# STUDY REFINEMENT OF SEMI-ANALYTICAL HALO ORBIT THEORY

## EXECUTIVE SUMMARY

**ESOC CONTRACT NO.:** 8625/89/D/MD(SC)

**ESOC TECHNICAL SUPERVISOR:** J. Rodriguez-Canabal

**AUTHORS:** G. Gómez [1], A. Jorba [2], J. Masdemont [2], C. Simó [1]

[1] Departament de Matemàtica Aplicada i Anàlisi, Facultat de Matemàtiques, Universitat de Barcelona, Gran Via 585, 08075 Barcelona, Spain.

[2] Departament de Matemàtica Aplicada I, ETSEIB, Universitat Politècnica de Catalunya, Diagonal 647, 08028 Barcelona, Spain.

**COMPANY:** Fundació Bosch i Gimpera, Balmes 21, 08007 Barcelona, Spain

EUROPEAN SPACE AGENCY. CONTRACT REPORT
The work described in this report is done under ESA contract. Responsability for the contents resides in the authors that prepared it

Barcelona, April, 1991

# Contents

# Introduction

In this work we study several problems related to the mission analysis of future planned or possible spacecraft missions. The work is divided in four Chapters, each one dealing with one of the following topics:

a) The first Chapter is devoted to the computation of quasiperiodic solutions for the motion of a spacecraft near the equilateral points of the Earth-Moon system. The motion of the bodies of the Solar System is assumed to be as close as possible to the real one. To this end the ephemeris for the actual computations have been taken from the JPL tapes.

As the problem of computing quasiperiodic solutions is numerically badly defined, the following methodology has been used:

a.1) First we develop an analytic model for the equations of motion near the equilateral points. This is done by a Fourier analysis of the full equations of motion where the ephemeris of the solar system bodies (in that case Earth, Moon and Sun are the relevant ones) are given by available analytical expressions.

a.2) Then we look for an analytical quasiperiodic solution with the basic frequencies present in the equations (that is: the mean elongation of the Sun, the mean motions of the Earth and the Moon and the mean motions of the lunar perigeon and node). This is done by using a suitable algebraic manipulator. Furthermore, we require that if in the analytical model we neglect the terms which are not in the RTBP model then we should find the equilateral relative equilibrium solution.

a.3) When the analytical approximation is available, we use a parallel shooting algorithm to obtain a numerical solution of the full equations of motion using ephemeris given by the JPL model. The end points of a given time interval are set equal to the ones found in the analytical solution. When the iterative process is finished we have a true orbit for the spacecraft close to the analytical quasiperiodic solution. As a complementary result the stability properties for this orbit are obtained.

b) The second Chapter gives a complete description of the orbits near the collinear point, $L_1$, between the Earth and the Sun in the RTBP model It is known that the vicinity of $L_1$ is an ideal place for the location of a solar observatory. This was the task of the ISEE-C spacecraft and will be the one of the SOHO one in the next future. Both have a halo orbit as nominal orbit. Halo orbits for the RTBP have been already studied. Furthermore, the perturbations produced by the non circular motion of Earth and Sun, the Moon and, to a lesser extend, the planets, are moderate. Hence one can use a combination of algebraic manipulation and parallel shooting, as described in Chapter 1, to obtain a numerical quasiperiodic solution using the JPL Ephemeris. Going back to the RTBP, which is the essential part of the problem, the following questions are addressed:

  b.1) The halo periodic orbits (in the synodic reference system) are not the only interesting orbits near $L_1$. There are also the quasiperiodic (eventually periodic) Lissajous orbits, lying on invariant tori. The halo orbits have the drawback of requiring a minimum amplitude in the planar direction orthogonal to the Earth-Sun line. This minimum amplitude is quite large. Other orbits confined to a lesser angular distance from the Sun (as seen from the Earth) are the Lissajous orbits. However they cross in front of the solar disc during their evolution. To assess the possibilities and requirements if Lissajous orbits are used, we have computed this two-dimensional family of orbits analytically at a very high order by using a suitable algebraic manipulator. The possible orbits and the advantages/disadvantages are discussed.

  b.2) All the orbits in the vicinity of $L_1$ are highly unstable. This is the reason why one should use a careful station keeping to be studied in Chapter 3. This prevents also to obtain by direct numerical simulation a full description of the vicinity of $L_1$. Furthermore, due to the 1 to 1 resonance giving rise to the halo orbits, the Lissajous orbits have been obtained in b.1) in a domain not reaching the halo orbits. To have a full picture of the possible motions we have to skip the unstable part of the motion. This is done by using the so called reduction to the central manifold (again via symbolic manipulation). It results a Hamiltonian system with two degrees of freedom which has been then numerically integrated and fully described. Beyond the Lissajous orbits there are also quasiperiodic orbits around the halo orbits, which can also be useful as nominal orbits.

c) In Chapter 3 methods to compute the nominal orbit and to design and test the control strategy for the quasiperiodic halo orbits are developed. Now we return to the full solar system model given by the JPL ephemeris including the solar radiation pressure. The nominal orbit considered is a quasiperiodic perturbation of a periodic orbit of the RTBP. As it is unstable it requires a station keeping which takes into account the behaviour near the nominal orbit. In the full process we distinguish several steps:

c.1) To obtain the one parameter family of halo orbits for the RTBP. This was already done in a previous work for the ESA [2] by using algebraic manipulation. The variational equations along the halo orbit were also integrated analytically by suitable series expansions.

c.2) The halo orbit has to be modified to include all the quasiperiodic perturbations to be added to the RTBP. This can be done, as in a), by obtaining analytically some terms and then by improving the results using a parallel shooting with JPL Ephemeris, or performing directly the parallel shooting (with the eventual use of a continuation parameter to ensure convergence of the method). The first approach has been taken, using some analytical quasiperiodic terms as found in [2]. The numerical refinement has been improved to cover long time periods. The orbit obtained depends on a vertical amplitude and an initial epoch.

c.3) The escape from the vicinity of the nominal halo orbit takes place along the unstable direction. When the tracking allows us to assure that such an escaping is taking place a manoeuvre should be done. The direction and the amount of this manoeuvre are easily computed if the solution of the variational equations along the nominal orbit are available. We follow again the methodology of [2]. The (x, y) manoeuvres are the ones with a larger gain and we have restricted to this case.

c.4) To test the strategy of the station keeping for a nominal halo orbit, several runs have been done. In those runs we have taken different magnitudes for the distribution of errors in tracking, execution of manoeuvres and effect of solar radiation pressure. We have also checked the effect of the changes in the nominal orbit and in the criteria to decide when to do a manoeuvre (or possible delays in the execution). The results are excellent (roughly only a 2.5% of the real amount of $\Delta v$ used for the ISEE-C is enough) and very robust. Furthermore the different runs allow to measure the effect of each one of the sources of error.

d) In Chapter 4 the transfer from the Earth to a halo orbit is studied. This problem can be seen in the general case as the one of reaching a nominal orbit by suitable manoeuvres. Then an optimization process is started (from an initial guess or approximated solution) to minimize the fuel consumption under some scientific or technical constraints. Typically one should do a big manoeuvre to inject the spacecraft in a transfer orbit, a mid course correction manoeuvre to correct for the errors of the previous one, and a final manoeuvre to insert into the nominal orbit. Our approach has been different. The target orbit has a stable manifold. It turns out that this manifold comes close to the Earth (without further passages near the Earth) at some times and for suitable initial epochs of the halo orbit. The work is divided as follows:

d.1) Given a vertical amplitude and an initial epoch the halo orbit is obtained as in c). Then the local stable manifold is approximated (at a given time) by integration of the variational equations obtaining the stable direction.

d.2) The manifold is globalized by backwards numerical integration (using the JPL ephemeris, of course). The orbits in the stable manifold leading to close approaches to the Earth are stored.

d.3) Additional computations of orbits in the stable manifold are done in order to have a finer mesh. Then the orbits suitable for the transfer are selected. To this goal one should skip orbits passing too close to the Moon, or reaching the vicinity of the Earth with a too low or a too high perigee distance or with a non allowable equatorial inclination. In this way one has the selection of the transfer possibilities and the amount of the manoeuvre to inject on the stable manifold. The mid course corrections to prevent from injection errors are unavoidable, but with this approach one can skip the injection manoeuvre in the nominal orbit.

We should like to conclude this Introduction by making some comments on the methodology. The general point of view is the Dynamical Systems approach, mainly Hamiltonian Systems with time-depending perturbations. The semianalytical approach (i.e. the use of algebraic manipulators with numerical coefficients) has been carried out as far as possible to obtain information not only about one orbit but a big amount of them. This has been combined with refined numerical methods to get real solutions to real problems. Furthermore the methods are presented in such a way thats its use in a big variety of similar problems becomes easy.

# Chapter 1

# Quasiperiodic Solutions Near the Equilateral Points of the Earth-Moon System

## 1.1 Introduction

The main purpose of this Chapter is to find quasiperiodic solutions near the equilateral libration points of the Earth-Moon system.

The equations of motion near these points can be written as (see [3])

$$\begin{aligned}
\ddot{x} = P(7)&\left[-\frac{x-x_E}{r_{PE}^3}(1-\mu_M) - \frac{x+x_E}{r_{PM}^3}\mu_M - x_E(1-2\mu_M)\right] + \\
&+P(1)+P(2)x+P(3)y+P(4)z+P(5)\dot{x}+P(6)\dot{y}, \qquad (1.1)\\
\ddot{y} = P(7)&\left[-\frac{y-y_E}{r_{PE}^3}(1-\mu_M) - \frac{y-y_E}{r_{PM}^3}\mu_M - y_E\right] + P(8)+P(9)x+ \\
&+P(10)y+P(11)z+P(12)\dot{x}+P(13)\dot{y}+P(14)\dot{z}, \qquad (1.2)\\
\ddot{z} = P(7)&\left[-\frac{z}{r_{PE}^3}(1-\mu_M) - \frac{z}{r_{PM}^3}\mu_M\right] + P(15)+P(16)x+ \\
&+P(17)y+P(18)z+P(19)\dot{y}+P(20)\dot{z}, \qquad (1.3)
\end{aligned}$$

where $r_{PE}$, $r_{PM}$ denote the distances from the particle to the Earth and Moon, respectively, given by $r_{PE}^2 = (x-x_E)^2+(y-y_E)^2+z^2$, $r_{PM}^2 = (x+x_E)^2+(y-y_E)^2+z^2$. We recall that $x_E = -1/2$, $y_E = -\sqrt{3}/2$ for $L_4$ and $x_E = -1/2$, $y_E = \sqrt{3}/2$ for $L_5$.

The functions $P(i)$ that appear in the equations are of the form:

$$P(i) = A_{i,0} + \sum_{j=1}^{m} A_{i,j}\cos\theta_j + \sum_{j=1}^{m} B_{i,j}\sin\theta_j,$$

with $\theta_j = \nu_j t + \varphi_j$, where $t$ denotes the normalized time. Values for all of the coefficients, frequencies and phases can be found in Appendix C of [4].

An expansion of the nonlinear terms of the equations of motion, in power series of the coordinates $(x,\ y,\ z)$, can be done easily using Legendre polynomials as it is described in 1.1.1.

Another possibility is to produce those terms by means of a recurrence based on the one of the Legendre polynomials.

### 1.1.1 Expansion of the Equations

We consider the potential $V = \frac{1-\mu}{r_{PE}} + \frac{\mu}{r_{PM}}$. Let $\psi$ be the angle between the vectors $(x_E, y_E, 0)$ and $(x, y, z)$ and $\rho^2$ be $x^2+y^2+z^2$. Then

$$\frac{1}{r_{PE}} = \frac{1}{\sqrt{1-2\rho\cos\psi+\rho^2}} = \sum_{n=0}^{\infty} \rho^n P_n(\cos\psi), \tag{1.4}$$

where $P_n$ is the Legendre polynomial of degree $n$:

$$P_n(\omega) = \frac{1}{2^n n!}\frac{d^n}{d\omega^n}(\omega^2-1)^n = \sum_{k=0}^{\left[\frac{n}{2}\right]} \alpha_{n,k}\omega^{n-2k}.$$

Here $\left[\frac{n}{2}\right]$ denotes the integer part of $\frac{n}{2}$ and

$$\alpha_{n,k} = \frac{(-1)^k}{2^n n!}\binom{n}{k}\frac{(2n-2k)!}{(n-2k)!}.$$

Substituting these expressions in (1.4) one obtains

$$\begin{aligned} \frac{1}{r_{PE}} &= \sum_{n=0}^{\infty} \rho^n \sum_{n=0}^{\left[\frac{n}{2}\right]} \alpha_{n,k} \cos^{n-2k}\psi = \\ &= \sum_{n=0}^{\infty}\sum_{k=0}^{\left[\frac{n}{2}\right]} \alpha_{n,k}(xx_E+yy_E)^{n-2k}(x^2+y^2+z^2)^k. \end{aligned} \tag{1.5}$$

Finally, in order to obtain the expansion for $L_4$ (the $L_5$ case can be done with an identical process) we put $x_E = -\frac{1}{2}$, $y_E = -\frac{\sqrt{3}}{2}$ and (1.5) becomes

$$\frac{1}{r_{PE}} = \sum_{n=0}^{\infty}\sum_{k=0}^{\left[\frac{n}{2}\right]} \beta_{n,k}(x+\sqrt{3}y)^{n-2k}(x^2+y^2+z^2)^k, \tag{1.6}$$

where

$$\beta_{n,k} = \frac{(-1)^{n-k}(2n-2k)!}{4^{n-k}(n-k)!k!(n-2k)!}.$$

To perform the expansion (1.6) by algebraic manipulation it is better to change the order of the sums:

$$\frac{1}{r_{PE}} = \sum_{k=0}^{[\frac{N}{2}]}(x^2+y^2+z^2)^k \sum_{n=2k}^{N} \beta_{n,k}(x+\sqrt{3}y)^{n-2k},$$

where $N$ denotes the expansion order wanted. With a similar computation it can be found that:

$$\frac{1}{r_{PM}} = \sum_{k=0}^{[\frac{N}{2}]}(x^2+y^2+z^2)^k \sum_{n=2k}^{N} \beta_{n,k}(-x+\sqrt{3}y)^{n-2k}$$

and, finally,

$$\begin{aligned} V &= \frac{1-\mu}{r_{PE}} + \frac{\mu}{r_{PM}} = \\ &= \sum_{k=0}^{[\frac{N}{2}]}(x^2+y^2+z^2)^k \sum_{n=2k}^{N} \beta_{n,k}\left[(1-\mu)(x+\sqrt{3}y)^{n-2k}+\right. \\ &\qquad \left. + \mu(-x+\sqrt{3}y)^{n-2k}\right]. \end{aligned} \tag{1.7}$$

Now, using a program which takes advantage of the particularities of the latter expression, the expansions of $\frac{\partial V}{\partial x}$ , $\frac{\partial V}{\partial y}$ and $\frac{\partial V}{\partial z}$ are obtained. This program has been written to give the final solution up to order nine. This is enough for practical purposes, because the contribution of higher order terms is much less than the threshold used to select the coefficients $A_{i,j}$, $B_{i,j}$.Its routines are:

**Subroutine DERIV** Given a polynomial of three variables $x$, $y$ and $z$, this routine returns three polynomials which are its derivatives with respect to $x$, $y$ and $z$. This is used at the end of the computation of (1.7) to obtain the expansion wanted.

**Subroutine PRODE** This routine performs the product of two polynomials of three variables. It is used to multiply the term $(x^2+y^2+z^2)^k$ by the result of the inner summatory of (1.7).

**Subroutine XYZK** Given a polynomial $S$, this routine returns $S \times (x^2+y^2+z^2)$. It is used to obtain the different powers of $(x^2+y^2+z^2)$ starting from $S=1$.

**Subroutine PART2V** This routine computes the result of the inner summatory of (1.7), taking advantage of the cancellations that occur in this formula

**Subroutine INIT** This routine fills a real array with the powers of $\sqrt{3}$ (they will be used inside `PART2V`) and a couple of integer arrays containing information about the way to store polynomials (in fact this is not necessary, but it allows to increase the speed of the program).

**Main Program** This routine initializes the parameters used by the program ($x_E$, $y_E$ and $\mu$) and it uses the formula (1.7) to compute $V$. Then the final result is obtained (by derivation) and the result is written in the files EXPAN.1 (expansion of the first equation), EXPAN.2 (second one) and EXPAN.3 (third one).

