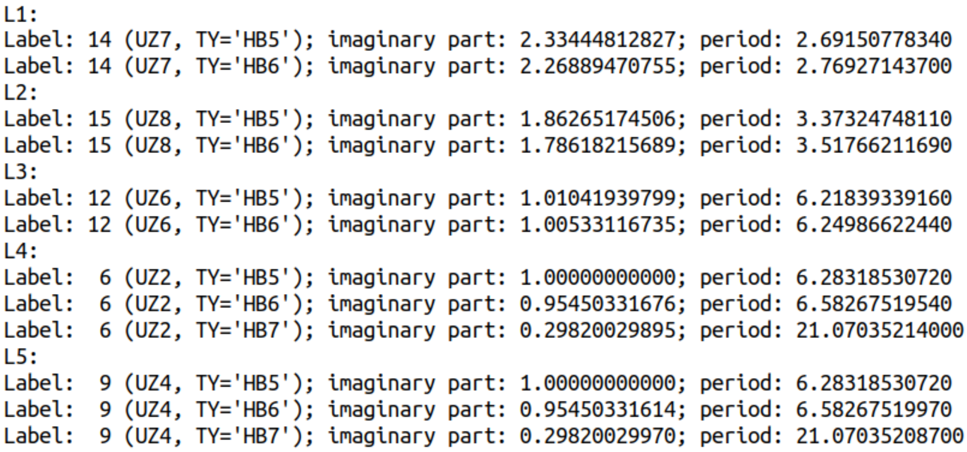
```
***Compute libration points
gfortran -fopenmp -O -c r3b.f90 -o r3b.o
gfortran -fopenmp -O r3b.o -o r3b.exe /home/astror/auto/07p/lib/*.o
Starting r3b ...

  BR    PT  TY  LAB    PAR(2)        L2-NORM         U(1)          U(2)         PAR(16)        PAR(5)        PAR(6)        PAR(7)
   1     1  EP    1   0.00000E+00   1.00000E+00   1.41070E-01   9.90000E-01   9.90000E-01   6.28319E+00   6.28319E+00   3.43182E+03
   1    25  BP    2  -5.87409E-11   1.00000E+00   4.99917E-01   8.66073E-01   8.66073E-01   6.28319E+00   6.28319E+00   0.00000E+00
   1    83  BP    3  -8.10663E-09   1.00000E+00   5.00000E-01  -8.66025E-01  -8.66025E-01   6.28319E+00   6.28319E+00   0.00000E+00
   1   141  BP    4  -4.41108E-08   1.00000E+00  -1.00000E+00  -7.76934E-08  -7.76934E-08   6.28319E+00   6.28319E+00   1.84647E+04
   1   181  UZ    5  -6.90239E-17   1.00000E+00  -1.33862E-01   9.91000E-01   9.91000E-01   6.28319E+00   6.28319E+00   0.00000E+00

  BR    PT  TY  LAB    PAR(2)        L2-NORM         U(1)          U(2)         PAR(16)        PAR(5)        PAR(6)        PAR(7)
   2    11  UZ    6   1.21500E-02   9.93981E-01   4.87850E-01   8.66025E-01   8.66025E-01   6.28319E+00   6.58268E+00   2.10704E+01
   2    28  BP    7   1.00000E+00   1.00000E+00  -5.00000E-01   8.66026E-01   8.66026E-01   6.28319E+00   0.00000E+00   0.00000E+00
   2    30  UZ    8   1.10000E+00   1.05357E+00  -6.00000E-01   8.66025E-01   8.66025E-01   6.28319E+00   0.00000E+00   0.00000E+00

  BR    PT  TY  LAB    PAR(2)        L2-NORM         U(1)          U(2)         PAR(16)        PAR(5)        PAR(6)        PAR(7)
   3    11  UZ    9   1.21500E-02   9.93981E-01   4.87850E-01  -8.66025E-01  -8.66025E-01   6.28319E+00   6.58268E+00   2.10704E+01
   3    28  BP   10   1.00000E+00   1.00000E+00  -5.00000E-01  -8.66025E-01  -8.66025E-01   6.28319E+00   0.00000E+00   0.00000E+00
   3    30  UZ   11   1.10000E+00   1.05357E+00  -6.00000E-01  -8.66025E-01  -8.66025E-01   6.28319E+00   0.00000E+00   0.00000E+00

  BR    PT  TY  LAB    PAR(2)        L2-NORM         U(1)          U(2)         PAR(16)        PAR(5)        PAR(6)        PAR(7)
   4    10  UZ   12   1.21500E-02   1.00506E+00  -1.00506E+00  -9.08979E-37  -9.08979E-37   6.21839E+00   6.24987E+00   0.00000E+00
   4    29  BP   13   9.99994E-01   1.01238E+00  -1.01238E+00   8.62889-131   8.62889-131   3.05941E+00   3.17060E+00   0.00000E+00
   4    64  UZ   14   1.21586E-02   8.36876E-01   8.36876E-01   0.00000E+00   0.00000E+00   2.69151E+00   2.76927E+00   0.00000E+00
   4    74  UZ   15   1.21495E-02   1.15568E+00   1.15568E+00   0.00000E+00   0.00000E+00   3.37325E+00   3.51766E+00   0.00000E+00
   4   105  BP   16   1.00001E+00   9.99994E-01   9.99994E-01   0.00000E+00   0.00000E+00   6.28322E+00   6.28326E+00   1.02694E+03
   4   107  UZ   17   1.10000E+00   9.58386E-01   9.58386E-01   0.00000E+00   0.00000E+00   6.56369E+00   7.11528E+00   1.13984E+01

 Total Time    0.362E-01
```

Figure 5.10 Screen output of the stationary solution analysis of the CR3BP.



```python
from AUTOclui import *

def write_lagrange(r):
    # Print all Lagrange points
    import math
    ltouz = {1: 7, 2: 8, 3: 6, 4: 2, 5: 4}
    for lnumber in (1, 2, 3, 4, 5):
        print("L"+str(lnumber)+":")
        uzlabel = 'UZ'+str(ltouz[lnumber])
        x = r(uzlabel)
        label = x["LAB"]
        for parnumber in (5, 6, 7):
            period = x.PAR(parnumber)
            if period != 0:
                print("Label: %2d (%s, TY='HB%s'); imaginary part: %.11f; "
                      "period: %.11f"%
                      (label, uzlabel, parnumber, 2*math.pi/period, period))
```

Figure 5.11 Python function to inspect the stationary solutions properties.



```
L1:
Label: 14 (UZ7, TY='HB5'); imaginary part: 2.33444812827; period: 2.69150778340
Label: 14 (UZ7, TY='HB6'); imaginary part: 2.26889470755; period: 2.76927143700
L2:
Label: 15 (UZ8, TY='HB5'); imaginary part: 1.86265174506; period: 3.37324748110
Label: 15 (UZ8, TY='HB6'); imaginary part: 1.78618215689; period: 3.51766211690
L3:
Label: 12 (UZ6, TY='HB5'); imaginary part: 1.01041939799; period: 6.21839339160
Label: 12 (UZ6, TY='HB6'); imaginary part: 1.00533116735; period: 6.24986622440
L4:
Label:  6 (UZ2, TY='HB5'); imaginary part: 1.00000000000; period: 6.28318530720
Label:  6 (UZ2, TY='HB6'); imaginary part: 0.95450331676; period: 6.58267519540
Label:  6 (UZ2, TY='HB7'); imaginary part: 0.29820029895; period: 21.07035214000
L5:
Label:  9 (UZ4, TY='HB5'); imaginary part: 1.00000000000; period: 6.28318530720
Label:  9 (UZ4, TY='HB6'); imaginary part: 0.95450331614; period: 6.58267519970
Label:  9 (UZ4, TY='HB7'); imaginary part: 0.29820029970; period: 21.07035208700
```

Figure 5.12 Information given by the *write_lagrange()* function about the libration points on screen.

5.4 Periodic orbits

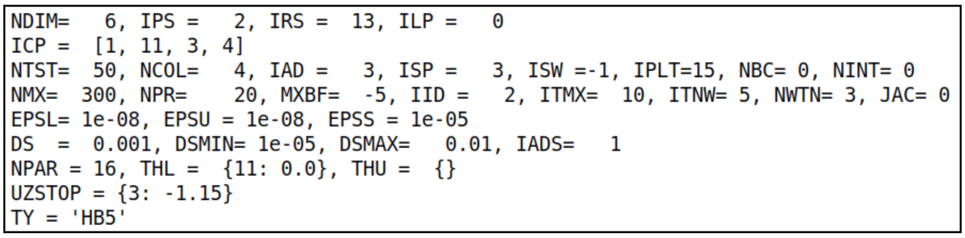
For the study of the periodic orbits the equations file will be the same as for the stationary solutions (*r3b.f90*), and the constants file will depend on the exact orbital family that has to be continued, as different families will need different stopping conditions or continuation step size to detect bifurcations along the family.

The classification of the orbital familes an its nomenclature will be the same as in [17], where it is shown that even though the equations of this system are apparently simple, the casuistry of the different orbital families is quiet complex. The families that will be shown in this section are only the ones that emerge from the libration points, with an explanation on how to compute them with AUTO and a brief commentary on its stability properties. Along these families there are bifurcations to others that won't appear here for the sake of brevity (they are represented in [17]), with the exception of the Halo orbits, that will be studied in detail in the next section. It is done this way because the Halo families can only be accessed through bifurcations as will be explained later on.

5.4.1 The planar Lyapunov families

The planar Lyapunov families are orbits that emanate form the $L1$, $L2$ and $L3$ Lagrange points and remain in the plane $z = 0$. First, we'll focus on the planar orbits that emerge from $L1$, which will be denoted as the **L**1 orbital family. The constants file used will be the one in figure (5.13), where the most relevant changes from the stationary case are:

- $IPS = 2$, for the computation of the periodic solutions.
- The main continuation parameter is now $PAR(1)$, the unfolding parameter, as it was explained in section (3.4.1). The other free parameters are the period ($PAR(11)$ in the AUTO enviroment), the energy of the orbit ($PAR(3)$) and $PAR(4)$, which was defined as the biggest Floquet multiplier (0 if more than one is outside the unit circle and 1 if all are inside or on the unit circle).
- The continuation step size and the maximum continuation step size are lower enough so AUTO can detect the existing branching orbits (this is commented later).
- In this case the stopping condition is close to the collision with the Moon.
- There is a new variable, *TY='HB5'*, which is there to choose the exact family that has to be continued. In the example of section (4.5) it wasn't necessary because there was only one pair of complex conjugate eigenvalues, so there was only one possible periodic orbit to calculate. Now, and for the $L1$ point, there are two possibilities (see figure (5.12)), so it has to be specified.



```
NDIM=   6, IPS =   2, IRS =  13, ILP =   0
ICP =  [1, 11, 3, 4]
NTST=  50, NCOL=   4, IAD =   3, ISP =   3, ISW =-1, IPLT=15, NBC= 0, NINT= 0
NMX=  300, NPR=   20, MXBF=  -5, IID =   2, ITMX=  10, ITNW= 5, NWTN= 3, JAC= 0
EPSL= 1e-08, EPSU = 1e-08, EPSS = 1e-05
DS  =  0.001, DSMIN= 1e-05, DSMAX=   0.01, IADS=   1
NPAR = 16, THL =  {11: 0.0}, THU =  {}
UZSTOP = {3: -1.15}
TY = 'HB5'
```

Figure 5.13 Constants file for the computation of the **L**1 family.

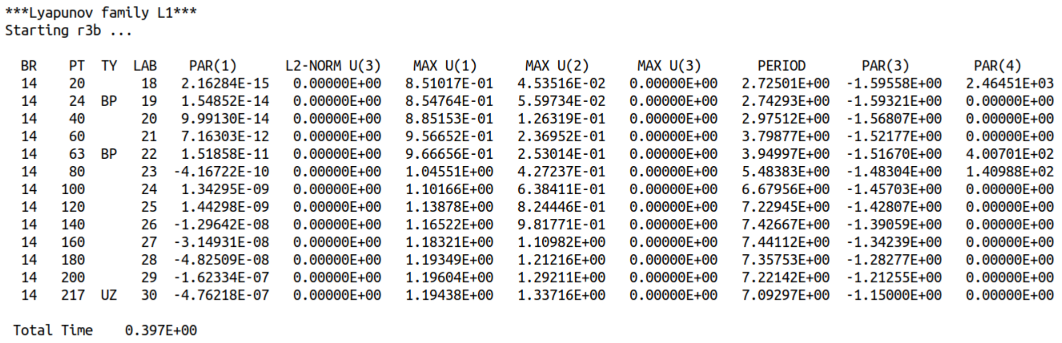
To compute the **L**1 family a new line has to be executed after the stationary solution analysis:

L1=run(r1('UZ7'), c='c.r3b.L1'),

so the point where the orbits are calculated from is $L1$. The screen output of this is in figure (5.14), where we can see that two orbits are branches to other families, they are the Halo (**H**1) and Axial (**A**1) families that emanate from **L**1. The results are stored using the *save* command, and the *s.* file has been used to extract the information of the periodic orbits, which have been plotted using Matlab. This is in figures (5.15) and (5.16). The two branching orbits are highlighted to better understand the result, and the Earth (left dot) and Moon (right dot) are in real scale.

By using the AUTO plotting function the period and biggest Floquet multiplier ($PAR(4)$) are represented in figure (5.17). The data in this figure is the result of applying a *relabel* to the output of the run in figure (5.14), so the labels don't coincide. The time in the period plot is not in seconds but in the scaled time τ of the non-dimensional equations of motion ($\tau = \omega t$, where ω is the rotational velocity of the frame of reference, which for the Earth-Moon system is $\omega = 2.665 \cdot 10^{-6}\ rad/s$, so that for $\tau = 5\ \rightarrow\ t \approx 20\ days$), which will be the unit used in this document most of the time.