## 1.2 Idea of the Resolution Method

The equations of motion can be written as

$$\begin{aligned} \ddot{x} &= f_1(t,x,y,z,\dot{x},\dot{y},\dot{z}), \\ \ddot{y} &= f_2(t,x,y,z,\dot{x},\dot{y},\dot{z}), \\ \ddot{z} &= f_3(t,x,y,z,\dot{x},\dot{y},\dot{z}). \end{aligned}$$

Define $G = (G_1, G_2, G_3)$, where $G_1 = f_1 - \ddot{x}$, $G_2 = f_2 - \ddot{y}$, and $G_3 = f_3 - \ddot{z}$. We are looking for quasiperiodic solutions of the form

$$\begin{aligned} x &= \sum x_k exp((k,\omega)t\sqrt{-1}), \\ y &= \sum y_k exp((k,\omega)t\sqrt{-1}), \\ z &= \sum z_k exp((k,\omega)t\sqrt{-1}), \end{aligned}$$

where $\omega = (\omega_1, \omega_2, ..., \omega_r)^t$ is a known set of basic frequencies, the ones that appear as basic frequencies, in the developments of the functions $P(i)$, and $(k,\omega)$ denotes the inner product of $k = (k_1, k_2, ..., k_r)^t$ and $\omega$. We consider $x$, $y$ and $z$ truncated up to some order. After substitution, $G$ may be considered as a function of $x_k$, $y_k$ and $z_k$, which means that looking for quasiperiodic solutions of this problem is more or less equivalent (this depends on the order of approximation taken for the trigonometric expansions of $x$, $y$, and $z$) to look for a zero of the function $G : \mathbf{R}^m \rightarrow \mathbf{R}^m$, where $m$ denotes the total number of coefficients. To solve this equation we use a Newton method. In order to calculate the Jacobian matrix, the chain rule is used in the following way: $\frac{\partial G}{\partial x_k} = \frac{\partial G}{\partial x} \cdot \frac{\partial x}{\partial x_k}$, and so on.

## 1.3 The Algebraic Manipulator

Now we are going to describe the algebraic manipulator specially built (in FORTRAN 77 language) to solve the latter problem. Here, a trigonometric series is stored as a vector of real numbers (each one corresponding to a coefficient), a vector of integers (of the same dimension) and an integer containing this dimension. This integer is only used in some operations to check an "overflow trigonometric series" condition. An auxiliary table, with the linear combinations of the basic frequencies, will also be used by the manipulator.

### 1.3.1 Storing Fourier Series

We need to store trigonometric series and handle them in a fast way. Moreover, we want the size of the vectors that contain them to be as short as possible. In order to get the latter condition, we have decided to store only the non-zero coefficients, positioned in a contiguous way (as a stack). This means that we must have a way to recognize those coefficients, because their position inside the vector does not allow to identify them.

For this reason we have taken an auxiliar integer table of frequencies which contains, for each column, different integer values. For example, if a column contains the integers $(k_1, \dots, k_r)$ it means that it refers to the frequency $k_1\omega_1 + \cdots + k_r\omega_r$, where $(\omega_1, \dots, \omega_r)$ is the vector of basic frequencies (obviously, the number of rows must be at least the number of frequencies). This table will be used in two different ways:

1. Given a position, to find the corresponding coefficients of the frequencies.

2. Given an integer vector, to search for its position inside the table.

The first use is easy and fast to do, but not the second one. At this point, it would be useful that elements of the table were given in some order, because we could use a binary search. One could think that it would be enough to fill the table at the beginning of the program using the selected order, but this does not work very well, due to the fact that when we start the program we do not know yet which frequencies will appear in the intermediate computations. We need to add frequencies to the table during the process, and we need to restart the order in a quick way. For this reason we have added to the table an integer vector which refers to the table in an ordered way and, when a new frequency is added (at the end of the table), the program only needs to modify this vector to get the table in order again. The order selected has been lexicographic:

$$(k_1^1, \dots, k_r^1) < (k_1^2, \dots, k_r^2) \Leftrightarrow \begin{cases} k_1^1 < k_1^2 & \text{if } k_1^1 \neq k_1^2 \\ (k_2^1, \dots, k_r^1) < (k_2^2, \dots, k_r^2) & \text{if } k_1^1 = k_1^2. \end{cases}$$

Now, let us see how we can use this. We suppose that the real vector of coefficients is called `x` and we define an integer vector called `nx`. The first component of `x` is always used to store the independent term (the average). The first component of `nx` is used to store the number of non-zero coefficients of the series (the last meaningful component of vector `x`). Now, let `j` be an integer less than or equal to `nx(1)` and we consider the coefficient `x(j)`. If `nx(j)` is even, `x(j)` corresponds to a cosinus, otherwise it corresponds to a sinus. In the column `m=nx(j)/2` (integer division) of the table described above we can find the frequencies corresponding to this coefficient.

Now we are going to see how all of these things have been programmed. In what refers to storing frequencies in the table, we restrict ourselves to the ones that have the first integer coefficient ($k_1$ using the above notation) greater than or equal to zero. If we have a frequency with $k_1 < 0$, we change its sign and, in the case that it corresponds to a sinus, the sign of the coefficient is also changed. If $k_1 = 0$ we impose that the latter condition holds for $k_2$. If $k_2 = 0$ we impose it to $k_3$, and so on.

All of these arrays have been included inside a common called taules:

```
common /taules/ taufre,ntau,ordtau,busc
```

taufre has been declared as integer*2 and ntau, ordtau and busc as integer*4. taufre($r$,$n$) contains the table of frequencies ($n$ is the maximum number of them and $r$ the number of basic frequencies), ntau is the actual number of frequencies in the table, ordtau($n$) is the vector that holds the order of taufre and busc(2,$m$) is an array used to get a quicker binary search. Let us see how it works. Given a frequency $(k_1, \ldots, k_r)$, $k_1 \geq 0$ the numbers busc(1,$k_1$+1) and busc(2,$k_1$+1) refer to the first and last place (of ordtau) where this $k_1$ appears, and then the search field is shorter. If there do not exist any frequencies with this $k_1$ the corresponding values of busc are $-1$.

Finally, we are going to comment on some points about another system of storing Fourier series used in some intermediate computations. As we will see later, in many operations the manipulator uses a real array as a workspace. This vector has twice more components that the table of frequencies and it is used to hold series in such a way that it allows to make sums faster. This is due to the fact that the position of a coefficient in this vector plays the same role that the index vector seen before. For example, with the same notation as before, the coefficient x(j) is put in the component nx(j) (this is only true for j $>$ 1, x(1) is put in the first component of the working vector). Note that if we have two vectors of this kind, the sum of them can be done by adding, component by component, the two vectors. The dissadvantage of having many vectors like this is the waste of memory that they require, because many of this components may be zero. For this reason we prefer to keep the Fourier series in the packed form seen before. We have only one working vector and we pass it to routines that make use of it.

### 1.3.2 Basic Subroutines

The following routines have been built with the aim of having an easy-to-use system to handle the tables of frequencies.

**Subroutine FREQ** The first parameter is an integer*4 variable coming from a Fourier series (for example, using the above notation, some value nx(k)) that contains information relative to a coefficient. Unchanged on exit. The second parameter is an `integer*4` array that will contain the frequencies corresponding to `n`, and the third parameter will contain 0 if it refers to a cosinus or 1 if it refers to a sinus.

**Subroutine POSI** This routine is inverse of the former one: it returns an integer which contains information about the frequency (following with the example, information to be stored in the vector `nx`). The first parameter is an `integer*4` vector of frequencies and the second one is an integer containing 0 or 1. These parameters are unchanged on exit. The routine returns an integer value `n` to be stored in an integer vector corresponding to a Fourier series. The routine performs a binary search of `fv` in the table of frequencies and, assuming that `fv` is found in the place `j`, makes `n=2*j+i`. If `fv` is not found (i. e., it is not in the table) the routine calls the subroutine `ACTUAL`.

**Subroutine ACTUAL** This routine adds a frequency to the table. The first parameter is the vector containing the frequency to be added and the second one (`lloc`) is the place (of `ordtau`) where this frequency must be (this value has been found during the binary search of the routine `POSI`). The elements which previously were stored in `ortau`, in places from `lloc` up to the end, are shifted one place to the right. Both parameters are unchanged on exit.

**Subroutine EMPAQ** As we have already seen, in many operations the manipulator uses a real array as a workspace. This routine packs a trigonometric series contained in the working vector.

**Subroutine NORMAL** The first parameter is an integer vector (`f`) containing a frequency, the second one is an integer saying how many frequencies we have. With this, the routine puts `f` in normal form (the first component non equal to zero must be positive) and returns `is=-1` if it has changed the sign of `f`, or `is=1` otherwise. If all of the components of `f` are equal to 0 it returns `is=0`. This routine is called during some operations (for example, a product of series).

**Subroutine WFREC** This routine returns the numerical value of the frequency corresponding to a place (given by means of a parameter) of the working vector.

**Subroutine INITAU** This routine initializes the tables used by the manipulator. It must be called at the beginning of the program, before making any other call to a routine of the manipulator. It has no parameters.

**Subroutine AVALUA** This routine computes the numerical value of a trigonometric series at a given time value `t`.

**Subroutine LLEGEIX** Given a channel (already opened), this routine reads a Fourier series through this channel and returns it in packed form.

**Subroutine ESCRIU** Given a channel (already opened), this routine writes a Fourier series given in packed form.

### 1.3.3 Making Operations

Now we are going to describe the main operations performed by the manipulator, which include sums, multiplications of Fourier series and multiplications of Fourier series by real numbers, differentiation and some special operations that we shall describe later. To keep only the meaningful terms of the series, we have included a threshold that allows to drop terms less (in absolute value) than this threshold, which is in a common named `control`.

**Subroutine SUMA** This routine performs the sum of two Fourier series. The method consists in expanding the first series inside the working array and then add the second series putting the results in the same working vector. Finally, the result is packed (this routine calls SUMAF).

**Subroutine SUMAF** Given a Fourier series in packed form and a Fourier series in expanded form, this routine adds them and puts the result in the same vector that contains the series in expanded form (the vector in expanded form is the working vector mentioned in section 1.3.1). This routine has been built to increase the speed of operations of type $a = a + b$.

**Subroutine PROD** This routine obtains the product of two series. The way to do this is to fill a working vector with zeros and to call the routine PRODF (see below). This routine returns the result inside the working vector and finally it is packed.

**Subroutine PRODF** Given two Fourier series in packed form (we can call them $a$ and $b$) and a Fourier series in expanded form called $w$, this routine performs the operation $w = w + a \times b$. To do this, each term of the first series is multiplied by each term of the second one. If the absolute value of the resultant coefficient is less than the threshold, it is dropped, otherwise the frequencies of the terms that are multiplying are computed. With this, the result is added to the corresponding position of the working array. As SUMAF, this routine has been built to increase the speed of some operations.

**Subroutine PR1C** This routine performs the product of a series containing only one term (whose coefficient is equal to one) by a Fourier series in packed form. The result is returned also in packed form. The unitary series is given by a number displaying the frequencies and if it is a sinus or a cosinus (this number is of the same kind as the components of the index vector (`nx`) seen before). The routine works with the same algorithm as `PROD`. This operation is needed a lot of times during the calculus of the Jacobian matrix. This routine can be avoided, but it is easy to write and allows to simplify the main algorithm (we do not need to fill a trigonometric array with an unitary series and to call `PROD`).

**Subroutine PR1CF** This routine makes the same operation as `PR1C`, but adding the result to the series contained in the working vector. This allows us to get more speed during some operations.

**Subroutine PRESC** This routine multiplies a real number by a trigonometric series. This is a very common operation in the main algorithm and we prefer to write it instead of using the routine `PROD`.

**Subroutine DF** This routine computes the derivative of a Fourier series as usual: for each term the integer vector $k$ containing the frequency is obtained, and the coefficient is multiplied by $(k,\omega)$ (the vector $\omega$ is stored in a common called `freque`) and if it is a cosinus, its sign is changed.

**Subroutine MDF2** This routine performs the second derivative of a trigonometric series, and multiplies it by $-1$. Note that it is faster to compute this directly than to call routine `DF` twice and then to call `PRESC`. The algorithm is of the same kind that in the latter routine.

**Subroutine NORMA** This routine returns the norm of a Fourier series. The norm that we have selected is the euclidean norm of the vector containing its coefficients.

## 1.4 The Newton Method

Now, using the manipulator described above, it is possible to write a program that looks for a zero of the function $G$ defined before, by means of a Newton method. For this purpose we need an initial condition close enough to the solution that we are looking for. Due to the fact that the perturbation is too big to take the zero solution as initial condition, we are going to use a Newton continuation method. For this reason we have added a continuation parameter $(h)$ multiplying the non constant terms of the perturbations. Thus, when $h$ is equal to 0 we have an autonomous differential system with an equilibrium point near the origin (the equilibrium point is not the origin itself because the constant terms of the perturbations are not multiplied by $h$). This point is close enough to the origin to be found by the

Newton process starting from the zero solution. Then we increase $h$ and this point becomes a small quasiperiodic orbit (see [6]) that can be found by the Newton method (otherwise the value of $h$ must be reduced to an intermediate value). Now the value of $h$ is increased again and the Newton process is started from the solution found for the last value of the parameter. If no problems appear (that is, bifurcations and/or turning points), this process can continue until $h$ reaches the value 1. We shall come back to this point later.

### 1.4.1 Some Remarks about the Jacobian Matrix

Due to the big quantity of coefficients appearing in the Fourier series, it is impossible to include all of them in the Newton method. For this reason we have selected some of them. The selected elements are stored in integer vectors (we store the indices corresponding to them as the vector `nx` seen before) and put in a common called `frenew`. Because of this, we have two different values of the function $G$: the ordinary value and the one obtained taking into account only the terms included in the Newton method. Note that we are solving with respect to this last value.

### 1.4.2 Adding terms to the Newton Process

The criterion to select the coefficients that will take part in the Newton process is based on the size of the amplitudes of the residual acceleration. At the beginning only the terms appearing in the initial condition are selected (if this is constant and equal to zero, only the independent terms take part in the process). When the reduced equation (that is, taking into account only the terms of the Newton process) is solved, the program checks the size of the amplitudes of the terms of the residual acceleration that are outside the Newton process. If it finds that some of them are larger than a given threshold, then they must be added to the Newton process, else this step of the continuation method is finished and we can start the next one.

We want to make a remark about adding terms to the Newton process. If we add too many terms to it, we can find out that the neighbourhood of convergence is too small and we are out of it. In order to avoid this, we have a control on the maximum number of terms going in the Newton process, and if this number is reached, the threshold is increased. We have also implemented a similar procedure if the program is going to take a few new terms for the Newton method. Of course, in this case the corresponding threshold is reduced.

### 1.4.3 High Level Routines

We start writing a routine that evaluates the function $G$. This is not a problem, using the algebraic manipulator and a series expansion for the terms coming from the RTBP. We also have continuation parameters in the terms of each degree and an integer variable containing the actual degree of the power expansion. These parameters are in a common called `conti`. This allows to select the degree at which the solution is wanted.

Then, we need a routine that computes the Jacobian matrix of $G$. This is easy using the chain rule as mentioned in 1.2 and the polynomial expansion of the equations. Note that in this part we also have the continuation parameters for the expansion. Moreover, in order to increase the speed, we realize that some powers and products of series that we need in this part have already been done during the evaluation of $G$, so, we keep these results in memory.

Once the Jacobian matrix is filled up, we need to solve a linear system with a big matrix (note that this matrix is not sparse at all). This system involves only the terms selected before. For this purpose, we have written a routine adapted to a paged environment. The method used is based on the Gaussian elimination.

Due to the fact that we do not know the stepsize for the continuation parameter, we have implemented an automatic stepsize control. This works in the following way: when the Newton method does not find the zero in a few iterations (this number is supplied by a control input file), it assumes that the continuation parameter has been increased too much. Then the program recovers the solution corresponding to the last value of the parameter (it has been stored previously by the program in a disk file), assigns to the parameter an intermediate value and starts the process again. Of course, if the difference between this intermediate value and the last successful one is less than a fixed threshold (given also by a control file) the program stops issuing the corresponding error message.