From figure (5.17) the stability properties of the **L**1 family can be deducted in some degree. First conclusion is that, as suspected, no stable orbits exist in this family, as the parameter $PAR(4)$ is either greater than one or zero. Only the orbits where there is a sole multiplier outside the unit circle (o.u.c) are represented in this figure, and it could seem that the most stable orbit is that with $LAB = 6$ (the one with the lowest only multiplier o.u.c), but that may not be the case. Taking advantage of the fact that the information regarding the Floquet multipliers is stored in the *d.* file, this has been exported to a Matlab script to extract this information, so the absolute value of all the multipliers can be inspected. Although there is no plot, it has been found out that multipliers 1 and 2 are always on the unit circle up to numerical precision. Only the information of multipliers 3 to 6 have been represented in figure (5.18). This figure tells us that the first zero interval of $PAR(4)$ is due to $M4$ becoming greater than 1 in that interval, but its value is barely over 1.5, so it won't affect the stability of the orbits in a critical way (since they are already unstable and the other multiplier greater than one is three orders of magnitude bigger). Next, the multipliers $M5$ and $M6$ interchange its values in the branching orbits, so that together they are apparently continuous. The next zero interval (the one that begins with the orbit $LAB = 6$ in figure (5.17)) is due to $M4$ having a spike, but the fact that $PAR(4)$ remains zero after the spike seems inexplicable with only the information in figure (5.18). To better understand why, a detailed view of multipliers 5 an 6 has been represented in figure (5.19). There we can see that $M5$ starts to increase after the spike of $M4$, so there are always two multipliers with absolute value greater than one. The question remains if the most stable orbit is that of $LAB = 6$ or the one with the lowest $M6$ multiplier, *minM6* orbit on forward. For this matter to be answered a Matlab simulation of equations (5.1) has been set up for three orbits to be compared: $LAB = 1$, $LAB = 6$ and *minM6*. Figure (5.20) shows the result for the $LAB = 1$ and $LAB = 6$ orbits. There we can see that the stability of orbit $LAB = 1$ is non-existent, as it barely completes one orbit before the particle breaks away from the expected path (the numerical precision of the simulation is at its maximum and the starting data has all the decimals that AUTO provides). On the other hand, orbit $LAB = 6$ clings close to the path for almost one orbit and a half before it goes away. Figure (5.21) shows the result for the *minM6* orbit, where it seems the particle remains close to the nominal orbit a bit longer than $LAB = 6$ before breaking away. This shows us that the stability of the orbits may require more information than the $PAR(4)$ to completely discern the most stable of all.



```
***Lyapunov family L1***
Starting r3b ...

 BR    PT  TY  LAB    PAR(1)      L2-NORM U(3)    MAX U(1)      MAX U(2)      MAX U(3)       PERIOD        PAR(3)        PAR(4)
 14    20       18   2.16284E-15   0.00000E+00   8.51017E-01   4.53516E-02   0.00000E+00   2.72501E+00  -1.59558E+00   2.46451E+03
 14    24  BP   19   1.54852E-14   0.00000E+00   8.54764E-01   5.59734E-02   0.00000E+00   2.74293E+00  -1.59321E+00   0.00000E+00
 14    40       20   9.99130E-14   0.00000E+00   8.85153E-01   1.26319E-01   0.00000E+00   2.97512E+00  -1.56807E+00   0.00000E+00
 14    60       21   7.16303E-12   0.00000E+00   9.56652E-01   2.36952E-01   0.00000E+00   3.79877E+00  -1.52177E+00   0.00000E+00
 14    63  BP   22   1.51858E-11   0.00000E+00   9.66656E-01   2.53014E-01   0.00000E+00   3.94997E+00  -1.51670E+00   4.00701E+02
 14    80       23  -4.16722E-10   0.00000E+00   1.04551E+00   4.27237E-01   0.00000E+00   5.48383E+00  -1.48304E+00   1.40988E+02
 14   100       24   1.34295E-09   0.00000E+00   1.10166E+00   6.38411E-01   0.00000E+00   6.67956E+00  -1.45703E+00   0.00000E+00
 14   120       25   1.44298E-09   0.00000E+00   1.13878E+00   8.24446E-01   0.00000E+00   7.22945E+00  -1.42807E+00   0.00000E+00
 14   140       26  -1.29642E-08   0.00000E+00   1.16522E+00   9.81771E-01   0.00000E+00   7.42667E+00  -1.39059E+00   0.00000E+00
 14   160       27  -3.14931E-08   0.00000E+00   1.18321E+00   1.10982E+00   0.00000E+00   7.44112E+00  -1.34239E+00   0.00000E+00
 14   180       28  -4.82509E-08   0.00000E+00   1.19349E+00   1.21216E+00   0.00000E+00   7.35753E+00  -1.28277E+00   0.00000E+00
 14   200       29  -1.62334E-07   0.00000E+00   1.19604E+00   1.29211E+00   0.00000E+00   7.22142E+00  -1.21255E+00   0.00000E+00
 14   217  UZ   30  -4.76218E-07   0.00000E+00   1.19438E+00   1.33716E+00   0.00000E+00   7.09297E+00  -1.15000E+00   0.00000E+00

  Total Time    0.397E+00
```

Figure 5.14 Screen output of the **L**1 family.

This detailed study of the **L**1 family has been conducted so that the information on the following families is understood right away, and only the Halo families will be analysed more closely due to the purpose of this work.

The last two members of the planar Lyapunov families are the **L**2 and **L**3 that emanate from points L2 and L3 respectively. The process followed is the same as in the **L**1 family, with changes in the stopping conditions and continuation step size, as well as the corresponding change in the *run* command, where the initial point has to be chosen correctly (*r1('UZ8')* for L2 and *r1('UZ6')* for L3). Figures (5.22) to (5.23) show some information of the **L**2 family, and figures (5.24) to (5.26) of the **L**3 family.

For the **L**2 family the only comments are that three bifurcations to other families are found (the bifurcation

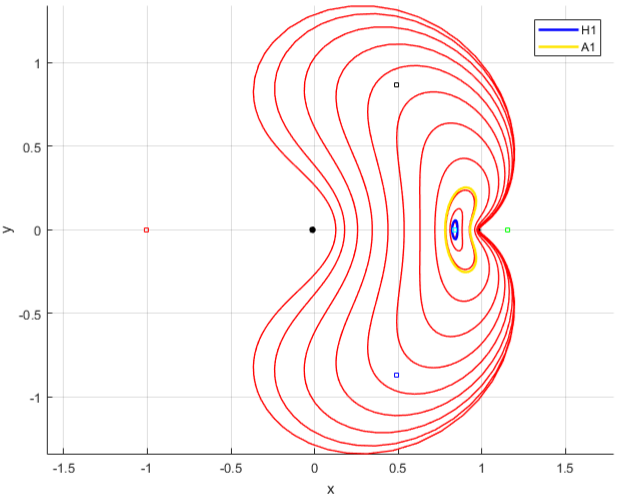


Figure 5.15 $\mathbf{L}1$ solution family, general view.

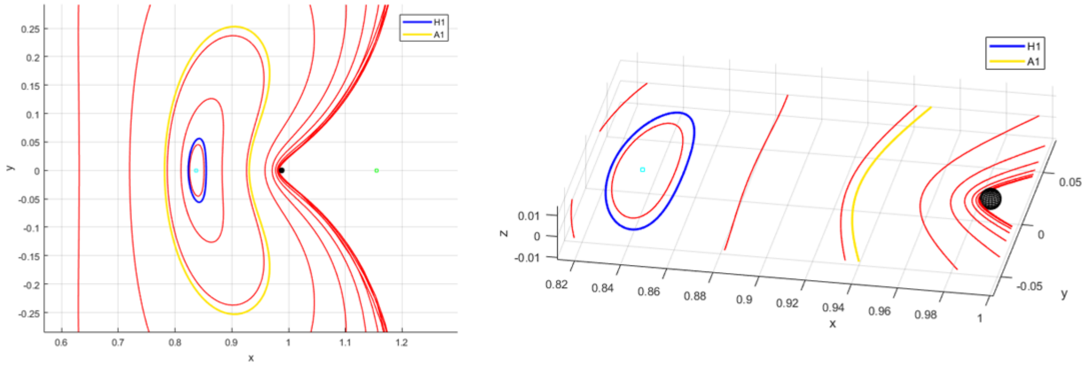


Figure 5.16 $\mathbf{L}1$ solution family, detailed views.

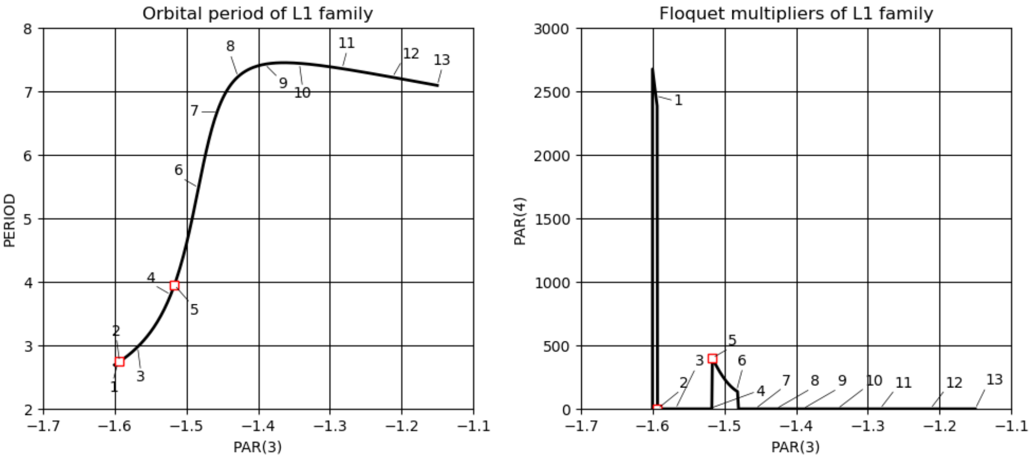


Figure 5.17 Period and biggest Floquet multiplier of the $\mathbf{L}1$ family.

to the $\mathbf{R}2$ family occurs after the collision with the Moon) and the the Floquet multipliers don't show any stable orbits whatsoever. The $\mathbf{L}3$ family on the other hand shows some interesting stability properties, as it has $PAR(4) = 1$ shortly after the last bifurcating orbit to the $\mathbf{S}3$ family. This points out that there are stable

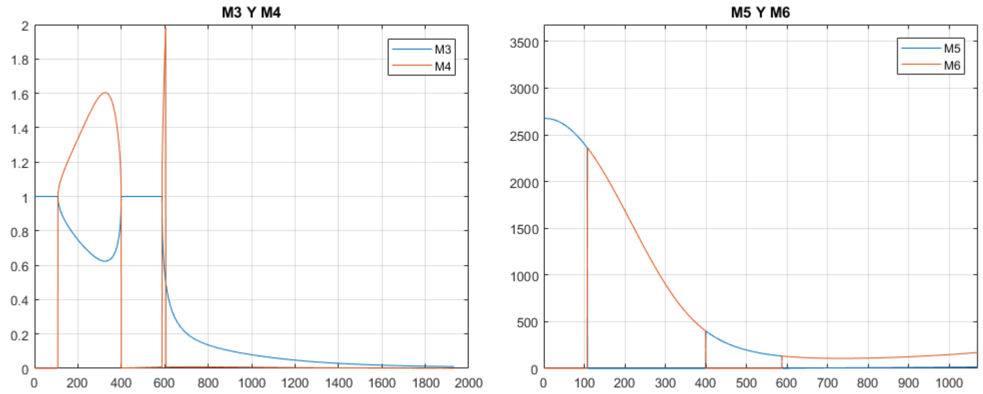


Figure 5.18 Floquet multipliers 3, 4, 5 and 6 of the **L**1 family.

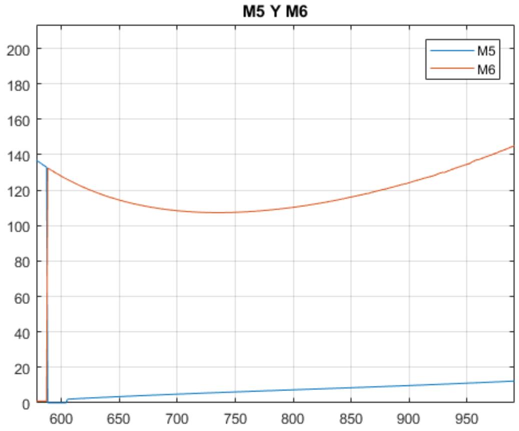


Figure 5.19 Detail of Floquet multipliers 5 and 6 of the **L**1 family.

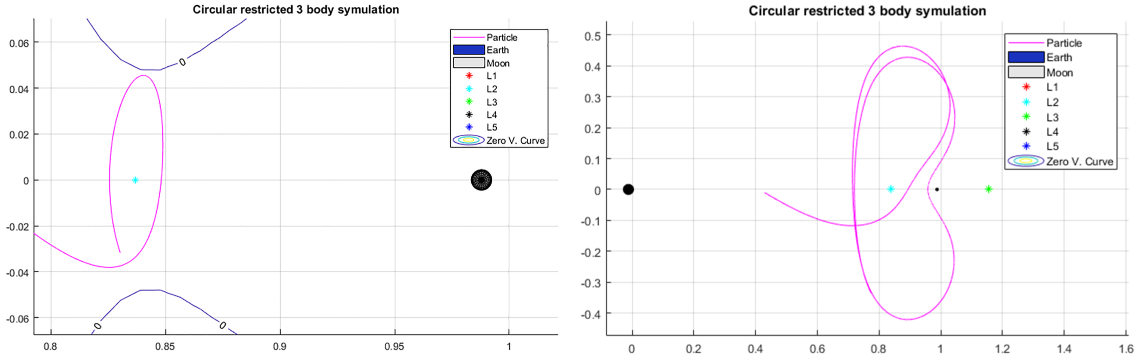


Figure 5.20 Simulation of orbits $LAB = 1$ (left) and $LAB = 6$ (right) of the **L**1 family.

orbits along this family, as the simulation of figure (5.26) confirms, where the simulation time is 4 times the orbital period and the path followed by the particle remains the same.

5.4.2 The Long-Period and Short-Period planar Lyapunov families

Emanating from the L4 and L5 libration points there are two symmetric families of planar orbits referred to as the "Long-Period" and "Short-Period" families, here they are shown for the L4 point.

The Short-Period, here denoted as **S**3 connects the L4 to the L5, while in the middle it connects to the **L**3 family (shown in the previous section). In order to compute it the *run* command has to start with the *r1('UZ2')* data point, and the constants file has to specify *TY='HB6'*. The result is in figures (5.27) and

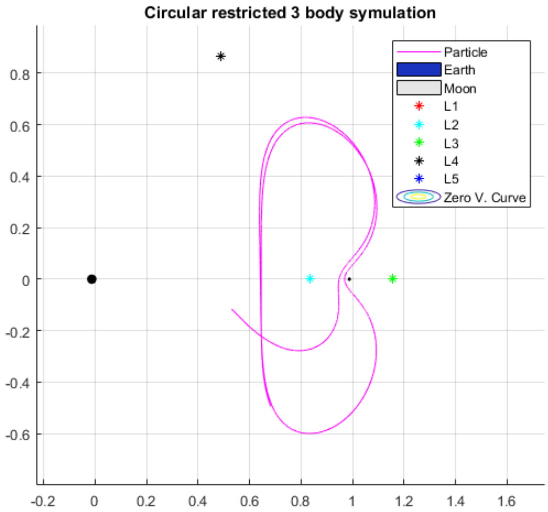


Figure 5.21 Simulation of orbit with lowest $M6$ of the $\mathbf{L}1$ family.

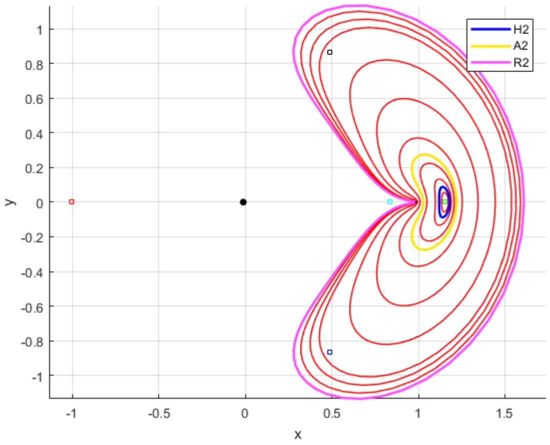


Figure 5.22 $\mathbf{L}2$ solution family, general view.

(5.28). As it is evident for the $PAR(4)$ diagram the family is always stable, which is compatible with the fact that the $\mathbf{L}3$ family was stable at the branching with $\mathbf{S}3$. The Long-Period family, here denoted as $\mathbf{L}4$, is computed similarly by specifying *TY='HB7'* in the constants file. The results are in figures (5.29) and (5.30). Even though there are no branch points along this family, there is a reverse period-quadrupling bifurcation at the end of it, where it connects with the $\mathbf{S}3$ family (the special nature of the connecting orbit makes it impossible to detect it from the $\mathbf{S}3$ orbit with this approach). As mentioned before, similar orbits emerge from L5 in a symmetric fashion.