**Subroutine INIPOT** This routine computes the required powers $x^iy^jz^k$, where $(x,y,\ z)$ is a vector of Fourier series, and stores the result (in the common `/pow/`) to be used by the routines that compute the function $G$ and its Jacobian matrix. If $n$ is the actual degree of the expansion we are working with, then the powers are computed and stored[1] up to order $n-1$. This is due to the fact that the powers of order $n$ are only needed once (during the computation of $G$), while the ones of order less than $n$ are used many times (in the computations of $G$ and its Jacobian matrix). The way to obtain the powers of $x$, $y$ and $z$ is the following: powers of orders less or equal than three are computed "explicitely" (using the algebraic manipulator, we have written directly the expressions for order two and, using these results, the ones of order three). The powers of degree greater or equal than four are obtained as a product of two powers previously computed. This routine calls `POSAP`, `TREUP` and `SPLIT`.

**Subroutine POSAP** This routine stores a Fourier series $a = x^iy^jz^k$ in the common `/pow/`. The parameters are the Fourier series `a`, `ia`, `na` (see section 1.3.1)

[1] except those of degree one

and i, j, k. The routine puts the Fourier series in the vectors real*8 pot and integer*4 npot. This common also has an integer*4 value containing the first free component of the vectors, in order to add the Fourier series at the end of the series already contained in the vectors. Once the series is added, this value is updated. To recover these series we need to write down where they are. This is done by means of the routine LLP (see below): given the integer vector of exponents i, j and k, this routine returns an integer value l. Then, in the column l of the array integer*4 inpo(2,161) we store the first and the last component of pot and npot in which we can find that Fourier series.

**Subroutine LLP** Given the integer vector $(i, j, k)$, the routine returns the column of inpo in which we can find the first and the last components of the arrays pot and npot containing the value $x^i y^j z^k$. More concretely, if $n = i + j + k$, then the returned value is $(((n+3)n+2)n-24)6+i(n+1)-((i-1)i)/2+j+1$.

**Subroutine TREUP** This routine extracts the Fourier series of $x^i y^j z^k$. In order to know where it is stored, a call to the routine LLP is done.

**Subroutine TREUPP** This routine works exactly as TREUP but returning the result multiplied by a given real*8 variable. It is used by the routine MONOMI (see below).

**Subroutine SPLIT** Given an integer vector $(i, j, k)$, this routine returns two integer vectors $(i_1, j_1, k_1)$ and $(i_2, j_2, k_2)$ satisfying $i_1+j_1+k_1 = E(n/2)$ ($E$=integer part), where $n = i+j+k$. This routine is used to know which powers we must multiply to get the power we want. This is used to obtain powers that are not in the common /pow/ (this happens with the powers of maximum order and during the filling of pow).

**Subroutine MONOMI** Given an integer vector $(i, j, k)$, a real number $p$, the Fourier series $x$, $y$ and $z$ and a working vector $w$ containing a Fourier series, this routine returns $w = w + p \times x^i y^j z^k$, where $w$ is the working vector. Before calling this routine a previous call to INIPOT (to compute the required powers of $x$, $y$ and $z$) is needed.

**Subroutine CORXET1** This routine computes the value between brackets ([ ]) in equation (1.1), by means of calls to MONOMI. The value is returned in expanded form inside a working vector.

**Subroutine CORXET2** Like CORXET1, but for equation (1.2).

**Subroutine CORXET3** Like CORXET1, but for equation (1.3).

**Subroutine FUNC** Routine that computes the value of function $G$ (see page 10), calling routines CORXET1, CORXET2, CORXET3 and using the manipulator to compute the other terms appearing in the equations.

**Subroutine DFUNC** This routine computes the Jacobian of function $G$, calling the following routines:

1. D1X: it computes $\frac{\partial G_1}{\partial x}$.
2. D1Y: it computes $\frac{\partial G_1}{\partial y}$.
3. D1Z: it computes $\frac{\partial G_1}{\partial z}$.
4. D2X: it computes $\frac{\partial G_2}{\partial x}$.
5. D2Y: it computes $\frac{\partial G_2}{\partial y}$.
6. D2Z: it computes $\frac{\partial G_2}{\partial z}$.
7. D3X: it computes $\frac{\partial G_3}{\partial x}$.
8. D3Y: it computes $\frac{\partial G_3}{\partial y}$.
9. D3Z: it computes $\frac{\partial G_3}{\partial z}$.

**Subroutine SISTEMA** This routine solves a linear system by means of LU decomposition (see [9]). The operations are done taking into account that FORTRAN stores the matrices as a sequence of columns, and trying to avoid big jumps inside the matrix.

## 1.5 The Program

The program uses many data files in order to read the values of the basic frequencies (the values $\omega = (\omega_1, \ldots, \omega_r)^t$), some phases related to them, control parameters (threshold to drop terms and to stop Newton method, maximum number of iterations and continuation parameters) and coefficients of the polynomial expansions of the equations. The program also reads a file containing the initial condition to start all the process (usually, it is taken equal to zero). During the execution, the program creates a file (called NEWTON.TMP) in which the actual solution is stored. This is due to the fact that if the new step of continuation does not succeed, the program can recover the last succesful solution (for the last continuation parameter, of course) to start again with a smaller parameter. This file is usually small (about 20K). During the run, the program writes an execution log in a file called QPL4.CON. If the execution is succesful, the result is stored in a file called QPL4.RES.

**Subroutine INIEXP** This routine reads the coefficients of the power expansion of the vectorfield from the files EXPAN.1, EXPAN.2 and EXPAN.3 for the first, second and third equations, respectively.

**Subroutine LLEQ** This routine reads one perturbing function $P$ (from the data file QUS.COE), and multiplies the non-constant terms of it by the continuation parameter $h$.

**Subroutine INQUS** This routine reads all the perturbing functions stored in the file QUS.COE and multiplies the non-constant terms of them by the continuation parameter. To do this, the routine LLEQ is used.

**Subroutine INIQP** This routine initializes all the program. It reads the values of the basic frequencies (and phases) of the perturbations (from the file FREQUE.VAL), the control parameters for the Newton method (from the file CONTROL.QP), the coefficients of the power expansion of the vectorfield, the values of the perturbations and the initial condition (from QPL4.DAD). This routine calls INIEXP and INQUS.

**Subroutine QAF** This routine returns the number of terms of the residual acceleration with amplitude larger (in absolute value) than a given threshold.

**Subroutine AFE** This routine adds the terms of the residual acceleration whose amplitude is larger (in absolute value) than a given threshold to the set of variables and to the equations to be solved by Newton's method.

**Subroutine CHECK** This routine is called before adding terms to the Newton process in order to avoid having out-of-range arrays.

**Subroutine CTI** From the residual acceleration (the value of $G$), this routine computes the independent term of the linear system that appears in the Newton method. This step needs a "conversion" from Fourier series to an array, using the vectors of coefficients that take part in the Newton process.

**Subroutine NOUP** This routine computes the new approximation to the solution of $G = 0$, using the solution of the linear system. As in CTI, here we need a "conversion" from an array to Fourier series.

**Subroutine SCOOBY** This routine looks inside the residual acceleration and searches for amplitudes larger (in absolute value) than a threshold `real*8 tol`, and adds them to the Newton process. The number of terms added must be between the `integer*4` values `mitn` and `matn`, modifying the value of `tol` if necessary. This routine calls the routine `AFE`.

**Subroutine CNFNA** This routine controls all the Newton process. It checks the actual step of the process and takes the following decisions:

- If the residual acceleration is not small enough, the new independent term is computed.
- If the value of $G$ is small enough, it looks for "big terms" inside the residual acceleration (calling `SCOOBY`), adding them if necessary.
- If $G$ is small enough and there are not terms inside the residual acceleration big enough to be added to the process, the continuation parameter is moved.
- If the number of iterations with this value of the continuation parameter is too large, it sets a flag telling the routine `NWTA` to recover the last succesful solution, and the continuation parameter step is halved.

The routine returns a flag containing what it has done, because the routine `NWTA` (this is the one that calls `CNFNA`) needs to know what is happening (see below).

**Subroutine NWTA** This routine performs the Newton method. It computes $G$ and then calls `CNFNA` in order to know what to do. Depending on the value of the flag returned by this routine, the action taken can be:

- To continue with the process computing the Jacobian matrix (`DFUNC`), the independent term (`CTI`), solving the linear system (`SISTEMA`), obtaining the new approximation (`NOUP`) and computing again $G$.
- To overwrite the last succesful solution stored in the disk file NEWTON.TMP with the solution obtained for the actual value of the parameter and to compute $G$ again (the continuation parameter has been increased inside `CNFNA`).
- To recover the last succesful solution from NEWTON.TMP and to compute $G$ again (the continuation parameter has been moved inside `CNFNA`).
- To return the control to the main program because the final solution has been found.

**Main Program** Here we can find all the declarations of the arrays. This program calls `INIQP` to initialize the execution and then it calls `NWTA` to perform the Newton method. Finally, it opens the file QPL4.RES and writes in it the final result.

### 1.5.1 Test of the Program

In order to check all of these routines, we have solved a simplified version of the problem: it is considered planar and only the two most relevant frequencies are retained. They correspond to the mean elongation of the Sun ($\psi$) and to the difference between the mean longitude of the Moon and the mean longitude of the lunar perigee ($M$). The equations of motion are (see [1]):

$$\begin{aligned}
\ddot{x} &= q_0\left[-\frac{x-x_E}{r_{PE}^3}(1-\mu)-\frac{x+x_E}{r_{PM}^3}\mu-x_E(1-2\mu)\right]+q_1+q_2x+ \\
&\quad +q_3y+q_4\dot{x}+q_5\dot{y}, \\
\ddot{y} &= q_0\left[-\frac{y-y_E}{r_{PE}^3}(1-\mu)-\frac{y-y_E}{r_{PM}^3}\mu-y_E\right]+q_6+q_7y+q_8\dot{x}+q_9\dot{y},
\end{aligned}$$

where $x_E$, $y_E$, $r_{PE}$ and $r_{PM}$ are defined as above, $q_0,\dots,q_9$ are functions of time:

$$q_i=\sum_{j=0}^{6}A_{ij}\cos\theta_j+B_{ij}\sin\theta_j,$$

with $\theta_0=0$, $\theta_1=2\psi-M$, $\theta_2=\psi$, $\theta_3=M$, $\theta_4=2\psi$, $\theta_5=2M$ and $\theta_6=2\psi+M$. The values of $\psi$ and $M$ are defined by $\psi=0.9251959855t+5.0920835091$ and $M=0.99154522215t+2.2415337977$. We recall that the origin of time is the year 2000.0, and that $2\pi$ units of it are equivalent to a sidereal period of the Moon. The coefficients $A_{ij}$, $B_{ij}$ are given in [1].

We start by setting the continuation parameter of the perturbation ($h$) to 1, and we select the degree of the power expansion equal to one. With this, we get a linear problem that the Newton method can solve easily. Then we add the quadratic terms (without need of any kind of continuation), and finally the third and fourth order terms. We have made another tests starting with the power expansion up to order four and moving the continuation parameter $h$ from 0 to 1. All the results obtained with these tests are in good agreement with the ones of [1].

## 1.6 Results of the Algebraic Manipulator

We have done several runs of this software package. Initially, we tried to use the same scheme as in [1], that is, to fix the continuation parameter $h$ to 1, and the degree $n$ of the power expansion to 1. With this we got a linear problem that the Newton process solved without troubles. Then we added the second degree terms and by using continuation we could solve this problem, but the number of terms taking part in the Newton process was very large and we had many troubles with the memory and swap needed by the program. Then the third degree terms could not be added directly (this is because they are meaningful in this problem) and we had to use continuation again. During this continuation the number of terms inside the Newton process became so high that we started to have troubles with the capacity of our computer. Nevertheless, the solution obtained by this method is good enough to be refined numerically by means of a parallel shooting (see [9]).

Due to the fact that this method does not seem to be the best to solve this problem (computer requirements are very high) we have implemented continuation for the perturbation (as it has already been mentioned during the description of the software). With this, we have obtained better results than the ones mentioned in the last paragraph. As an example, starting with the vectorfield expanded up to order six and giving to the Newton method a threshold $tol = 5 \times 10^{-5}$ (less than the accuracy of the model) to select the coefficients that will take part in the process, the program goes from continuation parameter $h = 0$ to $h = 1$ without any troubles, and the amount of time and memory needed are much less than before (this run is the one supplied as an example of the software).

Taking as initial condition the one supplied by this analytical approximation, we have integrated the model equations and Newton's equations of motion (using the JPL Ephemeris). In Figure 1.1 the solution given by the algebraic manipulator is compared with the integration of the model equations. The integration starts at the epoch 2000.0 and lasts for 1 year. Plots of the solution obtained integrating Newton's equations with the same initial conditions, for a time span of 2 and 5 years, are given in Figures 1.2 and 1.3 respectively. From now on, we shall call this the planar solution because it is close to the ($x$, $y$) plane.

Note that after two years the solution is still contained in a domain of moderate size around $L_4$, but after five years it has grown up to a rather large domain. This is due to the fact that the initial conditions are not good enough at this step and to the inherent instability of the motion around $L_4$. Furthermore one can guess (specially in Figure 1.3, $(x, z)$ projections) a sequence of Lissajous patterns. They suggest the existence of invariant unstable tori.

Now, if we try to improve the planar solution, supplying this solution as an initial contition for the manipulator, setting the continuation parameter $h$ to 1, selecting the degree of the approximation to six (the program can handle up to degree nine, but the terms of degree larger than six can be neglected due to the actual size of the solution) and fixing the criterion to take terms inside the Newton process to $tol = 1 \times 10^{-5}$ (we stress that this improvement is relative to the model equations, not to the JPL model), we find out that, though the residual acceleration of the initial condition is very small, the method diverges. If we try to proceed as for the planar solution (starting from $h = 0$) but with $tol = 1 \times 10^{-5}$, we find out that the Newton process starts to decrease the continuation step until the program stops because the continuation step is too small (this happens for a continuation value near 0.98). Now if we take the intermediate solution in which the process has stopped as an initial condition, we set again $h = 0$ but we fix the value $tol = 1000$ (we do not allow more terms inside the Newton process than the ones of the initial condition) and we start again the program (with this we are dealing with a system of algebraic equations of degree six and with fixed dimension) we find that near the point $h = 0.98$ the Jacobian of $G$ has a change of sign. This is related to some bifurcation or turning point.

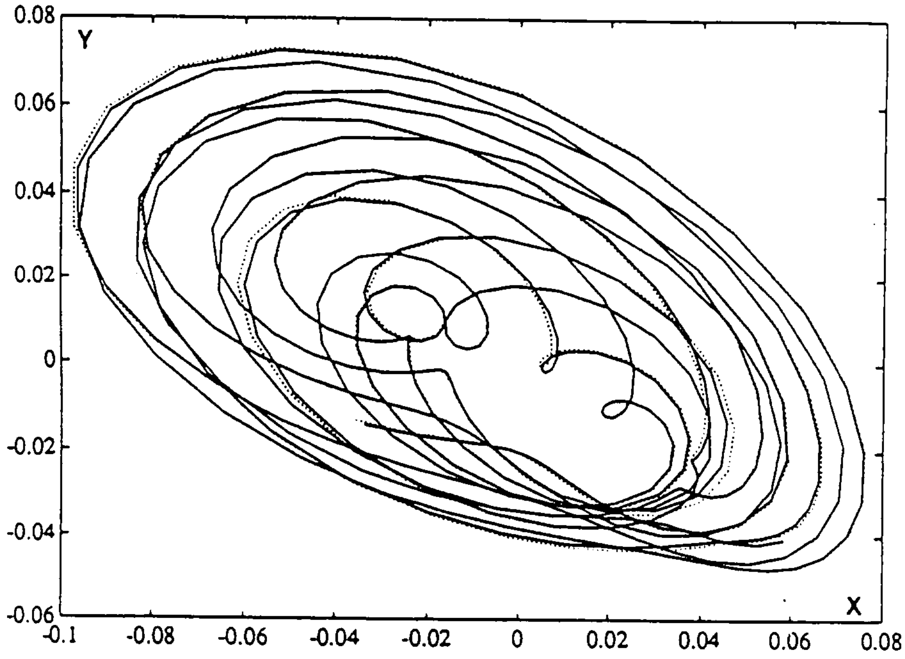


Figure 1.1 (a) (x, y) projection of the solution obtained integrating the model equations (dotted line) and the values of the solution provided by the manipulator (solid line).