5.4.3 The Vertical families

The Vertical families, here denoted as $\mathbf{V}1$-$\mathbf{V}5$ emanate from their respective libration points. To compute them the initial point has to be set as the corresponding libration point in the *run* command, and the TY variable has to be specified in the constants file as before (*TY='HB6'* for points L1 to L3 and *TY='HB5'* for the L4 and L5). Families $\mathbf{V}1$ to $\mathbf{V}4$ are represented in figures (5.31) and (5.32), as $\mathbf{V}4$ and $\mathbf{V}5$ connect in the middle and can be considered a single family, the $\mathbf{V}4/\mathbf{V}5$. The period and Floquet multipliers of families $\mathbf{V}1$, $\mathbf{V}3$ and $\mathbf{V}4$ are in figures (5.33) to (5.35), $\mathbf{V}2$ is not represented because its properties are very similar to $\mathbf{V}1$.

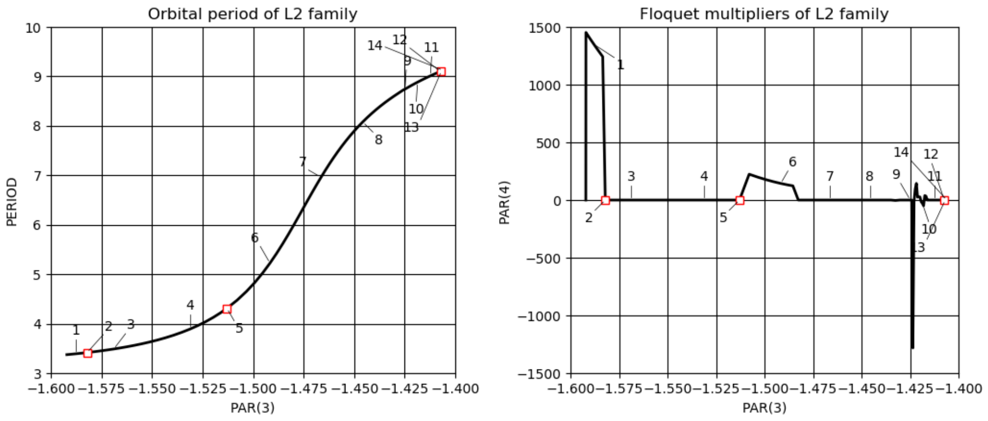


Figure 5.23 Period and Floquet multipliers of the **L**2 family.

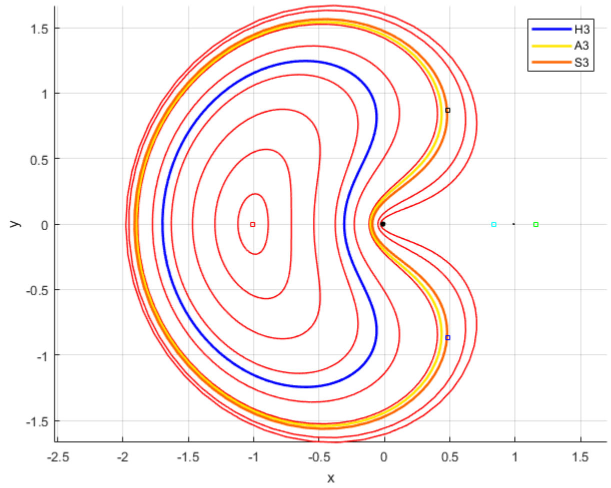


Figure 5.24 **L**3 solution family, general view.

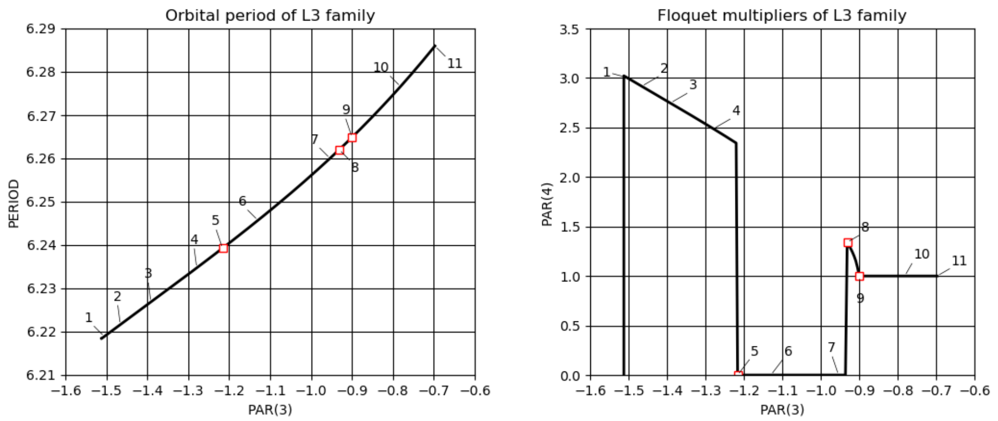


Figure 5.25 Period and Floquet multipliers of the **L**3 family.

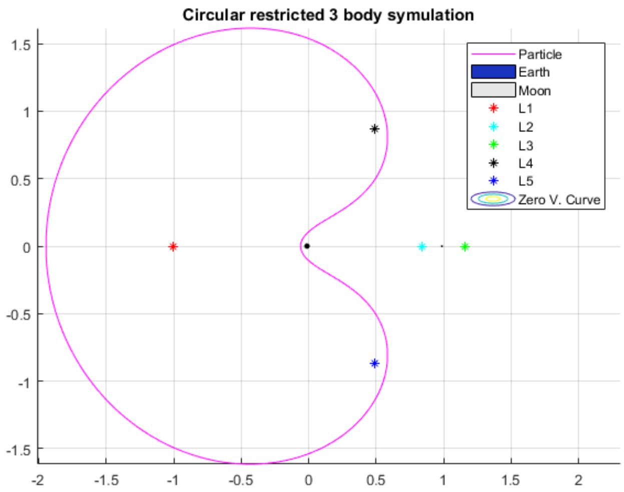


Figure 5.26 Simulation of orbit $LAB = 10$ of the $\mathbf{L}3$ family for several periods.

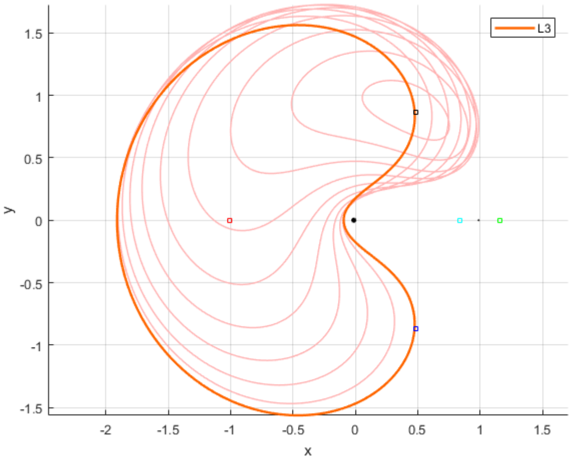


Figure 5.27 S3 solution family, general view.

5.5 The Halo Families

Up until now, all the orbital families that have been calculated are the result of continuing the cycles that emerge from the five libration points as has been explained. But in the process AUTO has revealed that some other paths of continuation are available through the bifurcating orbits. AUTO can detect other branches by monitoring the rank of the system matrix and computes the bifurcation directions, more information on this can be found in [18]. Now we use those branching orbits to accede to the *Halo orbital families*, which are the subject of study.

To achieve the branch switching we need to feed the orbit to the *run* command as in the next example for the Halo orbit $\mathbf{H}1$, that which emerges from the $\mathbf{L}1$ family:

H1 = run(L1('BP1'), c='c.H1'),

where, as in the case of the computation of emerging periodic orbits, the equations file isn't necessary (it is implicit in the solution orbit that we are feeding as input), and the *L1('BP1')* refers to the first bifurcating orbit of the $\mathbf{L}1$ family that we found, which was named after $\mathbf{H}1$, the Halo family that emanates from the Planar Lyapunov family of the L1 Lagrange point. For the constants file there are a couple of things to comment. First is that the constant *ISW* has to be set to -1 so that AUTO automatically performs the branch switching, and second that the continuation step size *DS* is now sign sensitive. This is because the Halo families have

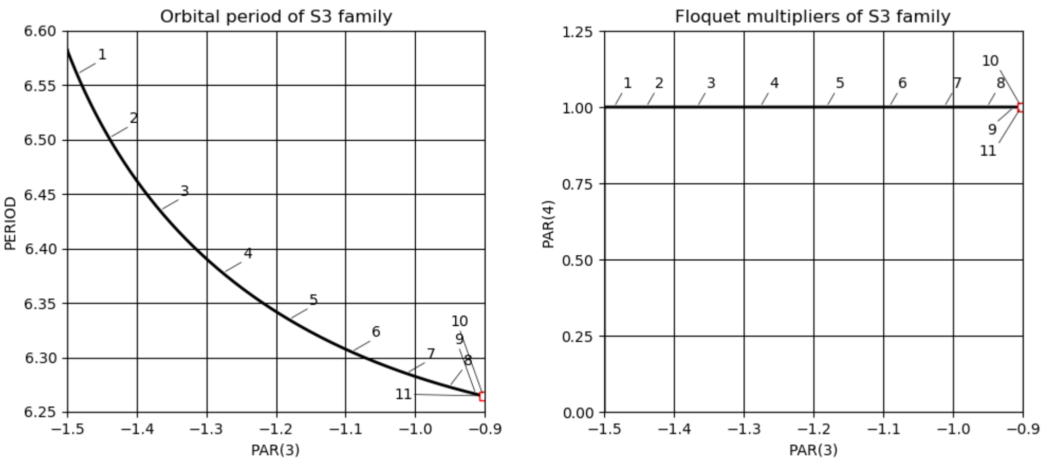


Figure 5.28 Period and Floquet multipliers of the **S**3 family.

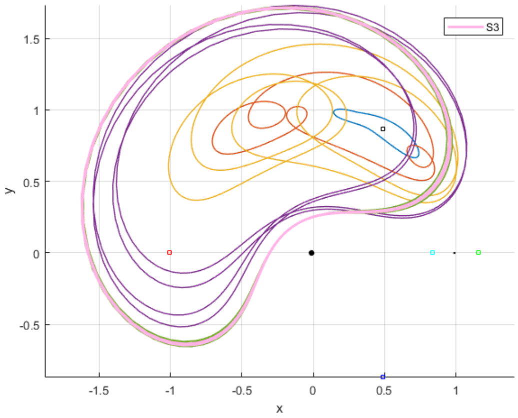


Figure 5.29 **L**4 solution family, general view.

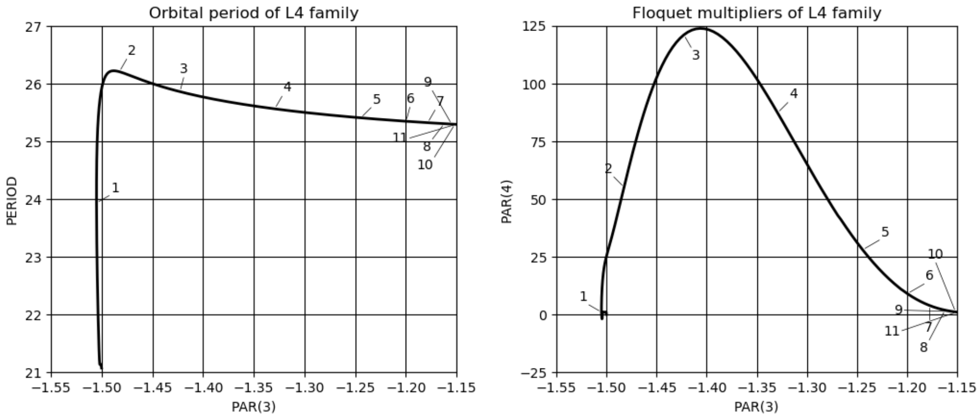


Figure 5.30 Period and Floquet multipliers of the **L**4 family.

two variants, the Northern and Southern case, a result of the symmetry in the z axis in the equations of the system. The rest of the constants have the same role as in the previous cases, so we now proceed to show the results of the Halo families.

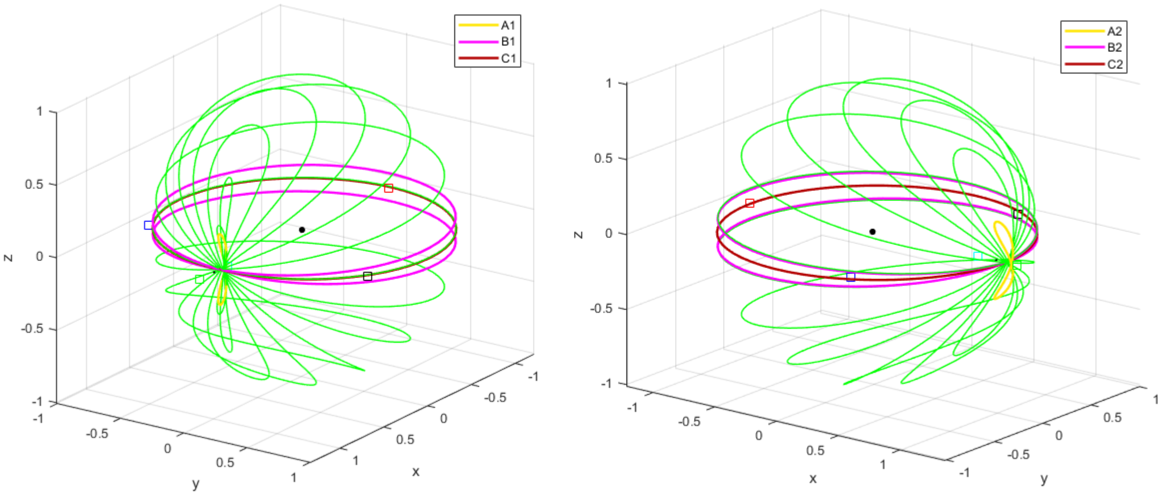


Figure 5.31 **V**1 (left) and **V**2 (right) solution family, general view.

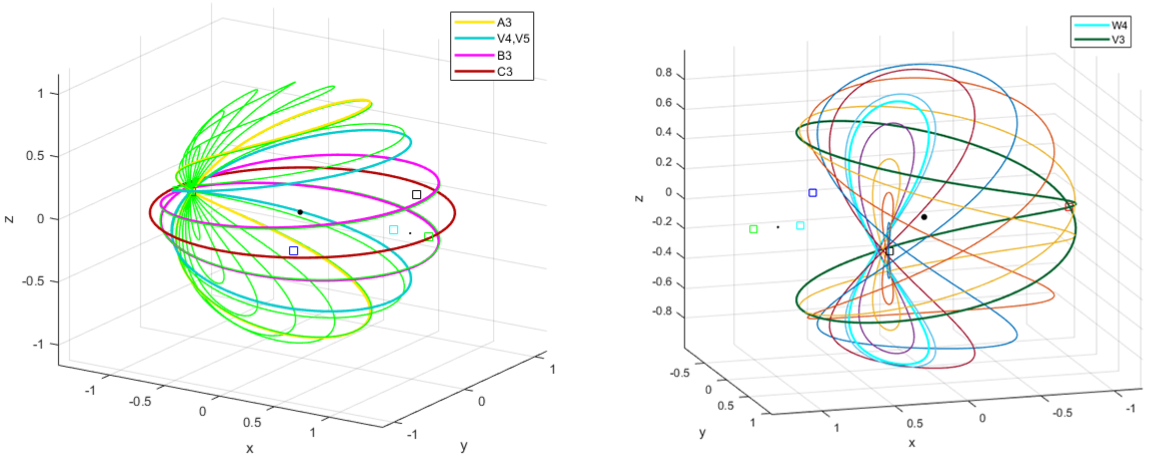


Figure 5.32 **V**3 (left) and **V**4 (right) solution family, general view.

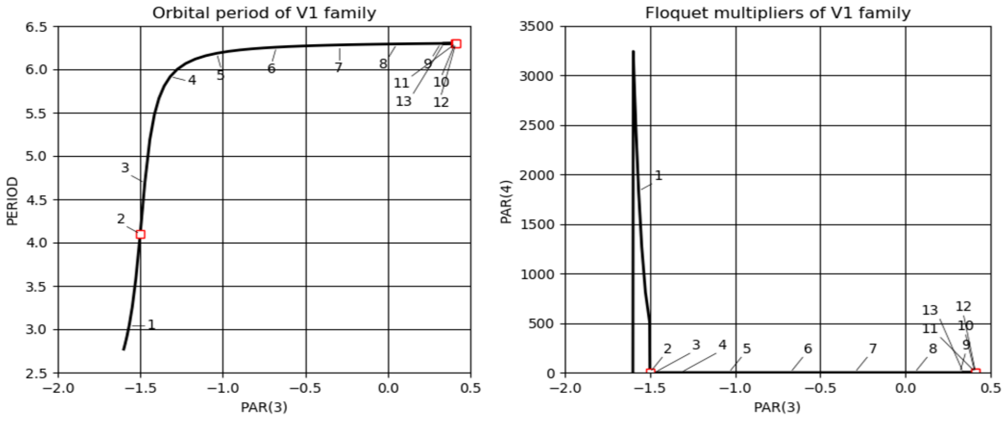


Figure 5.33 Period and Floquet multipliers of the **V**1 family.

The results for the **H**1 family are in figures (5.36) to (5.38). Figure (5.36) shows how the **H**1 family emerges from the L1, then it develops to a near rectilinear orbit around the moon to finally intersect with the **C**1 family. Had the family been continued then it would have resulted in the southern variant of this orbital

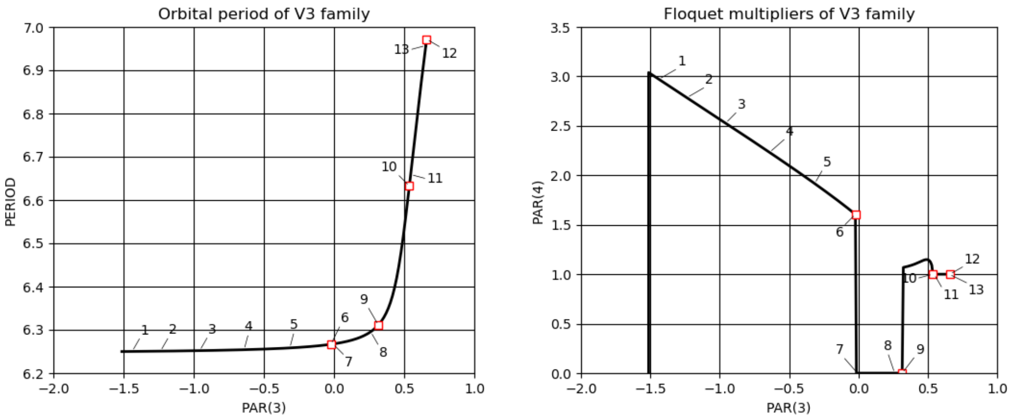


Figure 5.34 Period and Floquet multipliers of the **V**3 family.

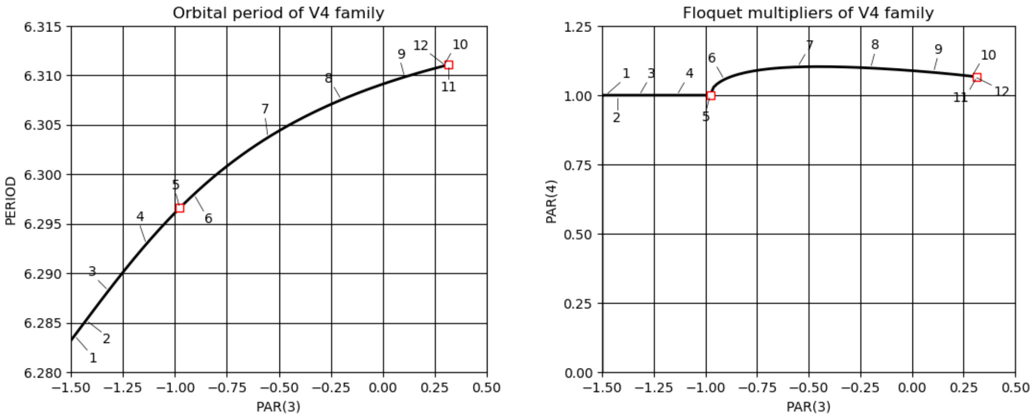


Figure 5.35 Period and Floquet multipliers of the **V**4 family.

family, whit the same properties. The Floquet multipliers in (5.37) only show that this family of orbits is really unstable at the beginning, but rapidly goes to a near stable behaviour. Even more, figure (5.38) shows how there is an actual stretch of the family with complete stability (in the CR3BP model). This orbits will be studied in detail in the next subsection.

For the **H**2 family results see figures (5.39) to (5.41), where a similar behaviour for the Floquet multipliers can be seen at the end of the family, more details in the next subsection.

Finally, the **H**3 family, though of less practical interest, has been represented in figures (5.42) to (5.43). A closer look at the Floquet multipliers shows that it is quasi-stable at the beginning (low $PAR(4)$) but becomes completely unstable later on.

Out of these three families the **H**1 and **H**2 have been proposed as suitable candidates for an inhabited facility in cis-lunar space [19]. A narrow stretch where they are called *Near Rectilinear Halo Orbits* or *NRHOs* presents some interesting properties, these are studied in some detail next.

5.6 Near rectilinear Halo orbits

As their name indicates, these subgroup of orbits are characterized by an elongated path as seen from Earth, but we are now going to formally delimit the stretch of orbits in both the **H**1 and **H**2 families we are referring to. In section (5.4.1) it was already commented that one of the properties of the Floquet multipliers is that there is always a pair equal to 1 (the trivial pair), now another property is required for this classification. In this context, the multipliers always appear in reciprocal pairs $(\mu_i, 1/\mu_i)$, so it is useful to define the stability index $\nu_i = \frac{1}{2}(\mu_i + 1/\mu_i)$ for the two non-trivial pairs. Similarly than with the Floquet multipliers, this metric

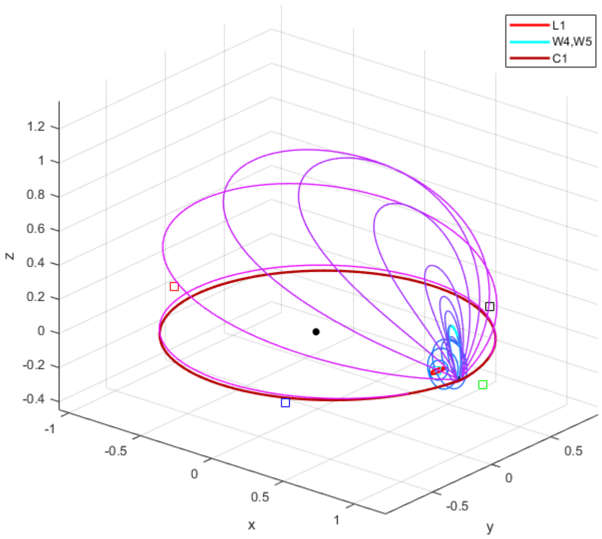


Figure 5.36 **H**1 solution family.

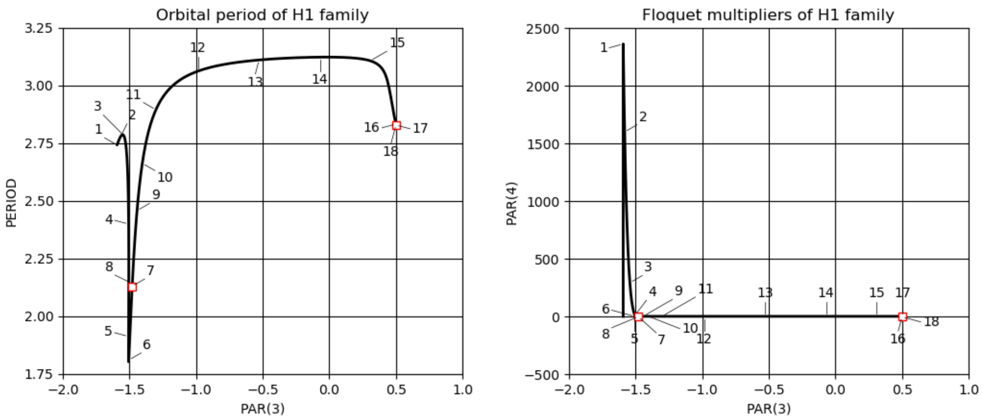


Figure 5.37 Period and Floquet multipliers of the **H**1 family.

indicates stability when both of them are less than one in modulus. For an unstable orbit a larger value of the stability index corresponds to a faster departure from the nominal path.

In general, the NRHOs are considered as the subset of Halo orbits possessing stability indices within some small bound surrounding ±1 and similar in magnitude. For this work, the stability switches of this metric will be use to delimit the boundaries of the NRHOs. Across the **H**1 and **H**2 families and near the Moon region a number of stability switches occur from linearly stable to unstable when continuing the families from the initial bifurcations. Unfortunately, this wasn't easily seen with the $PAR(4)$ vs $PAR(3)$ plots in last section, but the metric v_i in relation with the closest distance to the Moon of each orbit will appear as a more suitable way to discern the evolution of the stability properties. That is what is represented in figure (5.44) for both the **H**1 and **H**2 families in the region near the Moon.

Then, for the **H**1 family the orbits are considered of the NRHO subset between the first and forth stability switch (from right to left), marked with arrows in (5.44). This region starts with the orbits of around $17400Km$ of closest distance and ends a bit after the collision orbit (after this the orbits are of no real practical interest). For the **H**2 family the subset is considered between the first and third intersection, thus being the region between $15500Km$ and $90Km$ of closest distance. This definition applies to both the northern and southern members of the families. This plot also highlights the stable regions in green when both the indices are below 1 in modulus.

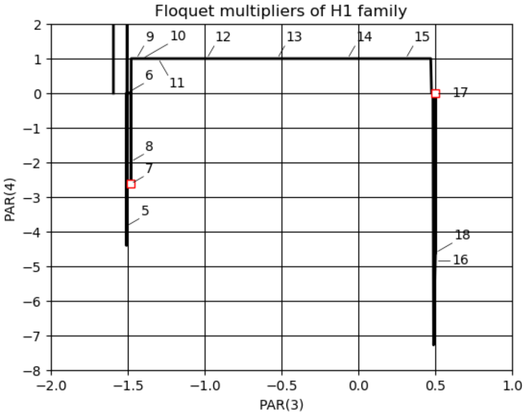


Figure 5.38 Detail of the Floquet multipliers in the **H**1 family.

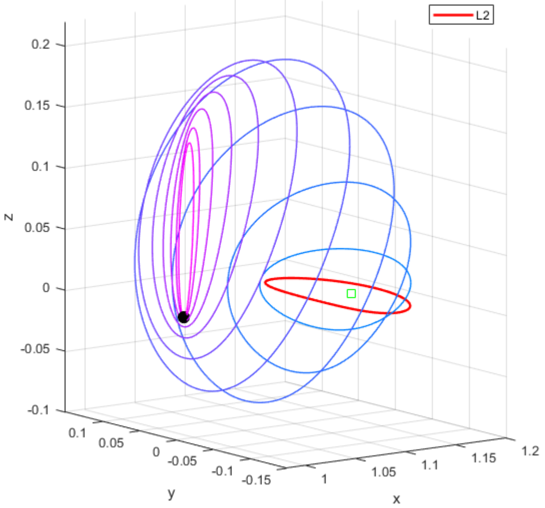


Figure 5.39 **H**2 solution family.

For this subset of orbits the period and energy values are represented in figures (5.45) and (5.46), with the period now in days and the energy is non-dimensional (5.3).

The subset of orbits under study is now set, and the next step is to study its properties regarding the stability. Figure (5.44) shows us that the stability inside the NRHO subset is significantly better than before it starts. In order to estimate the temporal scale of the dominant diverging motion a new metric is derived by considering the time constant, ψ, as the number of revolutions before departure:

$$\psi[rev] = \frac{1}{|Re(ln(\mu_{max}))|}$$

As it has been defined the time constant is infinite for a marginally stable orbit ($\mu_{max} = 1$). The time constant can be interpreted as an estimation of the number of revolutions for an initial perturbation to be amplify by a factor of approximately 3. For both the NRHOs families ψ is plotted in figure (5.47). This figure shows the previously indicated possibility of maintaining a station in NRHO motion over long periods of time while consuming few propellant resources. In order to better show the behaviour of these orbits simulations of some them are in figure (5.48). The 86 Km orbit has been included as an example of stable orbit very close to the surface, and the 290 Km orbit, though unstable, shows that it is possible for these kind of orbits to remain close to the reference for over a very long time, as the simulation shows very little dispersion over the seven periods.

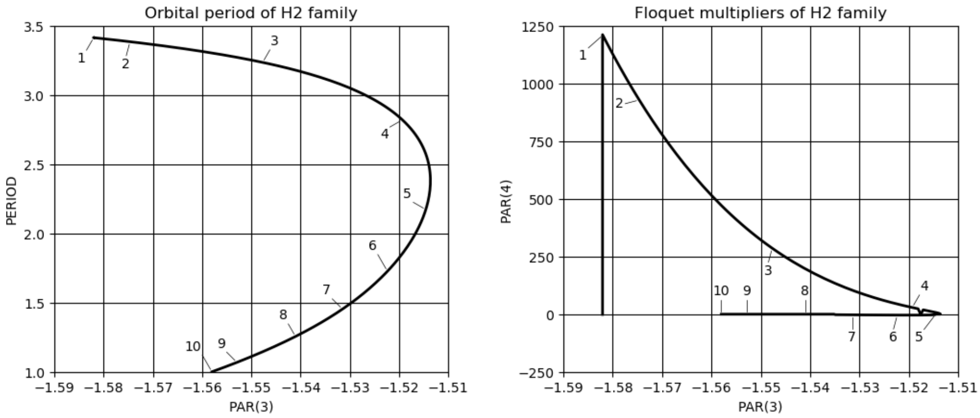


Figure 5.40 Period and Floquet multipliers of the **H**2 family.

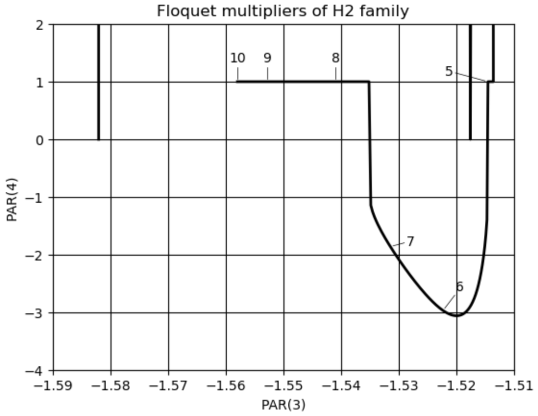


Figure 5.41 Detail of the Floquet multipliers of the **H**2 family.