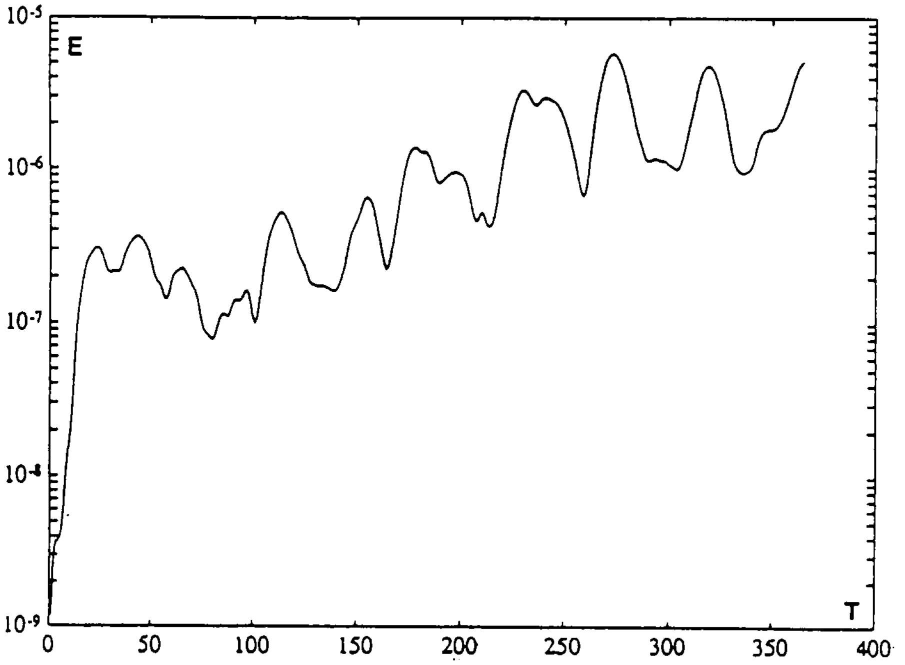


Figure 1.1 (b) Euclidean norm of the differences (in the phase space) between the solution obtained by integration and the one given by the manipulator.

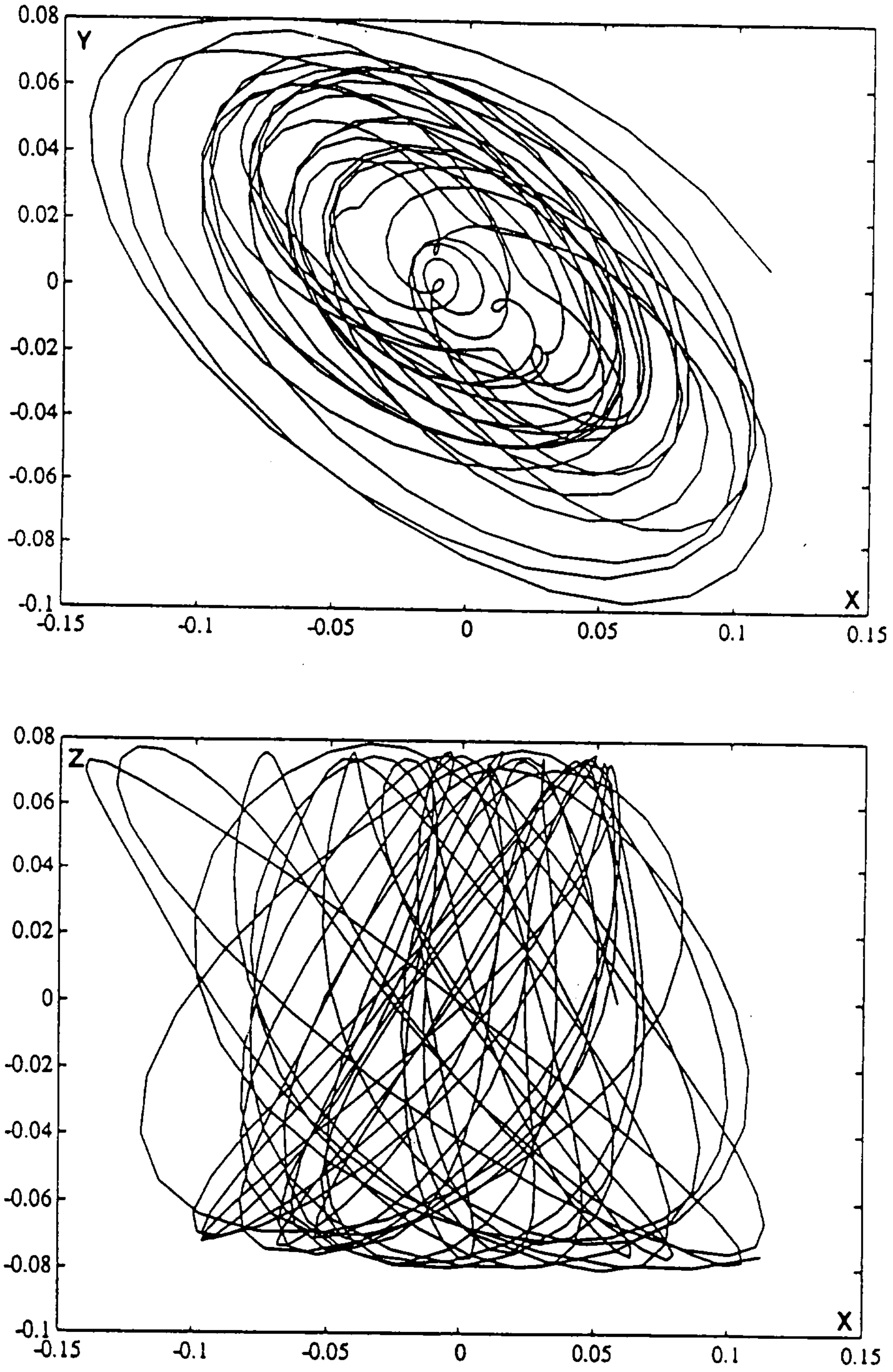


Figure 1.2 Projections of the solution obtained integrating Newton's equations of motion for a time span of 2 years. The initial condition is the one given by the manipulator and it has not been refined.

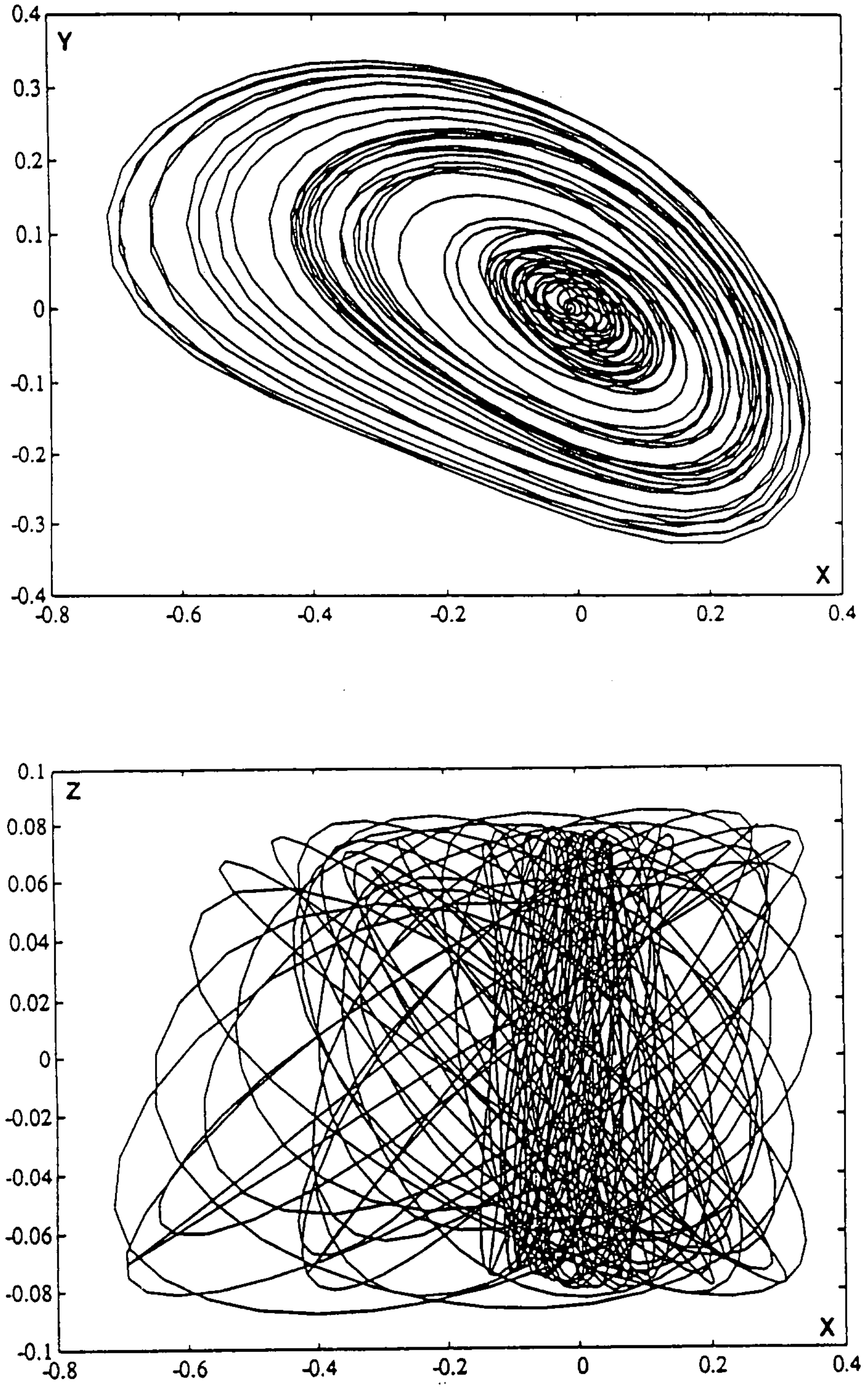


Figure 1.3 Same as Figure 1.2 but for a time span of 5 years.

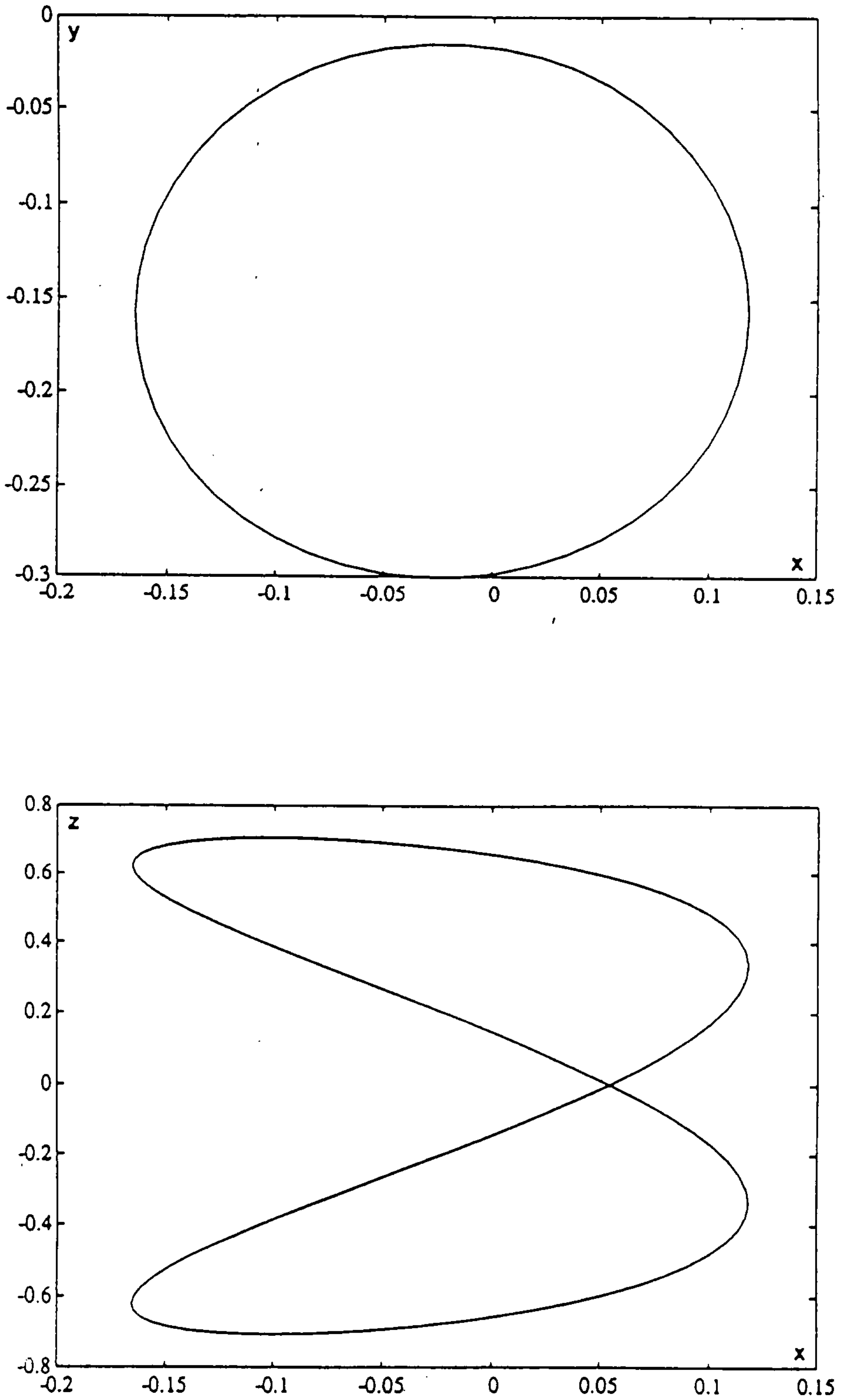


Figure 1.4 Projection of the periodic orbit of averaged model equations.

This last point seems related to the following fact: take the model equations and average them with respect to time. Then, this autonomous system has a periodic solution (see Figure 1.4) whose frequency is rather close to the difference between the first and the third basic frequencies of the model equations (they are, respectively, the mean longitude of the Moon (equal to one, because of the choice of the units) and the mean longitude of the ascending node of the Moon). This orbit seems to be related to the vertical oscilations of the RTBP. When the non-constant part of the perturbation is added, this periodic orbit bifurcates to a quasiperiodic orbit having the same basic frequencies than the ones of the perturbation. This leads us to the fact that the problem of finding a quasiperiodic solution near $L_4$ has lots of solutions, but probably only a few of them have as basic set of frequencies the one of the excitation. Furthermore only a part of the last ones can be obtained as a natural continuation of $L_4$.

## 1.7 Numerical Refinement

In order to obtain a good nominal orbit we have implemented a parallel shooting method (see [9]) to get a solution of the JPL model very similar to the one found with the algebraic manipulator.

Now, let us see some details about this parallel shooting. First of all, we split the time span $[a,b]$ in which we want to find the nominal orbit in several pieces $[t_i,t_{i+1}]$, $i \in \{0,\dots,n-1\}$, verifying that $t_0 = a$, $t_n = b$ and $h = t_{i+1}-t_i = (b-a)/n$ (different time intervals can be used but we have chosen here all of them equal). Now let $i$ be a value between 1 and $n-1$ and let $\mathcal{F}_i$ be the function defined as follows: if $x_i$ is a point of the phase space, then $\mathcal{F}_i(x_i) = y_{i+1}$, where $y_{i+1}$ is the value of the solution defined by the initial conditions $(t_i,x_i)$ at time $t_{i+1}$, under the flow of the JPL model. Now we consider the vector $(x_0,\dots,x_n)$, where $x_i$ are coordinates in the phase space corresponding to time $t_i$. We can define $\mathcal{F} = (\mathcal{F}_0,\dots,\mathcal{F}_{n-1})$ in the usual way:

$$(x_0,\dots,x_n) \xmapsto{\mathcal{F}} (y_1,\dots,y_n),$$

and note that, if the values $x_i$ are all from the same solution of the JPL model, then we should have that $y_i = x_i$, $1 \le i \le n$. Then we impose these conditions and this leads us to solve a nonlinear system of $6n$ equations and $6n+6$ unknowns. To do this, we have used again a Newton method and we have taken as initial guess the values provided by the planar solution obtained using the algebraic manipulator. Due to the fact that we have more unknowns than equations, we have added six more conditions: we have fixed the initial position (the three first components of $x_0$) and the final one (the three first components of $x_n$). With this, the system we deal with has $6n$ equations and the same number of unknowns. Finally, we note that the linear system that appears at each iteration of the Newton method has a band matrix. For this reason we have written a routine adapted to this kind of matrix. Another way to perform the parallel shooting procedure can be the following: instead of adding to the set of $6n$ equations the above mentioned six conditions (which in some sense can force the solution in a non natural way), replace them trying to minimize, with respect to some norm, the total corrections to be done at each step. This procedure has not been implemented.