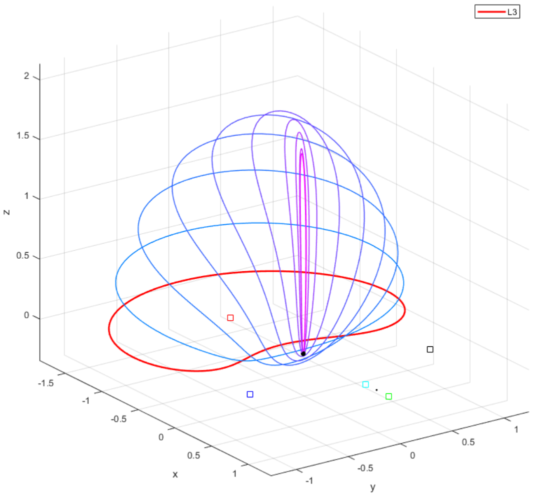


Figure 5.42 **H**3 solution family.

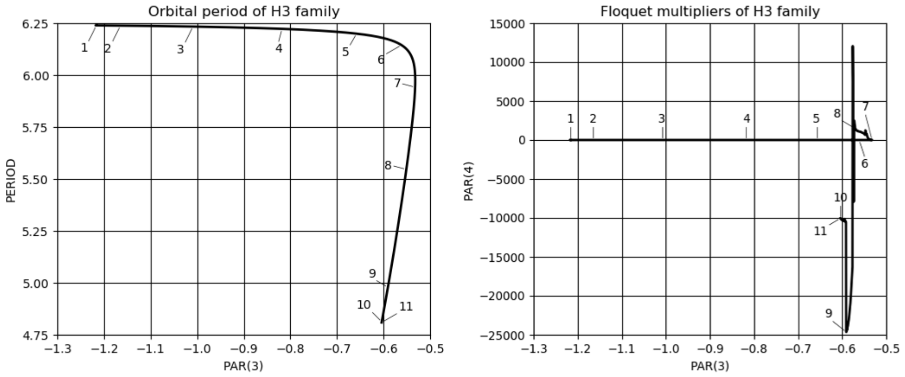


Figure 5.43 Period and Floquet multipliers of the **H**3 family.

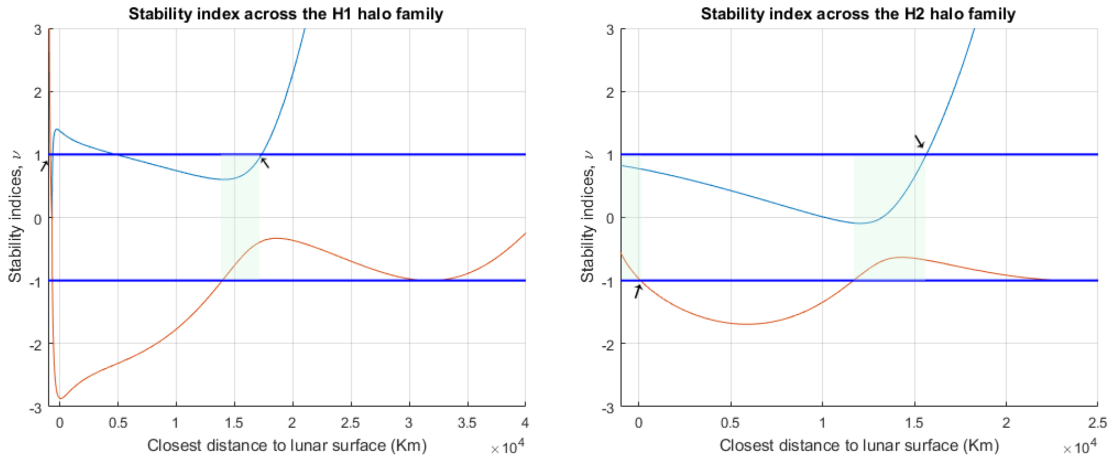


Figure 5.44 Region of stability switches in the vicinity of the Moon for the **H**1 and **H**2 families.

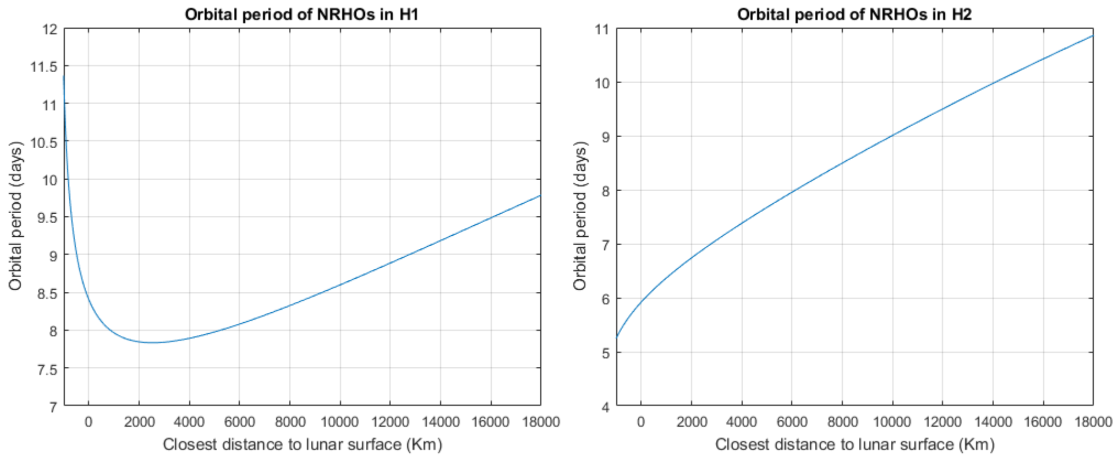


Figure 5.45 Period in the NRHO subset.

Another important factor when designing an orbit for a space station that is going to be inhabited are the thermal and power restrictions that come with the eclipsing of the sun. In this case the aforementioned station can be eclipsed by both the Moon and the Earth. To study the first type of eclipsing it is necessary to compare the period of the NRHO with the synodic period of the Moon ($T_{M-S} \approx 29.5306$ days), that is represented in figure (5.49) for the $y:1$ and $y:2$ resonances in the **H**2 NRHO family. The $y:1$ resonance means that the orbit performs y revolutions over one synodic period, and $y:2$ means y revolutions over two

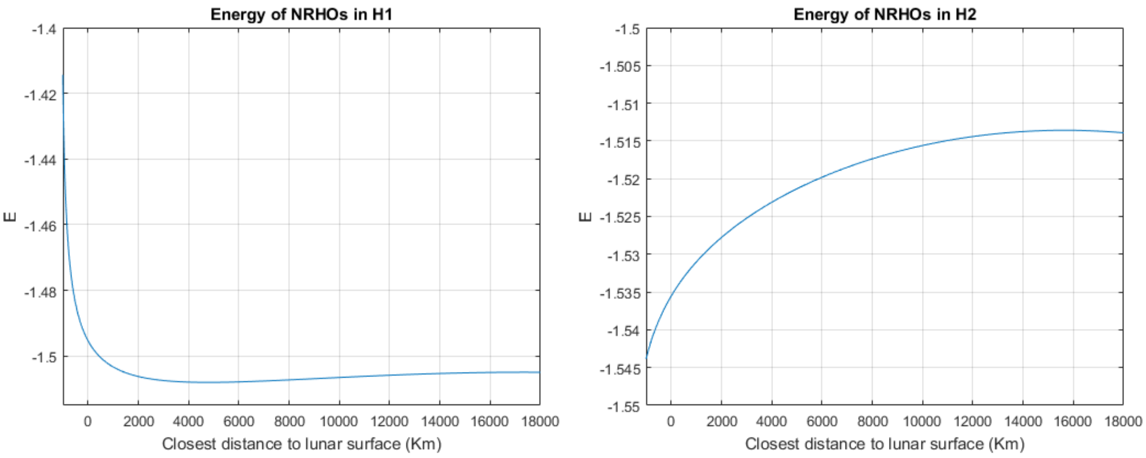


Figure 5.46 Energy in the NRHO subset.

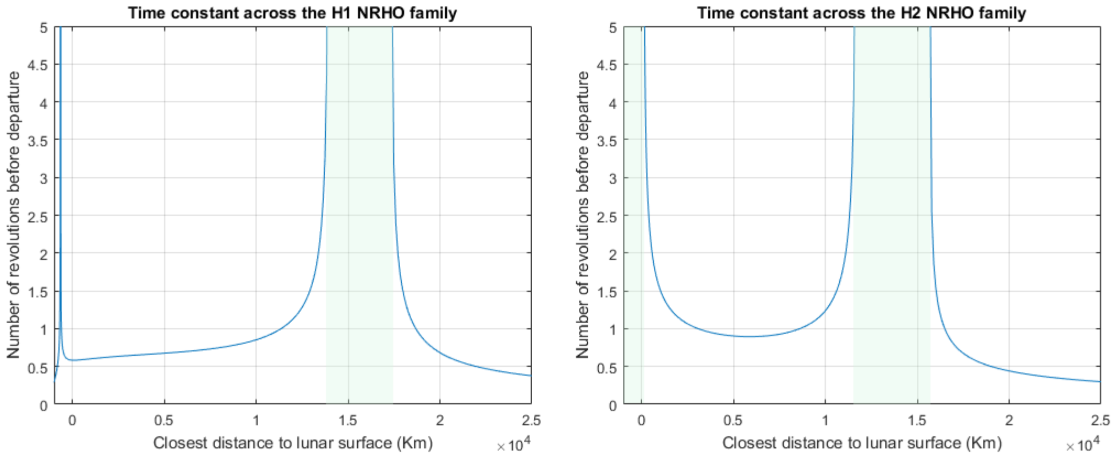


Figure 5.47 Time constant in the NRHO subset.

synodic period. As can be seen there are only two $y:1$ resonances in the NRHO subset for the H2 family (the $4:1$ and $3:1$ resonances) and two $y:2$ resonances (the $9:2$ and $7:2$), three of these orbits are in figure (5.50). For this representation a Moon centred Sun oriented frame of reference has been used, as it makes the resonances obvious. In this frame of reference the Sun is always in the same direction, and thus the orbits with resonances of the $y:1$ type allow for orientations such that the Moon's shadow never casts over it, the orbit orientation is set by choosing the insertion date correctly. The Earth's shadow issue isn't studied here, but a possible strategy is to try set the spacecraft apoapsis during each full Moon phase, thus avoiding the Earth's shadow. On the other hand, the $9:2$ resonance difficultly avoids the lunar eclipse because the shadow doesn't always cleanly pass through the gaps in the orbit, so epoch selection is more challenging in this case. Finally, the $3:1$ resonance has the added advantage of being stable in the CR3BP context, making it a very interesting option for a space station orbit (if no other mission restrictions are considered).

It is also important to study the way in and out from these orbits, and for that the computation of manifolds with AUTO will be shown in the next section.

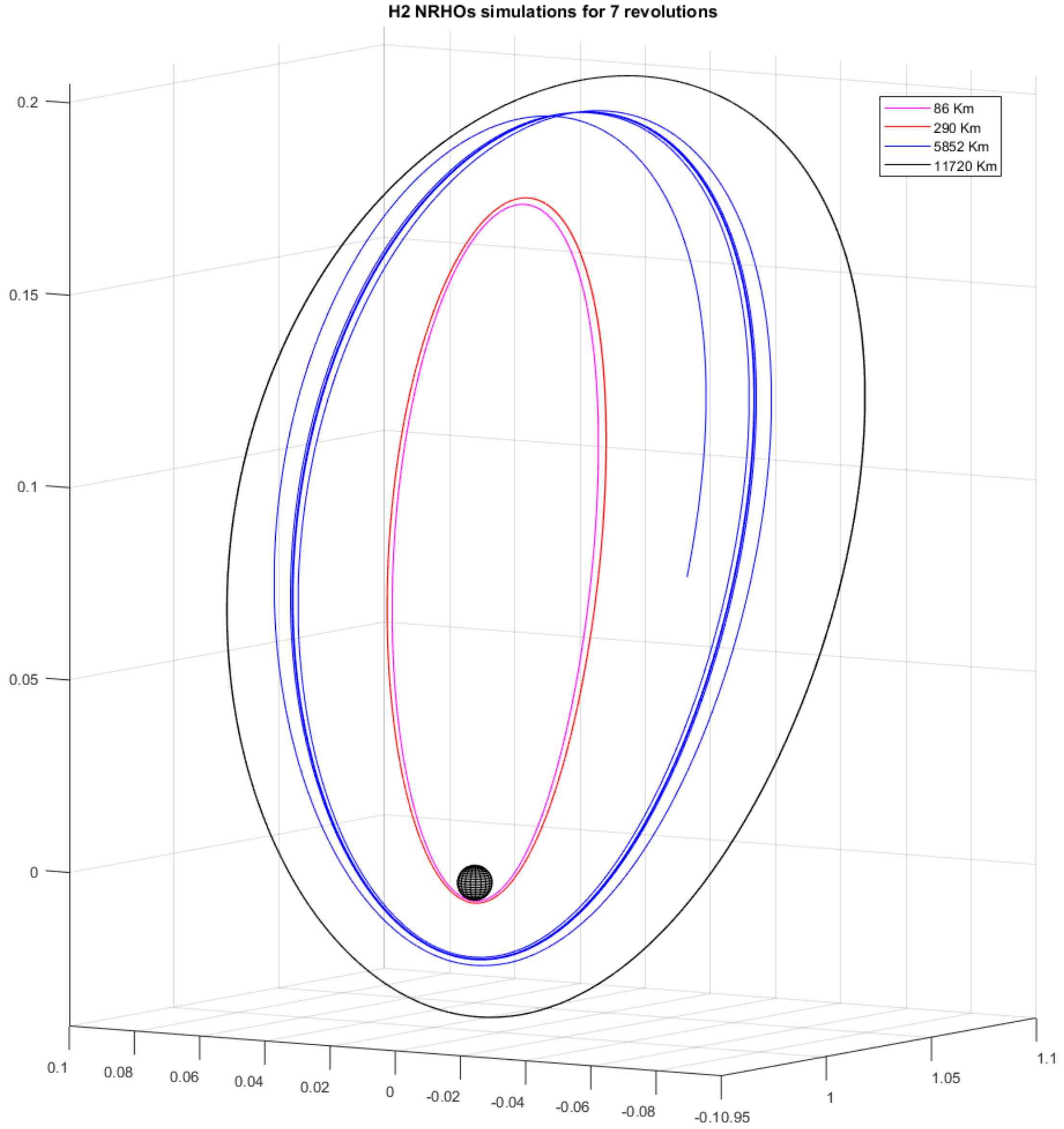


Figure 5.48 Simulation of various **H**2 NRHOs over 7 periods (slightly different in each of them).

5.7 Computation of manifolds with AUTO

Though the strategy to compute unstable manifolds has already been explained in section (3.5), the details to do it in AUTO are now commented in some depth. The general idea was to compute the eigenfunction associated to the only unstable multiplier. Once calculated, the manifold was approximated as the perturbed state $\mathbf{r}(0) = \mathbf{u}(0) + \varepsilon \mathbf{w}(0)$. This state was continued using time integration until a certain condition was met (some coordinate in the final state reaches a desired value), and the resulting orbit was used as the initial iteration to continue the whole manifold. The process is now exemplified with an orbit in the **H**1 family.

The process starts with the already calculated family of periodic orbits **H**1. First step is to prepare the initial solution for the eigenfunction computation, and in order to do that the python script in figure (5.51) has been used. The code receives as input the solution file of the **H**1 family (previously saved), the label of the desired orbit and the value of ε in the first approximation of the manifold that will be used (*step* variable in the code). If no Floquet multiplier is introduced then the code uses the value of $PAR(4)$, which in the present case is different from zero and positive. For this example the chosen orbit has $PAR(4) = 10.59$ and the closest distance to the Moon is 23600 Km.