**Subroutine SISBAN** This routine solves a linear system of equations with a banded matrix, using only the band of the matrix plus three diagonals. The method is based in Gaussian elimination with partial pivoting.

**Subroutine OMPP** This routine computes the initial guess for the Newton method, from a Fourier series corresponding to an approximation obtained by means of the algebraic manipulator. This guess is stored in a vector. This routine uses the routines AVALUA and DF of the manipulator.

**Subroutine COIN** This routine changes coordinates, from the normalized ones to ecliptic ones (JPL), to the vector of initial conditions obtained by OMPP.

**Subroutine FUN** This routine computes the value of the function we want to make zero. For this purpose it performs a numerical integration of the equations of motion in the solar system using the JPL tapes to obtain the position of the bodies. The routine also integrates the variational equations and fills the Jacobian matrix of the function. The device used in this numerical integration has been the Shampine-Gordon method (see [8]).

**Subroutine COOF** This routine performs a change of coordinates, from the ecliptic ones to normalized ones, to the vector of final conditions obtained by the Newton method.

**Main Program** This routine reads the basic frequencies of the perturbations from the file FREQUE.VAL, the initial Fourier series (to obtain the initial condition for the Newton method) from the file QPL4.RES, some control parameters from the file TPL4.CTL (initial time, time step, number of time steps for the Newton process as well as thresholds for the numerical integration and a tolerance to know when the Newton process has finished) and the model of the solar system used from the file MODEL.DAT. With this, the program performs a Newton method (using the routines mentioned above), writing in the console some information about what it is doing. When the solution is found, the program writes the initial condition of the final orbit in the file TPL4.RES.

### 1.7.1 Final Results

Now, using the program described above, it is possible to find an orbit for the JPL model with a behaviour similar to the one of the planar solution obtained by the manipulator. In Figure 1.5 we show the numerical refinement of the planar orbit obtained in section 1.6 for a time span of five years starting at the year 2000. The time step used in the parallel shooting has been of 1 day. To refine this orbit for longer time spans is very difficult (the parallel shooting does not converge) due to the fact that there exists a resonance (already mentioned in section 1.6) implying that this solution is not an exact solution of the system but a minimum of the residual acceleration. Nevertheless, the solution found for five years has a very slow "diffusion" and it goes away of $L_4$ slowly. As an example of this, Figure 1.6 shows its behaviour for more than seven years, and it looks like quasiperiodic.

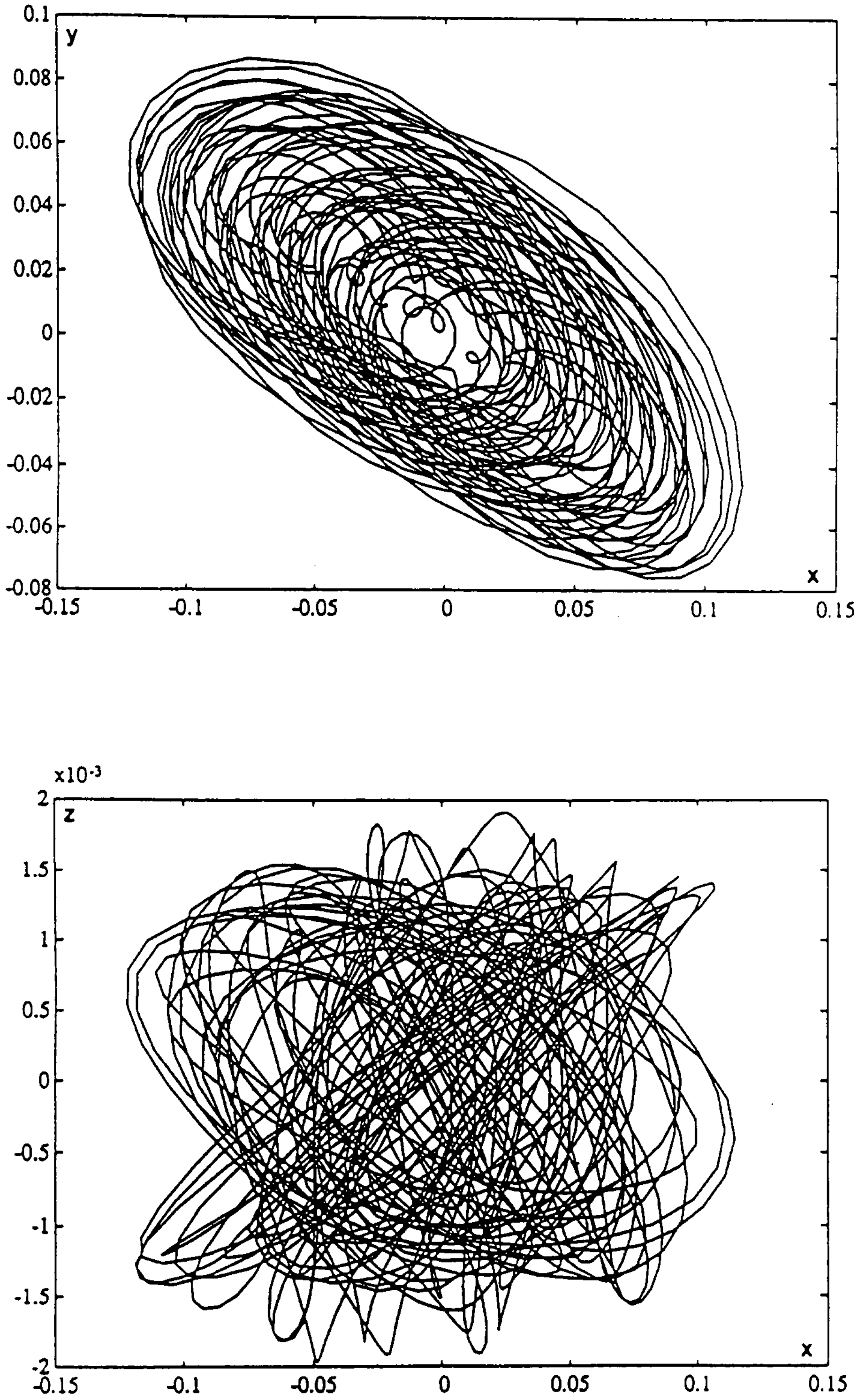


Figure 1.5 Numerical refinement of the planar orbit for a time span of 5 years.

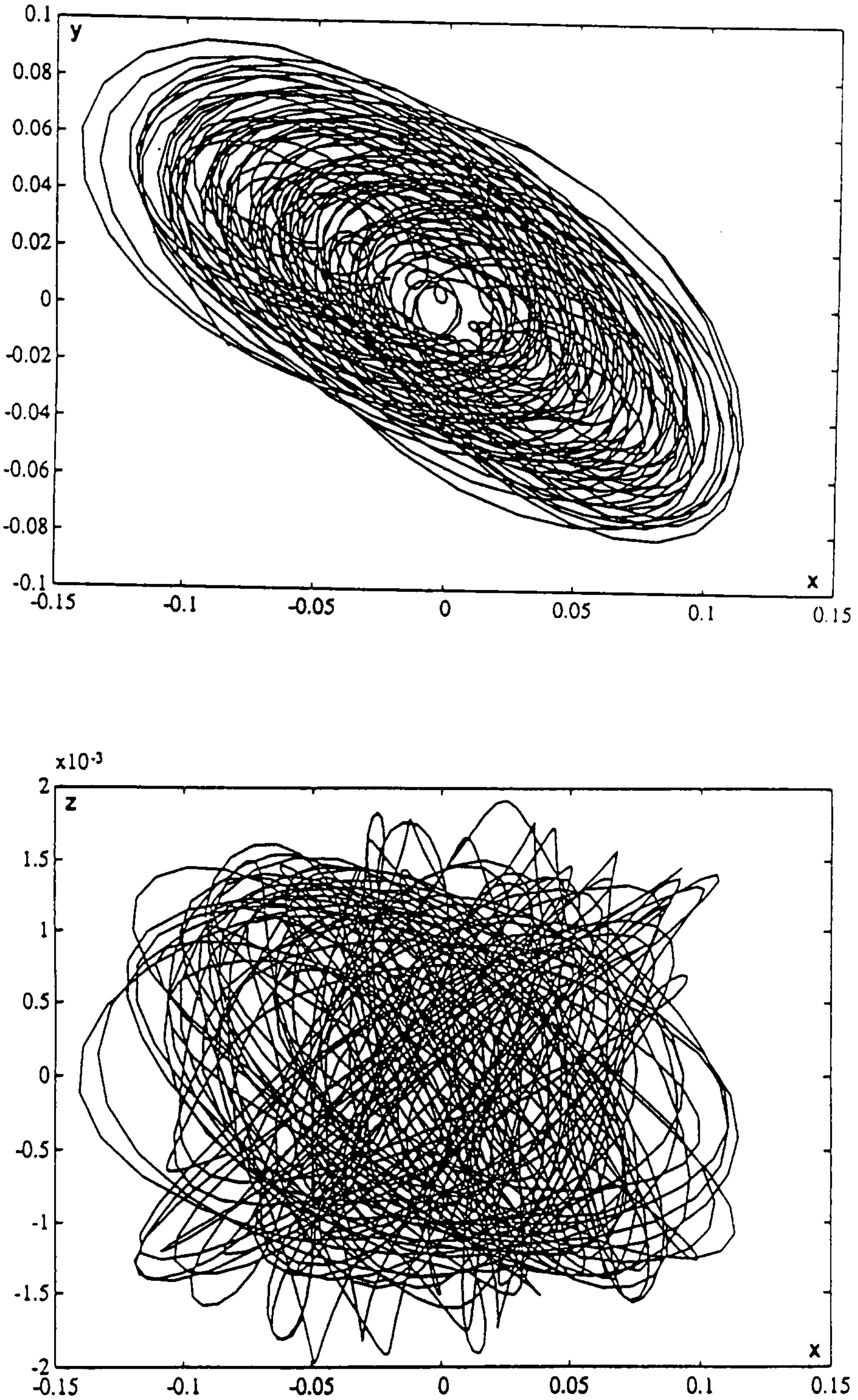


Figure 1.6 The same orbit of Figure 1.5 but integrated for a larger time span, which goes from 500 days before 2000.0 to 2100 days after 2000.0.

## 1.8 The Neighbourhood of the Computed Nearly Quasiperiodic Solution

To study both the stability properties of the computed solution and the relative motion of particles running in orbits close to that one, we have performed a variational study. Of course, in this case we can not talk about monodromy matrix because we never return to the initial conditions or, even if this happens, the subsequent motion will be different. But we can look for points returning close to the initial conditions on the phase space. The figure 1.7 gives the distance to the initial conditions in adimensional units as a function of time. One can see a typical quasiperiodic pattern and after, roughly, 1625 days, we are not too far from the initial point. Now we can consider the variational matrix after that time interval as if it were a monodromy matrix.

On the other hand, during the integration of the variational equations (using the JPL model, of course), we can record the spectral radius of the instantaneous variational matrix. That value has been plotted as a function of time in Figure 1.8, during the full period 2000.0 to 2005.0. Letting aside some local (in time) irregularities on the first 600 days, the pattern is quite neat. These irregularities can be due to the strong constraints on the parallel shooting method which forces the initial and final positions to coincide with the one given by the semianalytical approximation. From the end of the second year on, there are large oscillations with half a year period with superimposed smaller oscillations with period roughly equal to one half of the lunar sinodic period. This can be better seen in Figure 1.9, where we have plotted the quotient $\ln \rho(t)/t$. Here $t$ is the time elapsed since the initial epoch 2000.0 and $\rho(t)$ is the spectral radius (i.e. the maximum of the moduli of the eigenvalues) of the variational matrix at the current time. To skip the initial irregular behaviour this has been plotted starting at the year 2002.0 and the abscisae in the figure denote days since 2002.0. The limit of $\ln \rho(t)/t$ should give the maximal Lyapunov exponent. This seems to be close to $3.65 \times 10^{-3}$ if the time unit is one day. This implies an average increasing of the distance to the computed orbit at a rate of $\exp(365 \times 3.65 \times 10^{-3}) \approx 3.79$ per year. This is an extremely mild instability and offers no problems concerning station keeping.

We return to the approximate monodromy matrix after 1625 days. Let $\lambda_i$, $i = 1, \ldots, 6$ the eigenvalues of this matrix. The numerical values are

$$\begin{array}{lll} \lambda_1 = -30.664197, & \lambda_3 = -0.079456 + 0.996838\mathrm{i}, & \lambda_5 = -0.511657, \\ \lambda_2 = -1.954433, & \lambda_4 = -0.079456 - 0.996838\mathrm{i}, & \lambda_6 = -0.032611. \end{array}$$

They appear almost exactly in reciprocal pairs ($\lambda_j \lambda_{7-j} = 1$) as it happens for autonomous Hamiltonians. Note that the chosen epoch (2000.0 + 1625 days) is particularly good. For that epoch $\ln \rho(t)/t$ equals $2.11 \times 10^{-3}$. But we recall that the limit value $3.65 \times 10^{-3}$, say, is an average. The eigenvalue $\lambda_1$ (resp. $\lambda_6$) is related to a mild instability (resp. stability). Then $\lambda_2$ (resp. $\lambda_5$) to a very mild instability (resp. stability), and $\lambda_3$, $\lambda_4$ are related to neutral behaviour (on the average during that time interval). The corresponding eigenvalues, $e_1$ to $e_6$, are given in Table 1.1 with modulus normalized to 1.

| $e_1$ | $e_2$ | $e_3$, $e_4$ | $e_5$ | $e_6$ |
|---|---|---|---|---|
| −0.163767 | −0.286769 | 0.169256 ± (−0.395523)i | 0.321951 | 0.375073 |
| 0.964713 | 0.940385 | 0.946713 ± 0.000000i | 0.930086 | 0.904416 |
| 0.072496 | 0.053245 | −0.110340 ± 0.913048i | −0.033994 | −0.074290 |
| 0.189793 | 0.162855 | 0.208013 ± 0.002104i | 0.160132 | 0.176994 |
| −0.032347 | −0.059263 | 0.028111 ± (−0.093038)i | 0.061770 | 0.064810 |
| −0.013662 | −0.023880 | 0.137305 ± 0.035283i | −0.025978 | −0.017493 |

Table 1.1

We note that the neutral modes have a stronger relation with the vertical direction $(z,\ \dot{z})$ that the other ones. This is related to the suspected resonance between the proper vertical frequency of the restricted problem around $L_4$ and some of the exciting frequencies.

To see the behaviour of a nearby particle we have computed the following vectors: let $A(t)$ the solution of the variational equations starting with $A(0) = \mathrm{Id}$, where time is counted since the epoch 2000.0. Then $e_j(t) = A(t)e_j$, $j = 1, \ldots, 6$ will give the relative evolution of a particle with respect to the basic orbit (or of one particle with respect to another one if they are located initially at $p \pm \alpha e_j$, where $p$ is the initial condition corresponding to the quasiperiodic solution and $\alpha$ is a small quantity). The vectors $e_j(t)$, $j = 1, \ldots, 6$, with $e_j(0)$ of modulus 1, have been projected on the $(x,\ y)$ plane. Figures 1.10, 1.11, 1.12, 1.13, 1.14 and 1.15 present the results for the modes $e_1(t)$ to $e_6(t)$ respectively. To see the dominant exponentially increasing (resp. decreasing) character of the modulus of $e_1(t)$ (resp. $e_6(t)$) we present the figures 1.10 and 1.15. Figure 1.10 has been obtained as follows: let $(r(t),\ \theta(t))$ be the polar coordinates of the $(x,\ y)$ projection of $e_1(t)$. Compute $\rho(t) = \mathrm{argsinh}(r(t))$ and plot the polar coordinates $(\rho(t),\ \theta(t))$. Figure 1.15 has been obtained with the same procedure that 1.10, but with the scaling $\rho(t) = \mathrm{argsinh}(20r(t))/\mathrm{argsinh}(20)$. Figures 1.11, 1.12, 1.13 and 1.14 have not any kind of scaling. In figures 1.10 and 1.15 one can see the superposition of two dominant periodic contributions added to the exponential behaviour.

Finally if we consider two nearby particles located at symmetrical positions with respect to the quasiperiodic solutions, we suggest the posibility of putting them along the unstable direction. This is because the line joining the particles is sweeping all the directions close to the ecliptic. Assume the relative distance is initially 1 Km. Then, after one month, their distance will increase, on the average, by $\exp(30 \ln(\lambda_1)/1625) - 1 \approx 0.065$ Km.

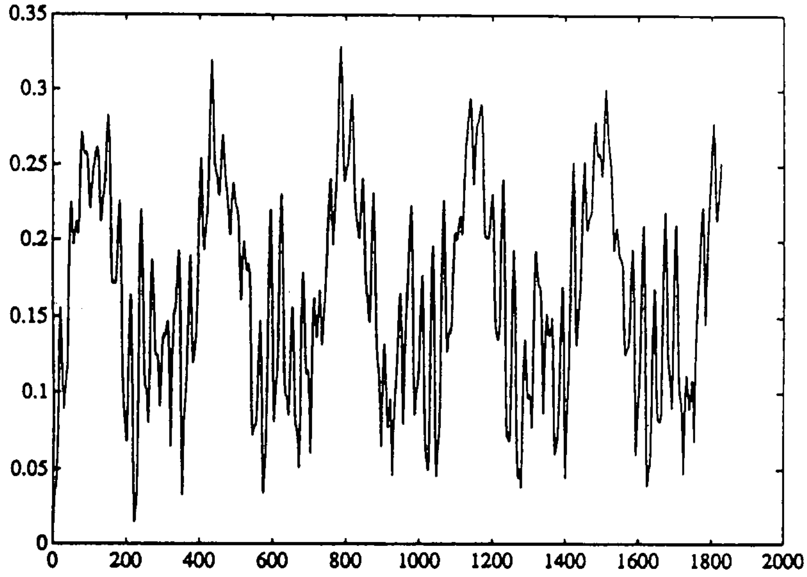


Figure 1.7 Distance to the initial conditions in adimensional units as a function of time (in days) from 2000.0 to 2005.0.

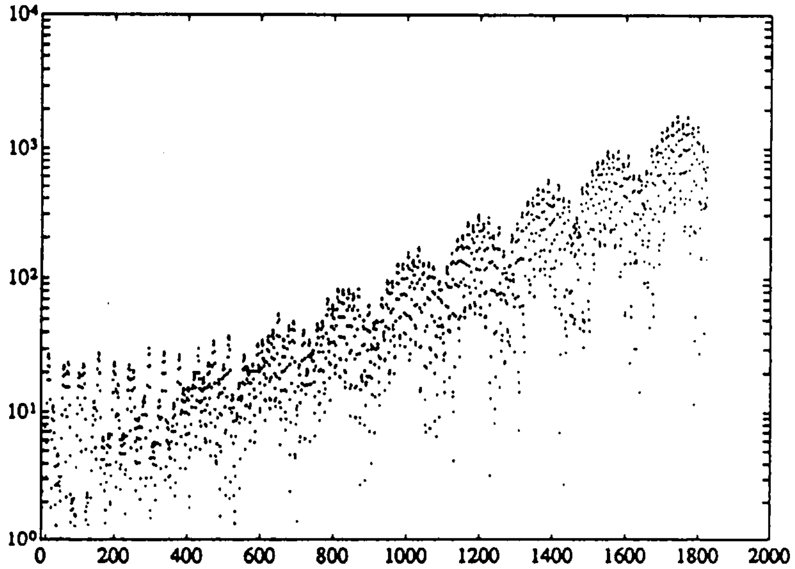


Figure 1.8 Spectral radius of the variational matrix as a function of time (in days) from 2000.0 to 2005.0.

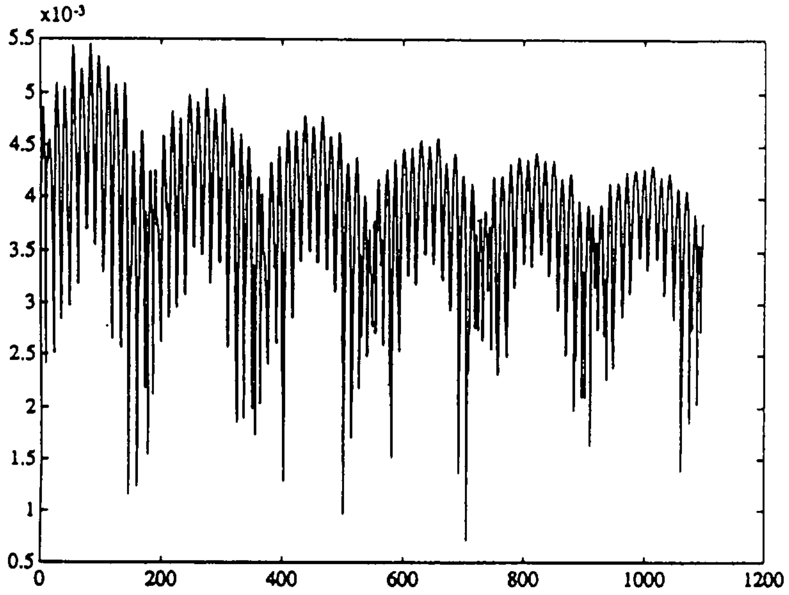


Figure 1.9 Values of $\ln \rho(t)/t$, where the time $t$ (in days) ranges from 2002.0 to 2005, and $\rho(t)$ is the spectral radius of the variational matrix.

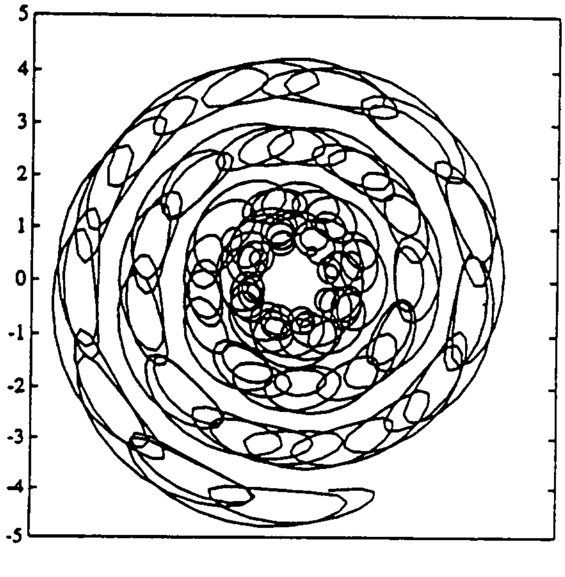


Figure 1.10 $(x, y)$ projection of the mode $e_1(t)$. Polar coordinates have been used, with the scaling $r \sim \mathrm{argsinh}(r)$. Time span: 1625 days.

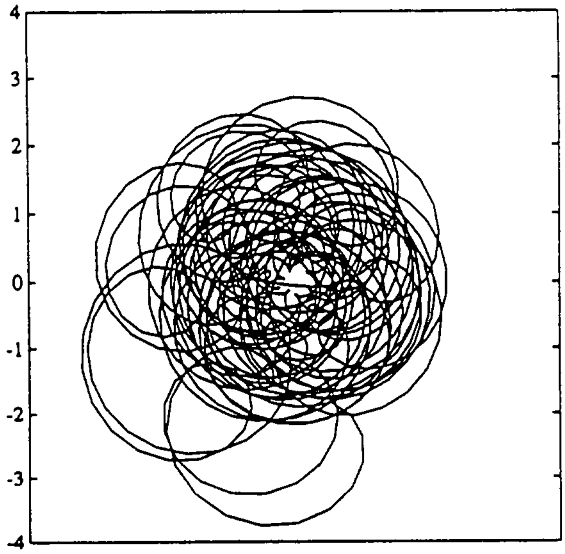


Figure 1.11 $(x,\, y)$ projection of the mode $e_2(t)$. Time span: 1625 days.

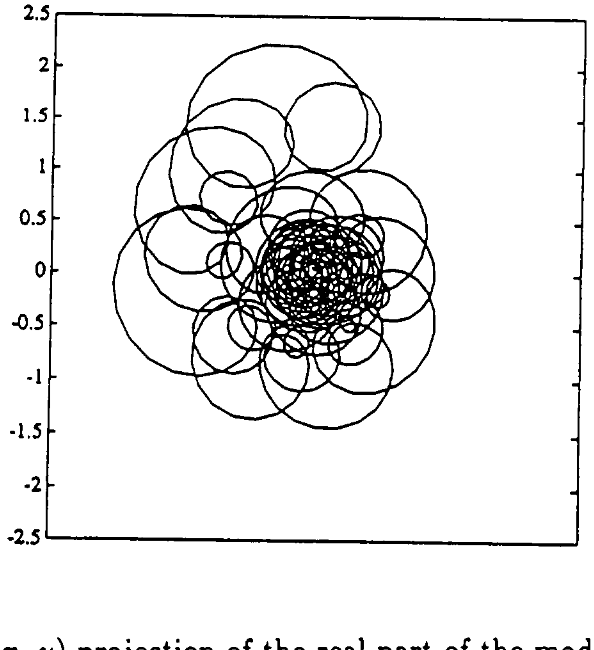


Figure 1.12 $(x,\, y)$ projection of the real part of the mode $e_3(t)$ (Note that $\mathrm{Re}(e_3) = \mathrm{Re}(e_4)$). Time span: 1625 days.

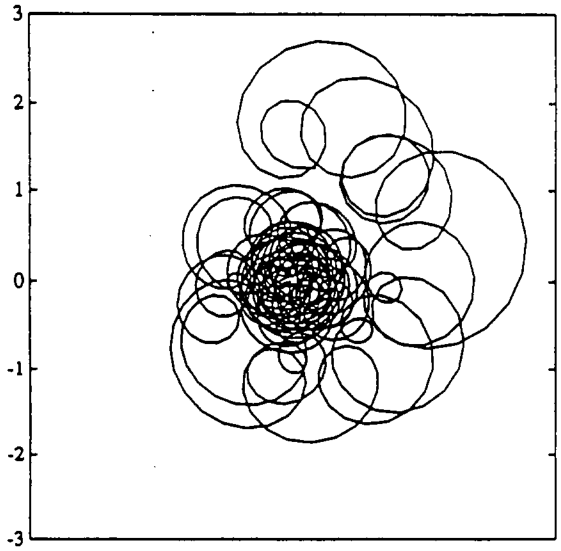


Figure 1.13 $(x, y)$ projection of the imaginary part of the mode $e_3(t)$ (Note that $\mathrm{Im}(e_3) = -\mathrm{Im}(e_4)$). Time span: 1625 days.

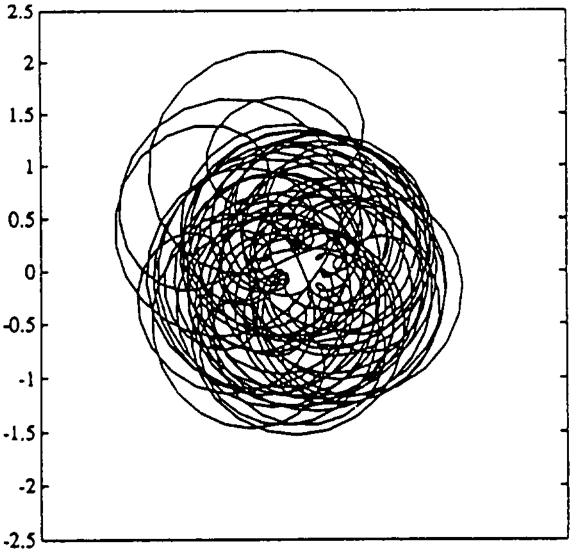


Figure 1.14 $(x, y)$ projection of the mode $e_5(t)$. Time span: 1625 days.

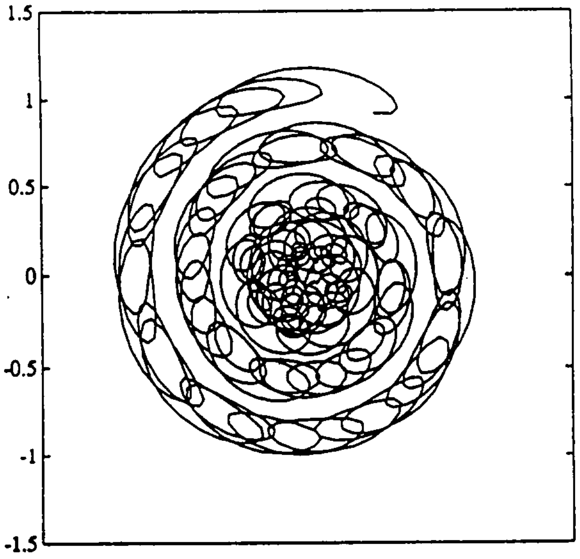


Figure 1.15 $(x,\ y)$ projection of the mode $e_6(t)$. Polar coordinates have been used, with the scaling $r \sim \text{argsinh}(20r)/\text{argsinh}(20)$. Time span: 1625 days.

## 1.9 Problems and Extensions

Looking for the behaviour of the orbits near the geometrical triangular points of the Earth-Moon system we are faced with several problems. If we consider first the RTBP the phase space is filled up by periodic orbits, two-dimensional and three-dimensional tori as well as stochastic zones. The last ones are extremely small near the equilibrium point but the size increases when the distance to the equilibrium point does. To this simplified model we add relatively very important perturbations, mainly due to the effect of the Sun and to the non circular motion of the Moon and the Earth. Those perturbations can be assumed to be quasiperiodic for moderate time intervals. This produces strong resonances in some domains of the phase space and therefore large deviations from the (relatively) clear situation found for the RTBP. For small perturbations we have just to add a kind of quasiperiodic swinging to the previous structure. But the actual size of the perturbations is big enough to produce bifurcations loosing uniqueness. Therefore this is a source of problems for the analytical approximate solutions and for the subsequent numerical refinement as stated in 1.6 and 1.7.

The global problem, that is, the study with great detail of the real behaviour in a full neighbourhood of the former equilibrium point is worthy. It is useful for the design of future missions. From the abstract point of view there is a lack of supporting theory. From the analytical and numerical points of view this is a formidable task. We think that the methodology to be used to deal with this problem would be useful in many other contexts.

# Chapter 2

# Global Description of the Orbits Near the $L_1$ Point of the Earth-Sun System in the RTBP

## 2.1 Introduction

In the spatial RTBP the $L_1$ and $L_2$ libration points have a linear behaviour of the type saddle × center × center. Under generic assumptions there exist nearby two dimensional tori. Because of the small divisors problems, on those tori the frequencies should satisfy a diophantine condition. But as we are interested in short time intervals the small divisors problems do not show up unless they are associated to low order resonances. Let $\omega_1$, $\omega_2$ be the two basic frequencies at $L_i$, i=1, 2. For small values of $\mu$ the frequencies are rather close ( for the Hill's problem, $\mu = 0$, they are $(28^{1/2}-1)^{1/2}$ and 2, respectively). When the amplitudes of the orbit change so do the frequencies. It is known that the Lyapunov orbits bifurcate to halo orbits when the $x$ amplitude ( $x$ is the axis with origin at $L_i$ in the direction of the primaries) is roughly 0.139 if the unit of distance is the one from the small mass to the libration point. Hence this is the distance corresponding to the 1 to 1 resonance giving rise to the halo orbits. If we keep our study in a smaller region the effect of this resonance is not too strong. Other resonances are found to higher order. As $\omega_1 \ / \ \omega_2 \simeq 29/28$ for the Hill's problem, the next resonances to play a significant role are of order 57. They are of small amplitude and long period. Therefore we can try to obtain the invariant tori formally to some order (less than 57, of course) and taking only into account the 1 to 1 resonance. Cutting the expressions to this order we obtain a foliation by invariant tori in some region around the equilibrium points. For some of the amplitudes the tori certainly do not exist, but the small size of the stochastic zone makes this fact irrelevant for short time interval applications, in our case a spacecraft running during a few years.