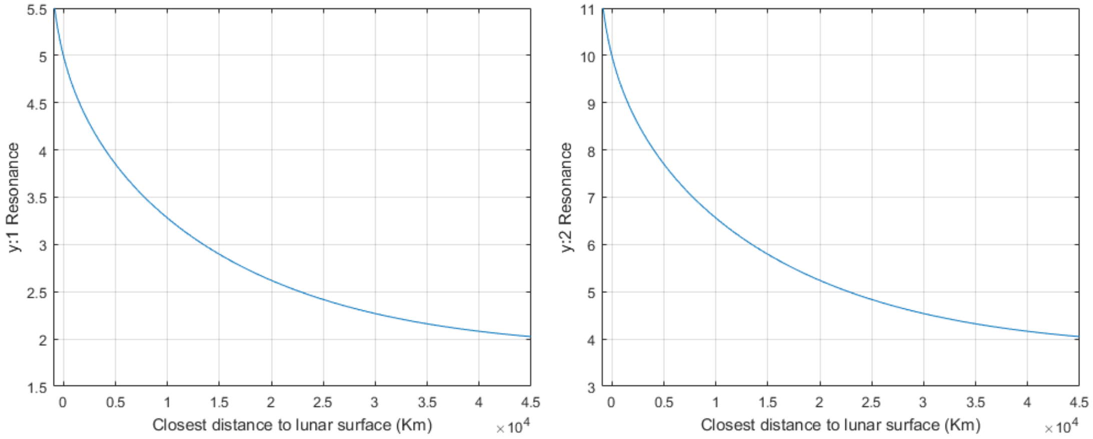


Figure 5.49 $y:1$ and $y:2$ resonances along the NRHOs in the **H**2 family.

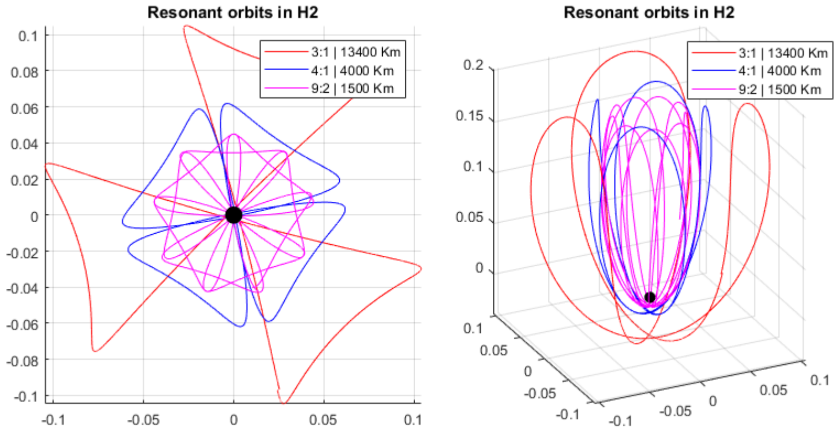
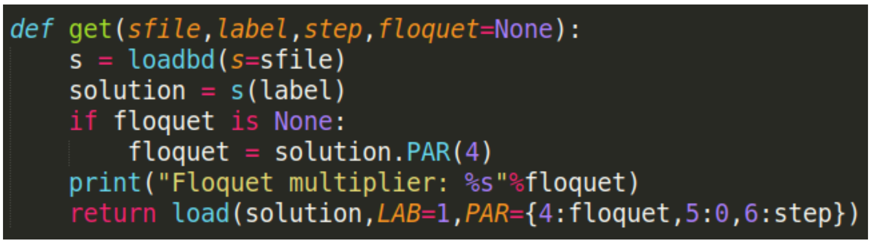


Figure 5.50 Some resonant **H**2 NRHOs in a Moon-Sun rotating frame of reference.

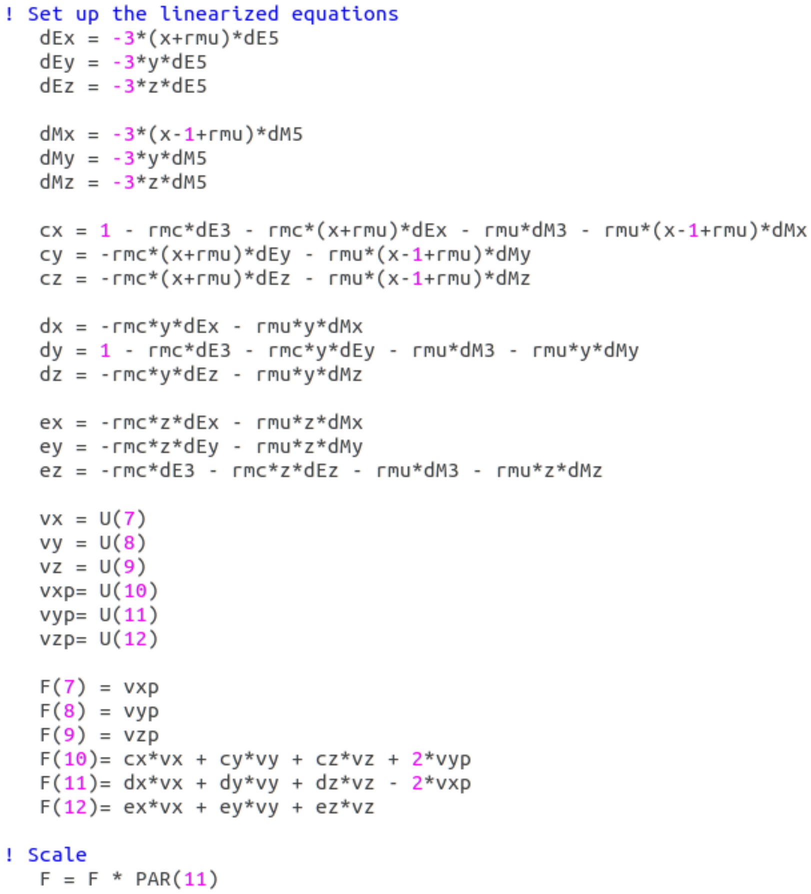


```python
def get(sfile,label,step,floquet=None):
    s = loadbd(s=sfile)
    solution = s(label)
    if floquet is None:
        floquet = solution.PAR(4)
    print("Floquet multiplier: %s"%floquet)
    return load(solution,LAB=1,PAR={4:floquet,5:0,6:step})
```

Figure 5.51 Python code to extract the information of the periodic orbit and prepare it for the eigenfunction calculation.

Next step is to compute the Floquet eigenfunction by solving the BVP outlined in section (3.5), and for that it is necessary to create a new AUTO problem with different equations and the appropriate boundary and integral conditions. In the equations file the $FUNC$ subroutine is the same except for the added linearised equations that extend the system in the eigenfunction variables, this is in figure (5.52). The boundary and integral conditions are in figure (5.53), where $PAR(5)$ is the adjoint variable ρ that acts eigenfunction modulus and will be used to stop the continuation at $\rho = 1$. The constants file needs to specify $IPS = 4$ in order to set a BVP, and the main continuation parameter is α, as in the periodic orbit computation.

The output of the eigenfunction calculation is then fed to the function in figure (5.54), so that the linear approximation of the manifold is calculated for the next step. This code creates a series of parameters needed for the following continuation problem, which is the time integration. For this, the equations file has the same $FUNC$ subroutine that the periodic orbit computation, but it includes some boundary conditions, as



```
! Set up the linearized equations
   dEx = -3*(x+rmu)*dE5
   dEy = -3*y*dE5
   dEz = -3*z*dE5

   dMx = -3*(x-1+rmu)*dM5
   dMy = -3*y*dM5
   dMz = -3*z*dM5

   cx = 1 - rmc*dE3 - rmc*(x+rmu)*dEx - rmu*dM3 - rmu*(x-1+rmu)*dMx
   cy = -rmc*(x+rmu)*dEy - rmu*(x-1+rmu)*dMy
   cz = -rmc*(x+rmu)*dEz - rmu*(x-1+rmu)*dMz

   dx = -rmc*y*dEx - rmu*y*dMx
   dy = 1 - rmc*dE3 - rmc*y*dEy - rmu*dM3 - rmu*y*dMy
   dz = -rmc*y*dEz - rmu*y*dMz

   ex = -rmc*z*dEx - rmu*z*dMx
   ey = -rmc*z*dEy - rmu*z*dMy
   ez = -rmc*dE3 - rmc*z*dEz - rmu*dM3 - rmu*z*dMz

   vx = U(7)
   vy = U(8)
   vz = U(9)
   vxp= U(10)
   vyp= U(11)
   vzp= U(12)

   F(7) = vxp
   F(8) = vyp
   F(9) = vzp
   F(10)= cx*vx + cy*vy + cz*vz + 2*vyp
   F(11)= dx*vx + dy*vy + dz*vz - 2*vxp
   F(12)= ex*vx + ey*vy + ez*vz

! Scale
   F = F * PAR(11)
```

Figure 5.52 Added lines for the BVP in the eigenfunction computation.

the problem is set up like a BVP with the integration time ($PAR(11)$) as main continuation parameter. The boundary conditions are in figure (5.55). The strategy followed has been to set the constant UZR as:

$$UZR = \{21 : [-0.5, -0.25, 0.0, 0.25, 0.5]\},$$

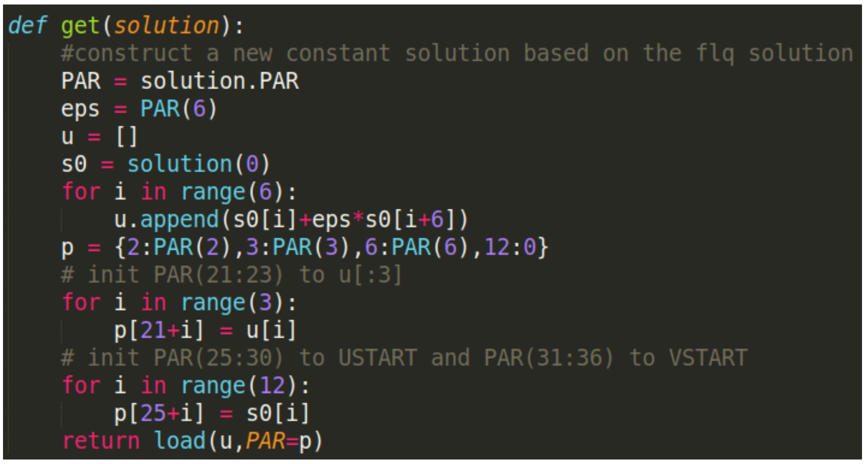
and then choose which one of the solutions is the one used to compute the manifolds. Notice that they are all the same orbit but with different x coordinate at the end of the integration.

The next and last step is to continue the orbits inside the manifold by changing the value of ε. For this problem the equations file is the same as in the time integration problem, and the constants file differs primarily in the ICP vector, where now $PAR(6)$ (ε) appears first and $PAR(21)$ can't be seen, because now it isn't a free parameter (we want the rest of the orbits in the manifold to end in the same plane).

The result of applying this series of calculations to the aforementioned orbit in the $\mathbf{H}1$ family is in figure (5.56). A few comments are necessary for this solution. By looking at figure (5.44) we can see that in the $\mathbf{H}1$ family and for more than 20000 Km the stability index is positive and, as stated before, for this orbit in particular $PAR(4) = 10.59$, which is a relatively big Floquet multiplier. This is important, as the chosen absolute value of ε_1 will determine the divergence speed of the state in the linear approximation of the manifold. Because here the value of $PAR(4)$ is big enough, we could choose a small value of $|\varepsilon_1|$, and the time integration will rapidly converge in the orbit that crosses the plane with $x = 0$, which is the stopping condition. In this case we have chosen $\varepsilon_1 = -10^{-5}$, which will give a very good approximation of the manifold, as ε indicates the distance in the direction of the eigenfunction (unit in modulus by construction) at which the state of the manifold is approximated by $\mathbf{r}(0) = \mathbf{u}(0) + \varepsilon\mathbf{w}(0)$, and the non-dimensional distance of 10^{-5} corresponds to 4 Km approximately (very small in this context), the resulting orbit after the time integration is in figure (5.57). Had we chosen a value of ε too small and AUTO wouldn't have been able to complete the integration to the plane $x = 0$, the reason for this is now explained. For this example the $NTST$ constant has been set to 200, which means that every solution in this continuation problem is made of 200 mesh intervals in the discretization, with 4 colocation points in them ($NCOL = 4$). With this set, AUTO can only *stretch out*

```
 SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
!---------- ----

   IMPLICIT NONE
   INTEGER, INTENT(IN) :: NDIM, ICP(*), NBC, IJAC
   DOUBLE PRECISION, INTENT(IN) :: PAR(*), U0(NDIM), U1(NDIM)
   DOUBLE PRECISION, INTENT(OUT) :: FB(NBC), DBC(NBC,*)

   DOUBLE PRECISION fm

! Periodicity boundary conditions
   FB(1:6) = U0(1:6) - U1(1:6)

! Floquet eigenvalue/vector boundary relation
   fm = PAR(4)
   FB(7:12) = U1(7:12) - fm*U0(7:12)

 END SUBROUTINE BCND
!----------------------------------------------------------------------

 SUBROUTINE ICND(NDIM,PAR,ICP,NINT,U,UOLD,UDOT,UPOLD,FI,IJAC,DINT)
!---------- ----

   IMPLICIT NONE
   INTEGER, INTENT(IN) :: NDIM, ICP(*), NINT, IJAC
   DOUBLE PRECISION, INTENT(IN) :: PAR(*)
   DOUBLE PRECISION, INTENT(IN) :: U(NDIM), UOLD(NDIM), UDOT(NDIM), UPOLD(NDIM)
   DOUBLE PRECISION, INTENT(OUT) :: FI(NINT), DINT(NINT,*)

! Integral phase condition
   FI(1) = DOT_PRODUCT(U(1:6),UPOLD(1:6))

! Integral Floquet eigenfunction normalization
   FI(2) = -PAR(5)+DOT_PRODUCT(U(7:12),U(7:12))

 END SUBROUTINE ICND
```

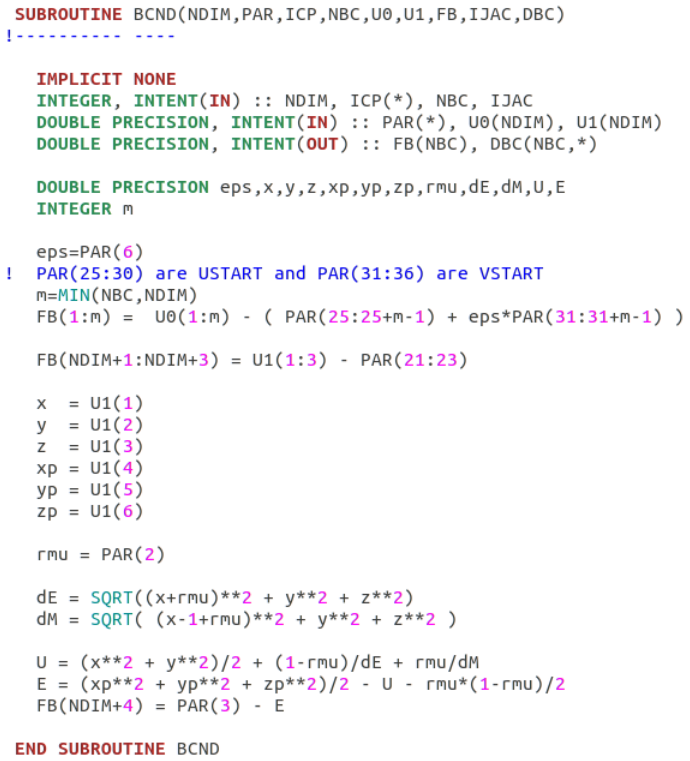
Figure 5.53 Boundary and integral conditions for the BVP in the eigenfunction calculation.



```
def get(solution):
    #construct a new constant solution based on the flq solution
    PAR = solution.PAR
    eps = PAR(6)
    u = []
    s0 = solution(0)
    for i in range(6):
        u.append(s0[i]+eps*s0[i+6])
    p = {2:PAR(2),3:PAR(3),6:PAR(6),12:0}
    # init PAR(21:23) to u[:3]
    for i in range(3):
        p[21+i] = u[i]
    # init PAR(25:30) to USTART and PAR(31:36) to VSTART
    for i in range(12):
        p[25+i] = s0[i]
    return load(u,PAR=p)
```