## 2.2 The Equations of Motion

We restrict the exposition to the $L_1$ libration point (using the convention that this relative equilibrium point is located between the primaries). We consider the following system of reference: The positive $x$ axis is directed from $L_1$ to the smaller primary. The $y$ axis is in the plane of sidereal motion of the primaries at $\pi/2$ from the $x$ axis in counterclockwise sense. The $z$ axis completes a positively oriented coordinate system. Let $\gamma = d(m_2, L_1)$, where $m_2$ denotes the position of the smaller primary. The origin is taken at the libration point and $\gamma$ is the unit of distance. The unit of time is such that the period of revolution of the primaries in the sidereal system is $2\pi$. Let $\mu$ be the mass of the smaller primary (in our case the Earth, or rather the Earth-Moon barycenter; $\mu \approx 0.3040357143 \times 10^{-5}$ in this Chapter) and $1-\mu$ the one of the bigger primary (the Sun in our application). The value of $\gamma$ is obtained from the Euler quintic equation:

$$\gamma^5 - (3-\mu)\gamma^4 + (3-2\mu)\gamma^3 - \mu\gamma^2 + 2\mu\gamma - \mu = 0.$$

It can be solved by Newton method starting at $(\mu/3)^{1/3}$.

Let $X$, $Y$, $Z$ be the usual synodical coordinates centered at the center of mass. The equations of motion are

$$\begin{aligned} \ddot{X} - 2\dot{Y} &= \Omega_X, \\ \ddot{Y} + 2\dot{X} &= \Omega_Y, \\ \ddot{Z} &= \Omega_Z, \end{aligned} \tag{2.1}$$

where

$$\Omega = \frac{1}{2}(X^2+Y^2) + (1-\mu)r_1^{-1} + \mu r_2^{-1},$$

being

$$\begin{aligned} r_1^2 &= (X-\mu)^2 + Y^2 + Z^2, \\ r_2^2 &= (X-\mu+1)^2 + Y^2 + Z^2. \end{aligned}$$

As we have described we make the change of coordinates

$$\begin{aligned} x &= -\frac{1}{\gamma}(X-\mu+1-\gamma), \\ y &= -\frac{1}{\gamma}Y, \\ z &= \frac{1}{\gamma}Z. \end{aligned} \tag{2.2}$$

Then

$$\begin{aligned} r_1^{-1} &= (1-\gamma)^{-1}(1+\frac{2\gamma}{1-\gamma}x+\frac{\gamma^2}{(1-\gamma)^2}\rho^2)^{-1/2} = \\ &= (1-\gamma)^{-1}\sum_{n\geq 0}(\frac{\gamma\rho}{1-\gamma})^n(-1)^n P_n(\frac{x}{\rho}), \\ r_2^{-1} &= \gamma^{-1}(1-2x+\rho^2)^{-1/2} = \\ &= \gamma^{-1}\sum_{n\geq 0}\rho^n P_n(\frac{x}{\rho}), \end{aligned}$$

where

$$\rho^2 = x^2+y^2+z^2$$

and $P_n$ denote the Legendre polynomials.

Let

$$c_n = \gamma^{-3}\left(\mu+(-1)^n(1-\mu)(\frac{\gamma}{1-\gamma})^{n+1}\right). \tag{2.3}$$

Then the equations of motion are written as

$$\begin{aligned} \ddot{x}-2\dot{y}-(1+2c_2)x &= \frac{\partial}{\partial x}\sum_{n\geq 3}c_n\rho^n P_n(\frac{x}{\rho}), \\ \ddot{y}+2\dot{x}+(c_2-1)y &= \frac{\partial}{\partial y}\sum_{n\geq 3}c_n\rho^n P_n(\frac{x}{\rho}), \\ \ddot{z}+c_2 z &= \frac{\partial}{\partial z}\sum_{n\geq 3}c_n\rho^n P_n(\frac{x}{\rho}). \end{aligned} \tag{2.4}$$

We recall also, for further use, that (2.1) can be written in Hamiltonian form. Introducing

$$\begin{aligned} P_X &= \dot{X}-Y, \\ P_Y &= \dot{Y}+X, \\ P_Z &= \dot{Z}, \end{aligned}$$

the Hamiltonian is

$$\begin{aligned} H &= \frac{1}{2}(P_X^2+P_Y^2+P_Z^2)+YP_X-XP_Y-(1-\mu)r_1^{-1}-\mu r_2^{-1} = \\ &= \frac{1}{2}(\dot{X}^2+\dot{Y}^2+\dot{Z}^2)-\Omega. \end{aligned} \tag{2.5}$$

Let $T_n=\rho^n P_n(x/\rho)$, $R_{n-1}=y^{-1}\partial T_{n+1}/\partial y = z^{-1}\partial T_{n+1}/\partial z$. Then (2.4) gives

$$\begin{aligned} \ddot{x}-2\dot{y}-(1+2c_2)x &= \sum_{n\geq 2}c_{n+1}(n+1)T_n, \\ \ddot{y}+2\dot{x}+(c_2-1)y &= y\sum_{n\geq 2}c_{n+1}R_{n-1}, \\ \ddot{z}+c_2 z &= z\sum_{n\geq 2}c_{n+1}R_{n-1}. \end{aligned} \tag{2.6}$$

We remark that $T_n$ ( resp. $R_{n-1}$) is a polynomial of degree $n$ (resp. $n-1$), in $x$, $y$, $z$.

The equations 2.6 can also be put in Hamiltonian form. Let $p_x = \dot{x}-y$, $p_y = \dot{y}+x$, $p_z = \dot{z}$. Then the correponding Hamiltonian is

$$K = \frac{1}{2}\left(p_x^2+p_y^2+p_z^2\right)+yp_x-xp_y-\sum_{n\geq 2} c_n\rho^n P_n\left(\frac{x}{\rho}\right). \tag{2.7}$$

The hamiltonians $H$ and $K$ are related by

$$H = K\gamma^2-\frac{1}{2}(1-\gamma-\mu)^2-\frac{\mu}{\gamma}-\frac{1-\mu}{1-\gamma}. \tag{2.8}$$

## 2.3 Formal Series Solutions

If we skip in (2.6) the right hand terms (of degree $\geq 2$) the solution of the system, restricted to the center manifold (i.e. deleting the exponentially increasing or decreasing terms) is

$$\begin{aligned} x &= \alpha\cos(\omega_1\tau+\phi_1), \\ y &= -\overline{k}\alpha\sin(\omega_1\tau+\phi_1), \\ z &= \beta\cos(\omega_2\tau+\phi_2), \end{aligned} \tag{2.9}$$

with

$$\overline{k} = (\omega_1^2+2c_2+1)/(2\omega_1).$$

The parameters $\alpha,\beta$ are arbitrary amplitudes and $\phi_1,\phi_2$ are arbitrary phases. The frequencies in (2.9) are $\omega_1 = ((2-c_2+(9c_2^2-8c_2)^{1/2})/2)^{1/2}$, $\omega_2 = c_2^{1/2}$. As (2.6) is an autonomous system we can take the origin of time such that $\phi_1 = 0$. Furthermore we denote $\omega_1\tau$ and $\omega_2\tau+\phi_2$ as $\theta_1$ and $\theta_2$, respectively. When the right hand terms of (4) are considered we should allow for varying frequencies depending on the amplitudes, according to the Lindstedt-Poincaré method to avoid secular terms. Then $\theta_1$ will be now $d\tau$ and $\theta_2$ will be $f\tau+\phi_2$, where $d = \omega_1+O(\alpha,\beta)$, $f = \omega_2+O(\alpha,\beta)$.

We look for solutions in complex exponential form of the type

$$\begin{aligned} x &= \sum x_{i,j,k,m}\alpha^i\beta^j\gamma_1^k\gamma_2^m, \\ y &= \sqrt{-1}\sum y_{i,j,k,m}\alpha^i\beta^j\gamma_1^k\gamma_2^m, \\ z &= \sum z_{i,j,k,m}\alpha^i\beta^j\gamma_1^k\gamma_2^m, \end{aligned} \tag{2.10}$$

where $\gamma_s = \exp(\sqrt{-1}\theta_s)$, $s = 1,2$. Due to the symmetries of the problem the coefficients in (2.10) satisfy the relations:

1. $i,j \geq 0, i+j \geq 1, \mid k \mid \leq i, \mid m \mid \leq j, i-k \equiv 0 \pmod 2$, $j-m \equiv 0 \pmod 2$ in all the cases,

2. $x_{i,j,-k,-m} = x_{i,j,k,m}$, $y_{i,j,-k,-m} = -y_{i,j,k,m}$, $z_{i,j,-k,-m} = z_{i,j,k,m}$. Hence it is enough, for the computations, to keep only the terms with $k > 0$ or, if $k = 0$, with $m \geq 0$,

3. For $x$ and $y$ one should have $j \equiv 0 \pmod 2$, and for $z$ one should have $j \equiv 1 \pmod 2$.

4. The Lindstedt-Poncaré method can be normalized in such a way that for $(k,m) = (1,0)$ the only term appearing in $x$ corresponds to $(i,j) = (1,0)$. In an analogous way if $(k,m) = (0,1)$ the only term in $z$ corresponds to $(i,j) = (0,1)$.

5. In $d$ and $f$ there are only terms of the form $\alpha^i\beta^j$ with $i$ and $j$ even.

A sum as $x$ (resp. $y$, resp. $z$) will be called of type $\underline{a}$ (resp. $\underline{b}$, resp. $\underline{c}$). The product of a series of type $\underline{a}$ by another of type $\underline{a}$, $\underline{b}$ or $\underline{c}$ gives again a series of type $\underline{a}$, $\underline{b}$ or $\underline{c}$, respectively. The product of a series of type $\underline{b}$ by another of type $\underline{b}$ (or of one of type $\underline{c}$ by another of type $\underline{c}$) gives again a series of type $\underline{a}$. The series $\dot{x}$, $\ddot{x}$, $\dot{y}$, $\ddot{y}$ and $\ddot{z}$, appearing in (2.6) are, respectively of types $\underline{b}$, $\underline{a}$, $\underline{a}$, $\underline{b}$ and $\underline{c}$.

The computation of the terms in (2.10) is done by increasing order of $n = i + j$. For $n = 1$ one has from (2.9) the values

$$x_{1,0,1,0} = 1/2, \quad y_{1,0,1,0} = \overline{k}/2, \quad z_{0,1,0,1} = 1/2.$$

The terms $T_m$ and $R_m$ in (2.6) are obtained by a recurrence which follows from the one of the Legendre polynomials

$$\begin{aligned} T_m(x,y,z) &= (2-\frac{1}{m})xT_{m-1} - (1-\frac{1}{m})\rho^2 T_{m-2}, \\ R_m(x,y,z) &= \frac{2m+3}{m+2}xR_{m-1} - \frac{2m+2}{m+2}T_m - \frac{m+1}{m+2}\rho^2 R_{m-2}, \end{aligned} \tag{2.11}$$

starting with $T_0 = 1$, $T_1 = x$, $R_0 = -1$, $R_1 = -3x$.

The derivatives with respect to $\tau$ are obtained by means of

$$\frac{d}{d\tau} = d\frac{\partial}{\partial\theta_1} + f\frac{\partial}{\partial\theta_2}.$$

Then $\dot{x}$, $\ddot{x}$, $\dot{y}$, $\ddot{y}$, and $\ddot{z}$ are found from (2.10).

Let $M$, $\sqrt{-1}N$ and $P$ be the right hand terms in (2.6). They are of types $\underline{a}$, $\underline{b}$ and $\underline{c}$, respectively. Assume $x$, $y$, $z$ have been obtained to order $n-1$. Then $M$, $N$ and $P$ are known to order $n$. Some terms of the left hand part of (2.6) are also known. For instance, if $x = x_1 + x_2 + \ldots + x_{n-1}$, where $x_j$ denotes the terms of order $j$ and, in a similar way, $d = d_0 + d_1 + \ldots + d_{\overline{n}}$, $f = f_0 + f_1 + \ldots + f_{\overline{n}}$, where $\overline{n} = n-2$ if $n$ is even and $\overline{n} = n-3$, if n is odd, then in $\dot{x}$ there are terms already known at order $n$ as

$$\frac{\partial x_{n-2}}{\partial \theta_1} d_2 + \frac{\partial x_{n-2}}{\partial \theta_2} f_2 + \frac{\partial x_{n-4}}{\partial \theta_1} d_4 + \frac{\partial x_{n-4}}{\partial \theta_2} f_4 + \ldots .$$

Transfering those terms of order $n$, already known when $x$, $y$, $z$ are available at order $n-1$, we obtain new right hand terms, whose components at order $n$ are denoted by $\overline{M}_n$, $\overline{N}_n$ and $\overline{P}_n$, and we obtain

$$\begin{aligned}
\frac{\partial^2 x_n}{\partial \theta_1^2} d_0^2 + \frac{\partial^2 x_n}{\partial \theta_1 \partial \theta_2} 2 d_0 f_0 + \frac{\partial^2 x_n}{\partial \theta_2^2} f_0^2 - 2 \left( \frac{\partial y_n}{\partial \theta_1} d_0 + \frac{\partial y_n}{\partial \theta_2} f_0 \right) - & \\
-(1+2c_2) x_n + \delta \left( \frac{\partial^2 x_1}{\partial \theta_1^2} 2 d_0 d_{n-1} - 2 \frac{\partial y_1}{\partial \theta_1} d_{n-1} \right) &= \overline{M}_n, \\
\frac{\partial^2 y_n}{\partial \theta_1^2} d_0^2 + \frac{\partial^2 y_n}{\partial \theta_1 \partial \theta_2} 2 d_0 f_0 + \frac{\partial^2 y_n}{\partial \theta_2^2} f_0^2 + 2 \left( \frac{\partial x_n}{\partial \theta_1} d_0 + \frac{\partial x_n}{\partial \theta_2} f_0 \right) + & \\
+(c_2 - 1) y_n + \delta \left( \frac{\partial^2 y_1}{\partial \theta_1^2} 2 d_0 d_{n-1} + 2 \frac{\partial x_1}{\partial \theta_1} d_{n-1} \right) &= \overline{N}_n \sqrt{-1}, \\
\frac{\partial^2 z_n}{\partial \theta_1^2} d_0^2 + \frac{\partial^2 z_n}{\partial \theta_1 \partial \theta_2} 2 d_0 f_0 + \frac{\partial^2 z_n}{\partial \theta_2^2} f_0^2 + & \\
+c_2 z_n + \delta \left( \frac{\partial^2 z_1}{\partial \theta_2^2} 2 f_0 f_{n-1} \right) &= \overline{P}_n.
\end{aligned} \tag{2.12}$$

Here $\delta = 0$ if $n$ is even and $\delta = 1$ if $n$ is odd.

From (2.12) and taking into account the properties 1) to 5) one obtains $x_n$, $y_n$, $z_n$ and, if $n$ is odd, also $d_{n-1}$ and $f_{n-1}$. In the determination of $x_n$, $y_n$, $z_n$ one should impose the conditions $x_{i,j,1,0} = 0$ if $(i,j) \neq (1,0)$ and $z_{i,j,0,1} = 0$ if $(i,j) \neq (0,1)$ (see property 4) ). This is what allows to obtain $d_{n-1}$ and $f_{n-1}$ for $n$ odd. More concretely, (2.12) can be written in components as

$$\begin{aligned}
-x_{i,j,k,m}(k d_{0,0} + m f_{0,0})^2 + \delta_{k,1}\delta_{m,0} x_{1,0,1,0}(-2 d_{0,0} d_{i-1,j}) + & \\
+2 y_{i,j,k,m}(k d_{0,0} + m f_{0,0}) + \delta_{k,1}\delta_{m,0} 2 y_{1,0,1,0} d_{i-1,j} - & \\
-(1+2c_2) x_{i,j,k,m} &= \overline{M}_{i,j,k,m},
\end{aligned} \tag{2.13}$$

$$\begin{aligned}
-y_{i,j,k,m}(k d_{0,0} + m f_{0,0})^2 + \delta_{k,1}\delta_{m,0} y_{1,0,1,0}(-2 d_{0,0} d_{i-1,j}) + & \\
+2 x_{i,j,k,m}(k d_{0,0} + m f_{0,0}) + \delta_{k,1}\delta_{m,0} 2 x_{1,0,1,0} d_{i-1,j} + & \\
+(c_2 - 1) y_{i,j,k,m} &= \overline{N}_{i,j,k,m},
\end{aligned} \tag{2.14}$$

$$\begin{aligned}
-z_{i,j,k,m}(k d_{0,0} + m f_{0,0})^2 + \delta_{k,0}\delta_{m,1} z_{0,1,0,1}(-2 f_{0,0} f_{i,j-1}) + & \\
+c_2 z_{i,j,k,m} &= \overline{P}_{i,j,k,m},
\end{aligned} \tag{2.15}$$

where $\delta_{r,s}$ denotes the delta of Kronecker. From (2.13) and (2.14) if $k = 1$ and $m = 0$ we have

$$x_{i,j,1,0}(-d_{0,0}^2 - 1 - 2c_2) + 2 y_{i,j,1,0} d_{0,0} +$$

$$\begin{aligned} +d_{i-1,j}(-x_{1,0,1,0}2d_{0,0}+2y_{1,0,1,0}) &= \overline{M}_{i,j,1,0}, \\ 2x_{i,j,1,0}d_{0,0}+y_{i,j,1,0}(-d_{0,0}^2+c_2-1)+ & \\ +d_{i-1,j}(-y_{1,0,1,0}2d_{0,0}+2x_{1,0,1,0}) &= \overline{N}_{i,j,1,0}. \end{aligned}$$

This is solved by putting $x_{i,j,1,0}=0$ ( note that $n \geq 2$) and obtainig then $y_{i,j,1,0}$ and $d_{i-1,j}$, the determinant being different from zero. If $(k,m) \neq (1,0)$ then one has

$$\begin{aligned} \left(-(kd_{0,0}+mf_{0,0})^2-(1+2c_2)\right)x_{i,j,k,m}+ & \\ +2(kd_{0,0}+mf_{0,0})y_{i,j,k,m} &= \overline{M}_{i,j,k,m}, \\ \left(-(kd_{0,0}+mf_{0,0})^2+c_2-1\right)y_{i,j,k,m}+ & \\ +2(kd_{0,0}+mf_{0,0})x_{i,j,k,m} &= \overline{N}_{i,j,k,m}, \end{aligned} \tag{2.16}$$

which can be solved for $x_{i,j,k,m}$ and $y_{i,j,k,m}$, the determinant being always different from zero.