Figure 5.54 Python code to compute the linear approximation of the unstable manifold.

the orbit up to a point while maintaining the tolerances. So if the initial point in the manifold is set so the integration will stay close to the orbit for a very long time, then it won't be possible to complete the orbit to the desired plane. The only way to achieve both things at the same time (extremely good approximation of the manifold and integration to the desired plane) would be to increment the discretization, which will slow down the computation significantly, and will make it impractical. This whole consideration is important when the chosen orbit has a smaller Floquet multiplier, as the chosen value of ε_1 will have to match both aspects of the problem. Another important aspect about ε_1 that hasn't been addressed is the sign, because if it is chosen incorrectly the state $\mathbf{r}(0) = \mathbf{u}(0) + \varepsilon\mathbf{w}(0)$ won't be on the manifold at all. In this particular example the correct sign is negative.



```
 SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
!---------- ----

   IMPLICIT NONE
   INTEGER, INTENT(IN) :: NDIM, ICP(*), NBC, IJAC
   DOUBLE PRECISION, INTENT(IN) :: PAR(*), U0(NDIM), U1(NDIM)
   DOUBLE PRECISION, INTENT(OUT) :: FB(NBC), DBC(NBC,*)

   DOUBLE PRECISION eps,x,y,z,xp,yp,zp,rmu,dE,dM,U,E
   INTEGER m

   eps=PAR(6)
!  PAR(25:30) are USTART and PAR(31:36) are VSTART
   m=MIN(NBC,NDIM)
   FB(1:m) =  U0(1:m) - ( PAR(25:25+m-1) + eps*PAR(31:31+m-1) )

   FB(NDIM+1:NDIM+3) = U1(1:3) - PAR(21:23)

   x  = U1(1)
   y  = U1(2)
   z  = U1(3)
   xp = U1(4)
   yp = U1(5)
   zp = U1(6)

   rmu = PAR(2)

   dE = SQRT((x+rmu)**2 + y**2 + z**2)
   dM = SQRT( (x-1+rmu)**2 + y**2 + z**2 )

   U = (x**2 + y**2)/2 + (1-rmu)/dE + rmu/dM
   E = (xp**2 + yp**2 + zp**2)/2 - U - rmu*(1-rmu)/2
   FB(NDIM+4) = PAR(3) - E

 END SUBROUTINE BCND
```

Figure 5.55 Boundary conditions in the time integration continuation problem and manifold computation problem.

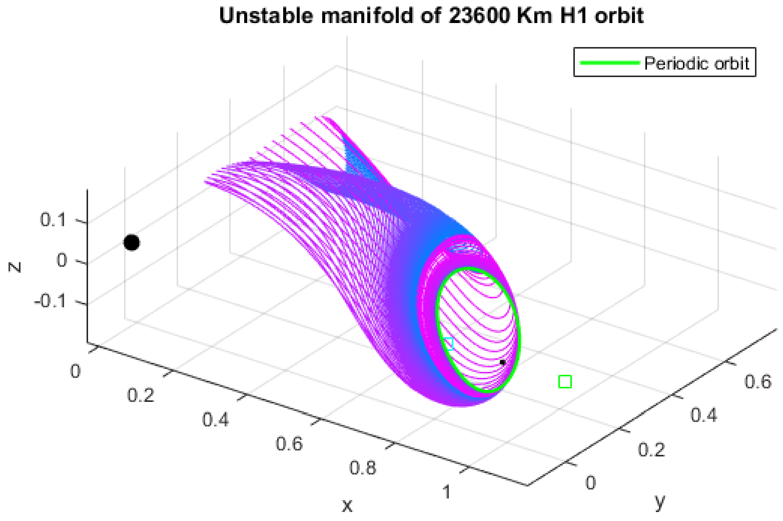


Figure 5.56 Unstable manifold computed for an orbit in the **H**1 orbital family.

To illustrate the importance of the absolute value of $PAR(4)$ in the election of ε_1 the next example of unstable manifold is that of the orbit with 18800 Km as the closest distance to the Moon (it is at the limit of the change in stability). In this case $PAR(4) \approx 2$, so even by perturbing the state in the direction of the unstable manifold the divergence will be slow. The problem of convergence commented before is now very relevant, and because of that the chosen value of the perturbation is $\varepsilon_1 = -0.025$, which in reality gives a poor linear approximation of the manifold. Nevertheless, the result in figure (5.58) shows that even with a relatively bad approximation of the manifold it is possible to compute it.

A final comment about the last step of the computation (continue the manifold from the first orbit) is that

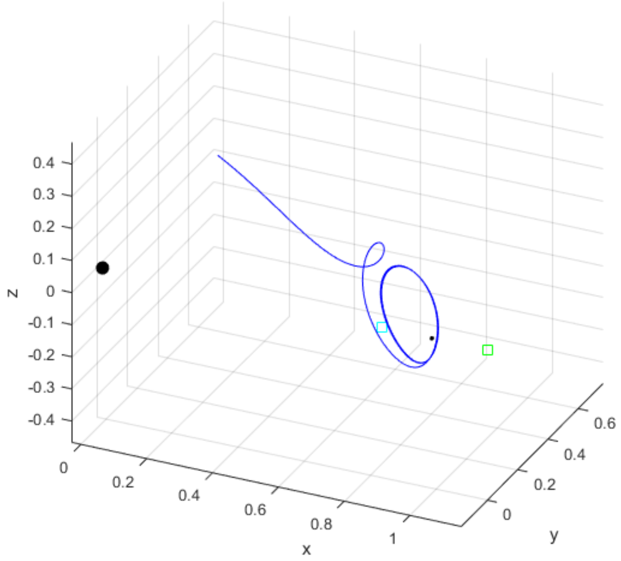


Figure 5.57 Initial orbit in the manifold of the 23600 *Km* **H**1 orbit.

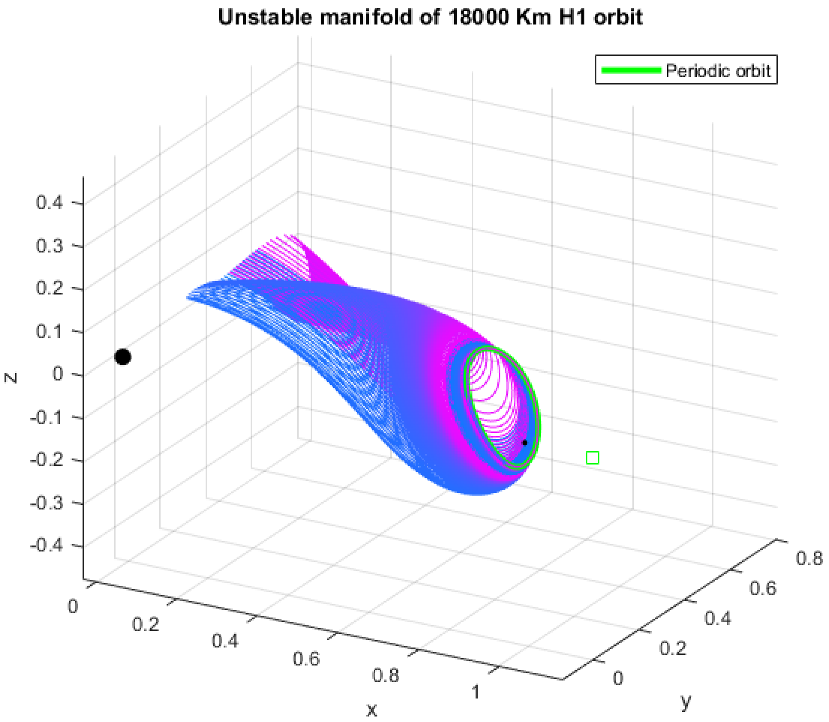


Figure 5.58 Unstable manifold computed for a quasi-stable orbit in the **H**1 orbital family.

the direction of continuation, positive or negative *DS*, is irrelevant as long as the absolute value is small enough to swept all of it.

5.7.1 Manifolds as the way in and out of NRHO's

Until now only the unstable manifolds had been shown as the structure that results from **forward** time integration of the perturbed state in the direction of the eigenfunction corresponding to the unstable Floquet multiplier (absolute value greater than one). This dynamical structure is relevant in the sense that can be used as a cheap way out of the periodic orbit to the *vicinity* of planet Earth, because only a small ΔV is needed to access it (we are using here the inherent properties of the periodic orbit as a tool for escaping). The exact position in the $x = 0$ plane where the unstable manifold leads to depends on the point around the periodic orbit that is used as entrance to the manifold, in figure (5.59) this relation is shown for the 23600 Km orbit.

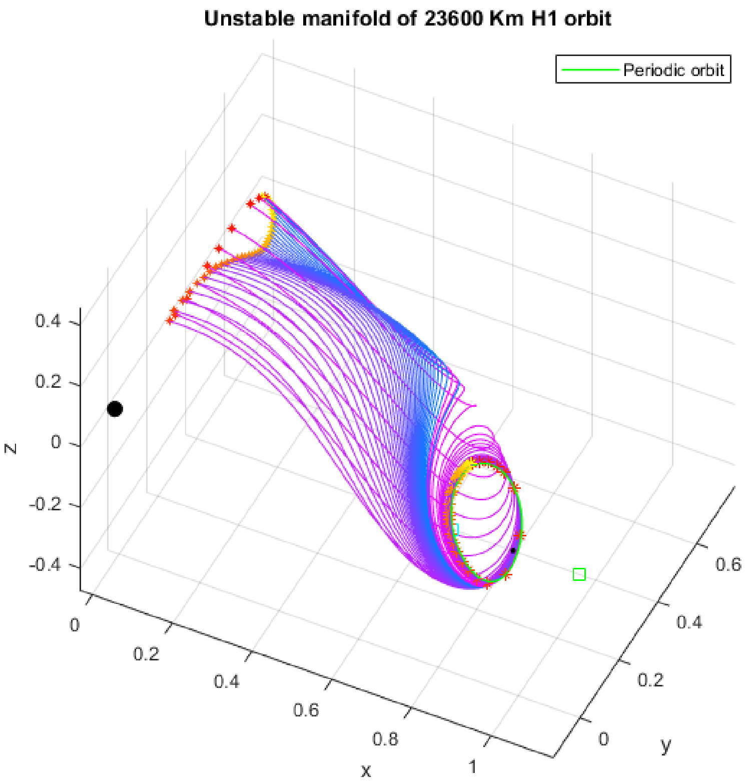


Figure 5.59 Unstable manifold of the 23600 Km orbit, **H**1 family, with the relation between the entrance point to the manifold and the destination in the $x = 0$ plane.

On the other hand, each unstable Floquet multiplier has its corresponding stable one, which is the inverse of it. With the same code that has been shown to compute the unstable manifold, the stable manifold can be calculated by introducing the value of the stable Floquet multiplier in the function of the first step as the fourth input (see figure (5.51)). This will create the initial solution for the calculation of the eigenfunction, but with the corresponding value of Floquet multiplier, resulting in the eigenfunction that approximates the stable manifold. There is another change, very simple but with an important consequence to the nature of the time integration step. If nothing else is changed, a forward time integration of the perturbed state in the direction of the stable manifold will lead the particle to the periodic orbit again (as it is the nature of that particular direction). The necessary change is then to set a negative value of DS, in order to perform **backwards** time integration, and stop it in the needed moment just like before. The resulting trajectory has the first point close to the periodic orbit, and the last point in the plane used as stopping condition, though this last point is the one that occurs first in time. The result of this calculation, compared to that of the unstable manifold, is in figure (5.60). As can be seen, the stable manifold is symmetric with respect to the unstable one. This structure has the opposite utility, as a spacecraft placed at the initial point (in time) of the manifold will ultimately arrive at the periodic orbit without use of any more propellant (aside from course corrections).

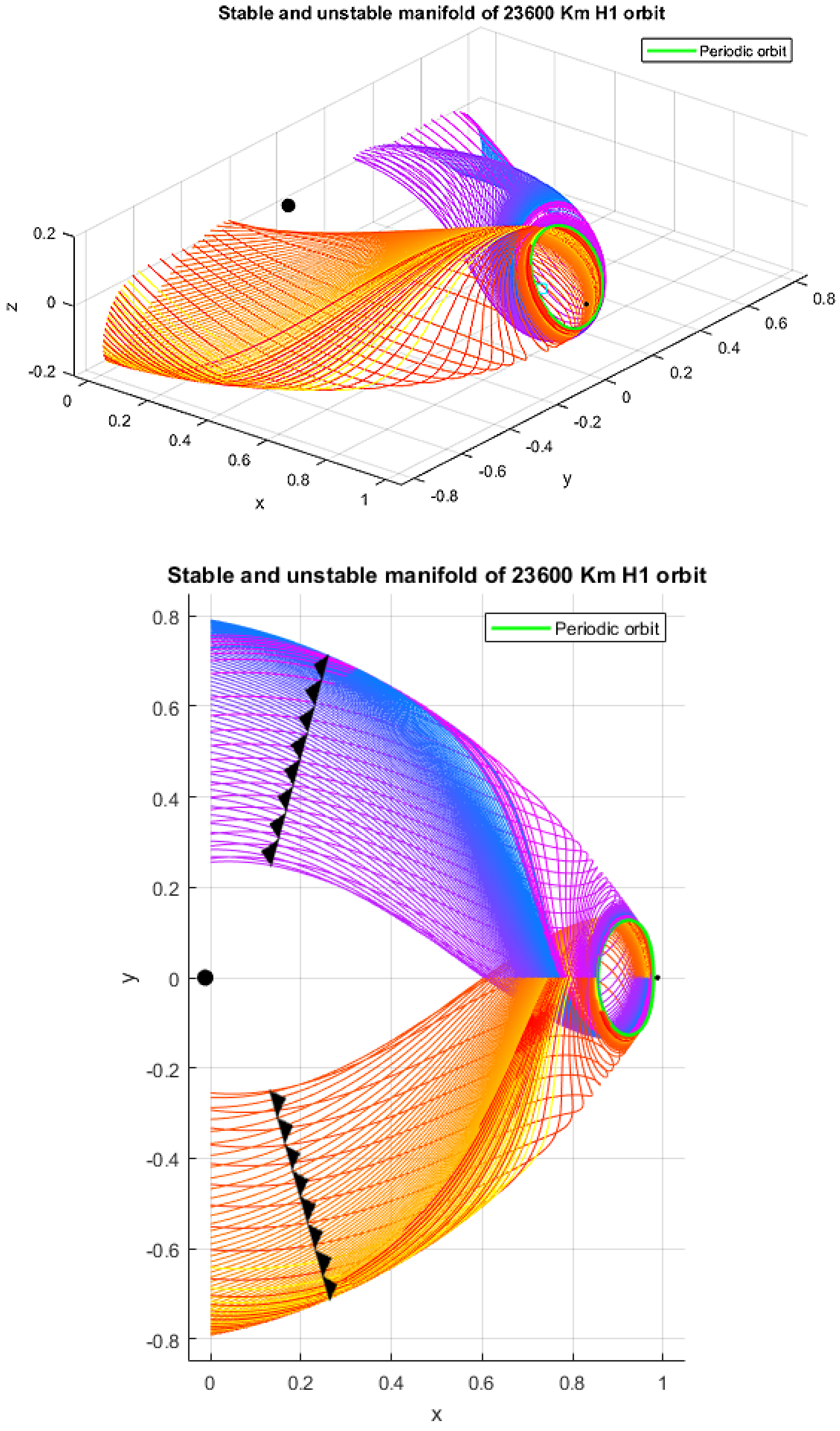


Figure 5.60 Stable and unstable manifold of the 23600 Km orbit, **H**1 family.

Another relevant aspect is the time required to access the orbit using this method. In figure (5.61) we can see the orbit that starts closer to the Earth from those in the stable manifold of figure (5.60), which is about 80000 Km from the surface. The dots are placed one day of travel apart from each other, meaning that 10 days pass before the spacecraft reaches the top of the orbit for the first time (this pass is still far from the orbit itself), a rather slow method to reach the orbit. A possible shortcut for this will be shown later on, though it is even worse for the orbits in the **H**2 family, as it is shown now.

The process used until now has also been applied to an orbit in the **H**2 family with the closest distance to the Moon of about 20000 Km, which has $PAR(4) \approx 10$, positive and relatively big, just as before for the

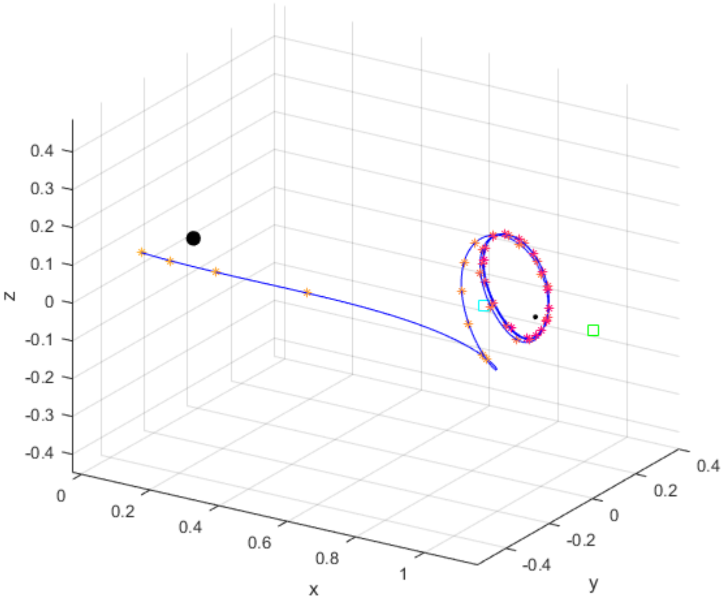


Figure 5.61 Orbit in the stable manifold of the 23600 Km orbit with the days represented as dots .

H1 family. The result is in figure (5.62), where it is obvious that now the dynamic of both the manifolds is quite different from the previous cases. Now, the manifolds grow from *behind* the orbit (as seen from the Earth) and turn around until they face the $x = 0$ plane very far from the Earth's surface (the closest orbit ends farthest than the Moon itself).

The next case is the first example of orbit with negative $PAR(4)$, it is the 6000 Km orbit of the **H**2 family (this is inside the NRHO family as defined before). The only difference is the sign of the Floquet multiplier, but this has a noticeable effect on the shape of the manifold's surface. The result for this orbit is in figure (5.63), where it is evident that now the surface is twisted rather than orientable.

5.7.2 An alternative path to the NRHOs

The alternative is explained in [20], and it consists in finding the trajectory that starts at the parking orbit around the Earth and ends in the insertion point in the stable manifold that ultimately leads to the periodic orbit. In order to do that one possible method is to fix the insertion point in the manifold and search for the impulse that, by backwards time integration, makes the trajectory start at the desired orbit (defined by its altitude, but with the inclination free). In particular, this work establish the insertion point in close proximity to the Moon so that the the spacecraft takes advantage of the gravity pull.

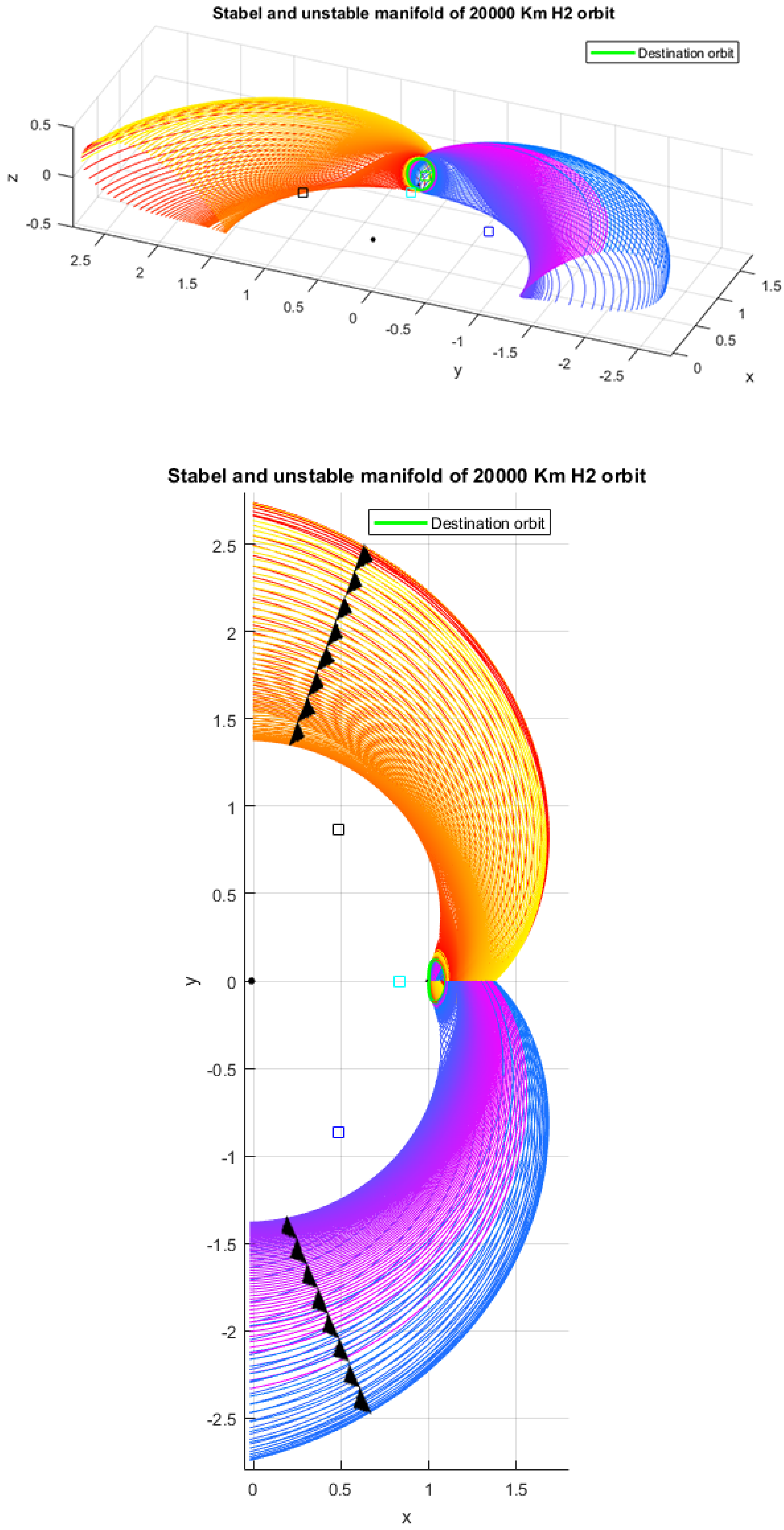


Figure 5.62 Stable and unstable manifold of the 20000 Km orbit, **H**2 family.

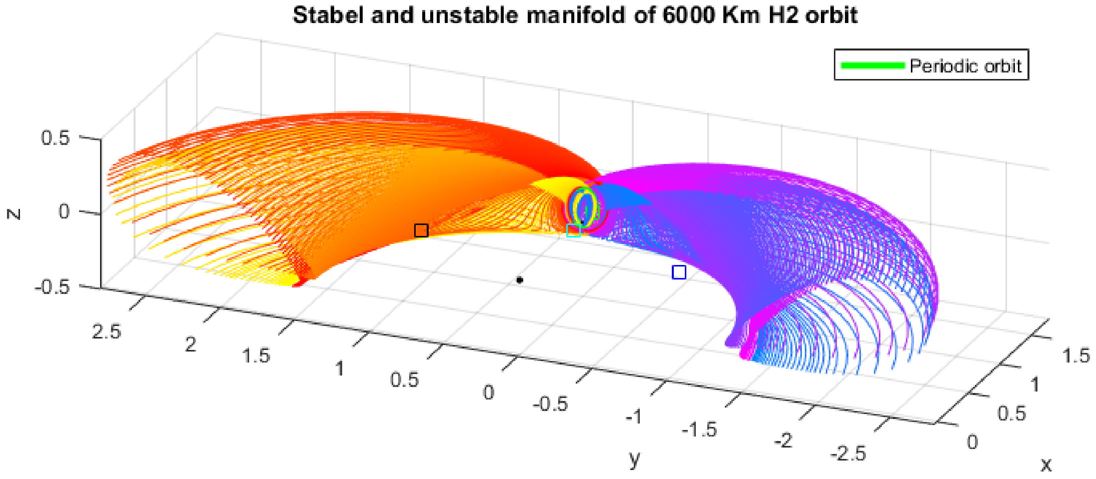


Figure 5.63 Manifolds in the 6000 *Km* orbit of the **H**2 family.

6 Conclusions and future work

The original goal of showing how AUTO works and applying it in the context of studying the periodic motion in the CR3BP has been accomplished. The main important concepts about non-linear dynamics and bifurcations have been presented. This has been complemented whit a brief overview of the way continuation is numerically performed inside the AUTO environment for the problem types at hand, which has set the groundwork for the next chapter, the guide on the AUTO usage for the context of this work. AUTO has then been used as a tool for studying the Circular restricted three body problem. This way, it has been relatively easy to study the periodic motions that emerge from the Lagrange points which has ultimately lead to the study of the Halo orbits and its properties. Inside of this group of orbits the Near rectilinear halo orbits have been studied for the already proposed DSG project in some of the relevant aspects of this mission (stability, departure times, synodic resonances, eclipsing...). In relation to the last, the manifolds associated to these orbits have been calculated using AUTO as well, with some commentaries about its potential for being the dynamical structure used as access to the cycles.

The AUTO software has presented itself as an extremely powerful and versatile software that can perform otherwise complex calculations in a very simple way from the user side, up to a point that the computation of periodic orbits can be done with a couple of Python code lines once the problem has been set up properly. This has allowed for a very complete study of the dynamical properties of the cycles that emanate from the equilibria in the CR3BP, as well as a more in depth study of the Halo orbits that are suppose to be the future place for the Deep Space Gateway, all of it with detailed descriptions in how the computations have been performed. On the other hand, this study has the drawback of being a simplification of the real dynamical environment that the DSG will face, which includes the next added difficulties: The motion of the Moon around the Earth has different from zero, albeit small, eccentricity, which changes the problem type from autonomous (easily studied with AUTO) to non-autonomous. Although it is technically possible to study time dependent problems with AUTO it adds difficulty in the sense that it would be necessary to include an extra equation for the time (now a state variable), and the study of periodic motion has to be explicitly set as a BVP of the type $IPS = 4$. It could then be faced as a continuation problem with the orbit for $e = 0$ as initial iteration. The second difficulty to be addressed is that of the other perturbations that are potentially critical in long term orbits, such as the Sun's or even Jupiter's. This could be studied separately whit higher fidelity models on the orbits that have already been computed to see the consequences of this simplification.

Aside from the study that has been conducted here it could be interesting to go deeper in the understanding on how to access the NRHOs and specifically how to perform rendezvous with a spacecraft already in it.

List of Figures

Bibliography

[1] Henri Poincaré and R Magini. Les méthodes nouvelles de la mécanique céleste. *Il Nuovo Cimento (1895-1900)*, 10(1):128–130, 1899.

[2] Hanspeter Schaub and John L Junkins. *Analytical mechanics of space systems*. Aiaa, 2003.

[3] Deep space gateway to open opportunities for distant destinations. *https://www.nasa.gov/feature/deep-space-gateway-to-open-opportunities-for-distant-destinations*. Accessed: 22-05-2018.

[4] Stanford encyclopedia of philosophy: Henri poincaré. *https://plato.stanford.edu/entries/poincare/#Phy*. Accessed: 02-04-2018.

[5] Christos Skiadas. *The Foundations of Chaos Revisited: From Poincaré to Recent Advancements*. Springer, 2016.

[6] Robert W Farquhar. The utilization of halo orbits in advanced lunar operations. 1971.

[7] Ryan Whitley, Roland Martinez, Gerald Condon, Jacob Williams, David Lee, Diane Davis, Gregg Barton, Sagar Bhatt, Jiann-Woei Jang, Fred Clark, et al. Cislunar near rectilinear halo orbit for human space exploration. 2016.

[8] Yuri A Kuznetsov. *Elements of applied bifurcation theory*, volume 112. Springer Science & Business Media, 2013.

[9] Steven H Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. CRC Press, 2018.

[10] W Koon. Poincaré map, floquet theory, and stability of periodic orbits. Technical report, Technical report, Control and Dynamical Systems: California Institute of Technology, 2006. 46, 47, 2006.

[11] Eusebius J Doedel, BW Kooi, GAK Van Voorn, and Yu A Kuznetsov. Continuation of connecting orbits in 3d-odes (i): Point-to-cycle connections. *International Journal of Bifurcation and Chaos*, 18(07):1889–1903, 2008.

[12] HM Osinga and J Galan-Vioque B Krauskopf. Numerical continuation methods for dynamical systems, 2007.

[13] Uri Ascher, J Christiansen, and Robert D Russell. Collocation software for boundary-value odes. *ACM Transactions on Mathematical Software (TOMS)*, 7(2):209–222, 1981.

[14] Carl De Boor and Blâir Swartz. Collocation at gaussian points. *SIAM Journal on Numerical Analysis*, 10(4):582–606, 1973.

[15] Renato C Calleja, Eusebius J Doedel, Antony R Humphries, A Lemus-Rodríguez, and EB Oldeman. Boundary-value problem formulations for computing invariant manifolds and connecting orbits in the circular restricted three body problem. *Celestial Mechanics and Dynamical Astronomy*, 114(1-2):77–106, 2012.

[16] Auto-07p : Continuation and bifurcation software for ordinary differential equations. *http://www.dam.brown.edu/people/sandsted/auto/auto07p.pdf*. Accessed: 25-05-2018.

[17] Eusebius J Doedel, Volodymyr A Romanov, Randy C Paffenroth, Herbert B Keller, Donald J Dichmann, Jorge Galán-Vioque, and André Vanderbauwhede. Elemental periodic orbits associated with the libration points in the circular restricted 3-body problem. *International Journal of Bifurcation and Chaos*, 17(08):2625–2677, 2007.

[18] Eusebius Doedel. Numerical analysis of nonlinear equations. *Lecture Notes. Concordia Univ., Montreal*, 2010.

[19] Emily M Zimovan, Kathleen C Howell, and Diane C Davis. Near rectilinear halo orbits and their application in cis-lunar space. In *3rd IAA Conference on Dynamics and Control of Space Systems, People's Friendship University of Russia,(RUDN University), Moscow, Russia*, 2017.

[20] Dawn Perry Gordon. Transfers to earth-moon l2 halo orbits using lunar proximity and invariant manifolds. *Purdue University, August*, 2008.