From (2.15) if $k=0$ and $m=1$ we obtain

$$z_{i,j,0,1}(-f_{0,0}^2+c_2)+z_{0,1,0,1}(-2f_{0,0}f_{i,j-1})=\overline{P}_{i,j,0,1},$$

which is solved for $f_{i,j-1}$ if one puts $z_{i,j,0,1}=0$ according to property 4). If $(k,m) \neq (0,1)$ then

$$[-(kd_{0,0}+mf_{0,0})^2+c_2]z_{i,j,k,m}=\overline{P}_{i,j,k,m}, \tag{2.17}$$

and we obtain $z_{i,j,k,m}$ because $c_2-(kd_{0,0}+mf_{0,0})^2$ is only zero if $k=0$ , $m=1$.

Had we not introduced $d$ and $f$ we would find the impossibility to obtain $x_{i,j,1,0}$, $y_{i,j,1,0}$ from (2.16) and $z_{i,j,0,1}$ from (2.17) because of zero divisors. The choices $x_{i,j,1,0}=0$, $z_{i,j,0,1}=0$ are arbitrary, but they can be considered as a normalization. In this way the parameters $\alpha$ and $\beta$ can be interpreted as the coefficients of $\cos\theta_1$ and $\cos\theta_2$, respectively, in the final expressions of $x$ and $z$.

We summarize the steps to follow to obtain $x$, $y$, $z$ to order 2, 3,... . Assume that they are known to order $n-1$ (and $d$, $f$ to order $n-2$ if $n$ is even, to order $n-3$ if $n$ is odd). Then we compute $\rho^2$ to order $n$, $T_m$ to order $n$ up to $m=n$, $R_m$ to order $n-1$ up to $m=n-1$. With this we have $M_n$, $N_n$ and $P_n$. Next we obtain the required first and second derivatives of $x$, $y$, $z$ with respect to $\theta_1$, $\theta_2$ and the terms $d^2$, $2df$, $f^2$ to orders 2 to $n-2$ (resp. $n-1$) if $n$ is even (resp. if $n$ is odd). From this we get $\overline{M}_n$, $\overline{N}_n$, $\overline{P}_n$ and we solve for the $x_{i,j,k,m}$, $y_{i,j,k,m}$ $z_{i,j,k,m}$ with $i+j=n$ and, if $n$ is odd, for $d_{i,j}$ and $f_{i,j}$ with $i+j=n-1$.

### 2.3.1 A Program to Obtain the Formal Series Solutions

As said before the properties 3) to 5) allow to reduce the amount of terms to store. For $x$, $y$, and $z$ the terms are stored in the following order. A term $(i,j,k,m)$ precedes a term $(\bar{\imath},\bar{\jmath},\bar{k},\overline{m})$ if $i+j<\bar{\imath}+\bar{\jmath}$. If $i+j=\bar{\imath}+\bar{\jmath}$, then if $i>\bar{\imath}$. If also

$i = \bar{\imath}$ then if $k > \bar{k}$. Finally, if also $k = \bar{k}$, if $m > \bar{m}$. The following instructions allow us to run over all the terms of $x$, $y$ or $z$ up to order $n$ (note that we do not save the space corresponding to the known fact that $x_{i,j,1,0} = 0$ if $(i,j) \neq (1,0)$ and that $z_{i,j,0,1} = 0$ if $(i,j) \neq (0,1)$ ). The parameter IS takes the value 0 for $x$ and $y$, and the value 1 for $z$. The index L is the current index of the term:

```
     L = 0
     DO 1 NA = 1, N
     NAM = NA - IS
     NA1 = MOD (NAM, 2)
     DO 2 I = NAM, NA1, -2
     J = NA - I
     DO 3 K = I, NA1, -2
     JF = -J
     IF (K. EQ. 0) JF = 0
     DO 4 M = J, JF, -2
     L = L + 1
     ..................
   4 CONTINUE
   3 CONTINUE
   2 CONTINUE
   1 CONTINUE
```

The location where the terms of indices $(i, j, k, m)$ should be stored is computed as $\phi(i,j) + ((i-k)(j+1) + j + 2 - m)/2$, where $\phi$ is a function which depends on $i$ and $j$. It is computed initially and stored in a table. Of course, the expression of $\phi$ is different for $x$ and for $z$ (for $y$ is the same as for $x$) but each part takes only a half table. The number of terms for $z$ up to order $n$ is slightly less than for $x$ and $y$, but we do not save this space. For $x$ and $y$ the total number of terms up to order $n$, $n$ odd, is given by $(j^4 + 7j^3 + 20j^2 + 26j + 6)/6$, where $j = (n-1)/2$. For $n = 15$, 25 or 35 this takes the respective values 995, 6005 or 20690.

The functions $d$, $f$, $d^2$, $df$, $f^2$ are stored with the element $(i,j)$ at the place $((i+j)(i+j-2) + 4j + 8)/8$. If $n$ is odd we only require room for them up to order $n-1$ and each one takes $(j+1)(j+2)/2$ positions, where again $j = (n-1)/2$.

Faced to the alternative more CPU time or more space in memory we have selected the second one (see later). Hence the functions $T_m$, $R_m$ are kept in memory for all the required values of $m$. It is possible to keep just two of them for the recurrence. Then $T_m$, $R_m$ are stored in $T(*,m)$, $R(*,m)$, respectively, where $*$ ranges from 1 to the dimension of $x$ for order $n$. The maximum value of $m$ is $n$ for $T$ and $n-1$ for $R$. We list the routines used to obtain $x$, $y$, $z$, $d$, $f$ and we note that at order $n$ only terms of degree $n$ ( or $n-1$ for $d$, $f$, etc) are computed, because all the terms of lower degree, required for the intermediate computations, are kept in memory.

**Subroutine TAULA** Computes the auxiliar function $\phi$ up to order $n$.

**Subroutine GAMCES** Gives $\gamma$, $c_j$, $j = 1, 2, ..., n+1$, $\omega_1$, $\omega_2$, $\overline{k}$ from $\mu$ and the index (1 or 2) of $L_i$.

**Subroutine RO2CN** Computes the terms of a given order of $\rho^2$ from $x$, $y$, $z$.

**Subroutine PRODN** Produces the terms of a given order of the product of two functions.

**Subroutine SUMN** Similar to `PRODN` but for the sum. It also allows to multiply the input functions by a number.

**Subroutine TRN** Computes the terms of a given order of $T_m$, $R_{m-1}$, $m = 2, ...,$ current order, from $x$ and $\rho^2$.

**Subroutine GET** Copies $T(*, j)$ or $R(*, j)$ to a single vector.

**Subroutine GETN** Same as `GET` but just for the terms of a given order.

**Subroutine PUTN** Puts the terms of order $n$ of a vector on $T(*, j)$ or $R(*, j)$.

**Subroutine MNPN** Computes the terms of a given order of $M$, $N$, $P$ from $c_j, j \geq 3$, $T_j$, $R_{j-1}$, $j \geq 2$, and $y$, $z$.

**Subroutine DERN** Gives the terms of order $n$ of the required derivatives of $x$, $y$, $z$ with respect $\theta_1$ and $\theta_2$.

**Subroutine DDFFN** Produces the terms of required order of $d^2$, $2df$, $f^2$ from $d$, $f$.

**Subroutine MNPBN** From $M$, $N$, $P$, the previous derivatives and products $d^2$, etc. gives $\overline{M}$, $\overline{N}$, $\overline{P}$.

**Subroutine PROLCN** Entering a series as $x$, $y$, $z$ and another one as $d$, $f$, $d^2$, etc. gives the product.

**Subroutine SOLVEN** Solves the terms of order $n$ of $x$, $y$ and $z$ and, if $n$ is odd, the ones of $d$, $f$ of degree $n-1$.

**Subroutine OUT** Outputs the results.

We refer to the listings of the programs for a more detailed description and also to the Users and Programmers Guide [4].

Finally we make some a priori comments on the running space and CPU time. The programs have been structured in such a way that we can modify a few parameter instructions by setting

$$NG = (j^4 + 7j^3 + 20j^2 + 26j + 6)/6, \quad NG1 = 2j + 1, \quad NG2 = (j^2 + 3j + 2)/2,$$

if the maximal order to run the program is $n = 2j + 1$. One needs $20 + 2n$ vectors of dimension $NG$ and a few ones of dimensions $NG1$, $NG2$. The total space required (in double precision) is $(4j^5+56j^4+236j^3+691j^2+881j)/6+ctant$. If $n = 35$ this is close to 15.5 Mbyte, and it is close to 4 Mbyte if $n = 25$. To keep in mind a simpler expression, the required space up to order $n$ is proportional to $n^4(n+23)$ for $n$ large. An analysis of the program shows that the CPU time is roughly proportional to $n^8$. Had we stored just two consecutive terms of $\{T_m\}$, $\{R_m\}$ the required space would be proportional to $n^4$ but the computing time would be to $n^9$, becoming prohibitive for large orders.

## 2.3.2 Results and Formal Tests

The program described in the above section, named **LISFIL**, has been run for several values of $n$, for $L_1$ and $\mu = 0.3040357143 \times 10^{-5}$ (the adopted value of the Barycenter / (Sun + Barycenter) mass ratio). The CPU times on an HP 9000 / 835S computer are shown for different $n$.

| n | 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 | 25 | 35 |
|---|---|---|---|---|---|---|---|---|---|---|
| CPU in sec. | 0.4 | 1.4 | 3.6 | 9.1 | 20.9 | 44.6 | 91.4 | 176.4 | 328.0 | 4338.7 |

Table 2.1

In [4] the values of $d_{i,j}$, $f_{i,j}$ up to order 21 and of $x$, $y$, $z$ up to order 5 are given.

To have a first look to the results Figure 2.1 shows the analytical orbits computed to order 35 for the following values of $(\alpha, \beta)$: $(0.1, 0.05)$, $(0.1, 0.3)$, $(0.05, 0.3)$, $(0, 0.3)$. In all the plots the $(y, z)$ projection is displayed and the unit of length is $\gamma$ (roughly $1.5 \times 10^6$Km).

The results have been formally tested in the following sense. With the solution available in the form of (2.10) it has been substituted in (2.6). The left and right hand sides must be equal. To allow for rounding errors if one term to the left and to the right should be exactly zero (because of the symmetries) we have asked it to be less than $10^{-15}$ in absolute value. Otherwise it has been required that the absolute value of the difference be less than $10^{-10}$ times the maximum of the absolute values of the corresponding terms in both sides. Program TESLAN did the check and found that up to order 16 all the elements passed the test. For increasing orders, specially beyond order 30, some 500 terms (including $x$, $y$, $z$) did not passed the test, reaching relative errors of $10^{-4}$ for a few ones of them. But most of those are due to terms which are rather small.

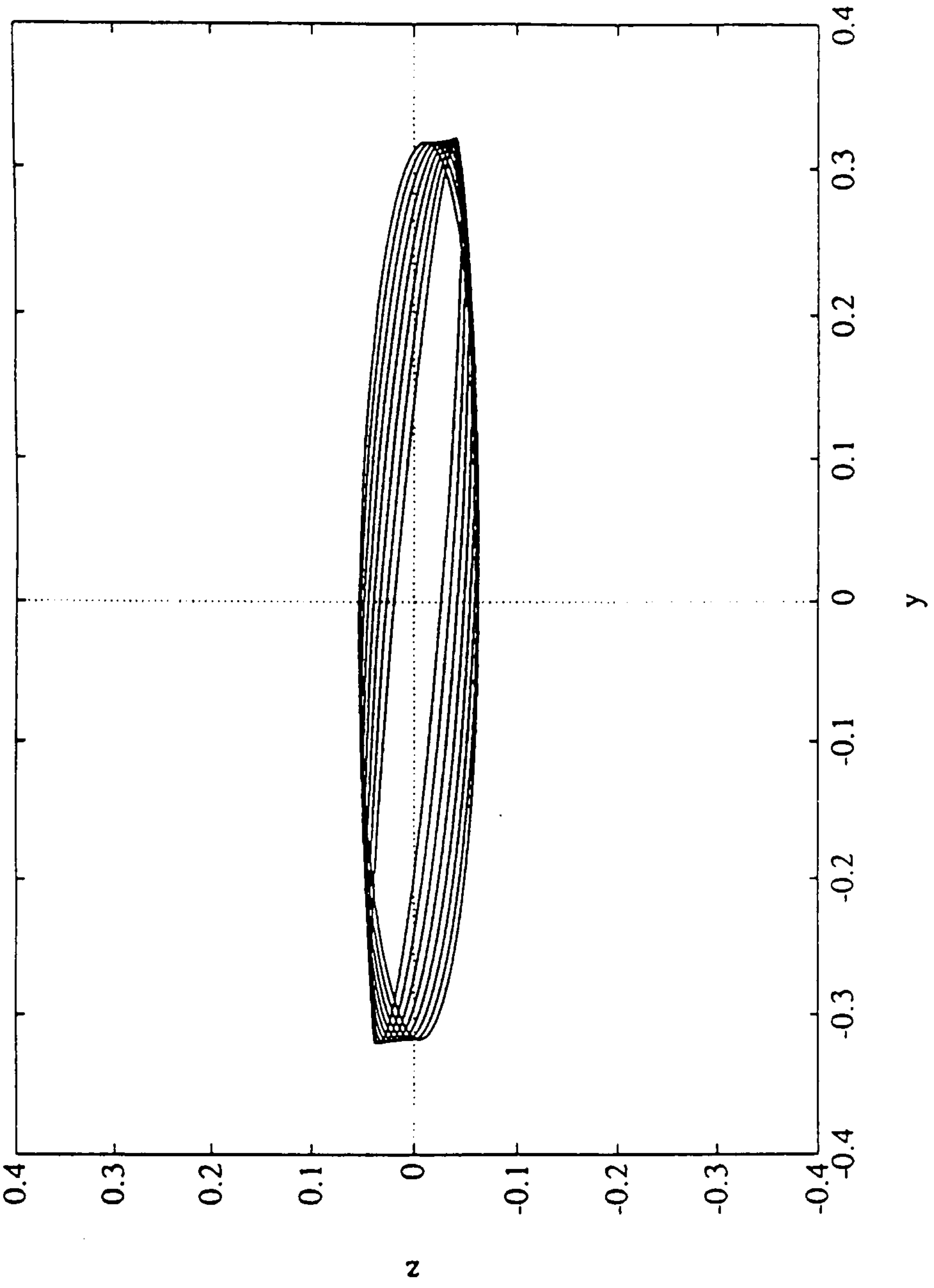


Figure 2.1.a Lissajous orbit for $\alpha = 0.1$, $\beta = 0.05$, $\varphi_1 = \varphi_2 = 0$. Final time $t_f = 25$.

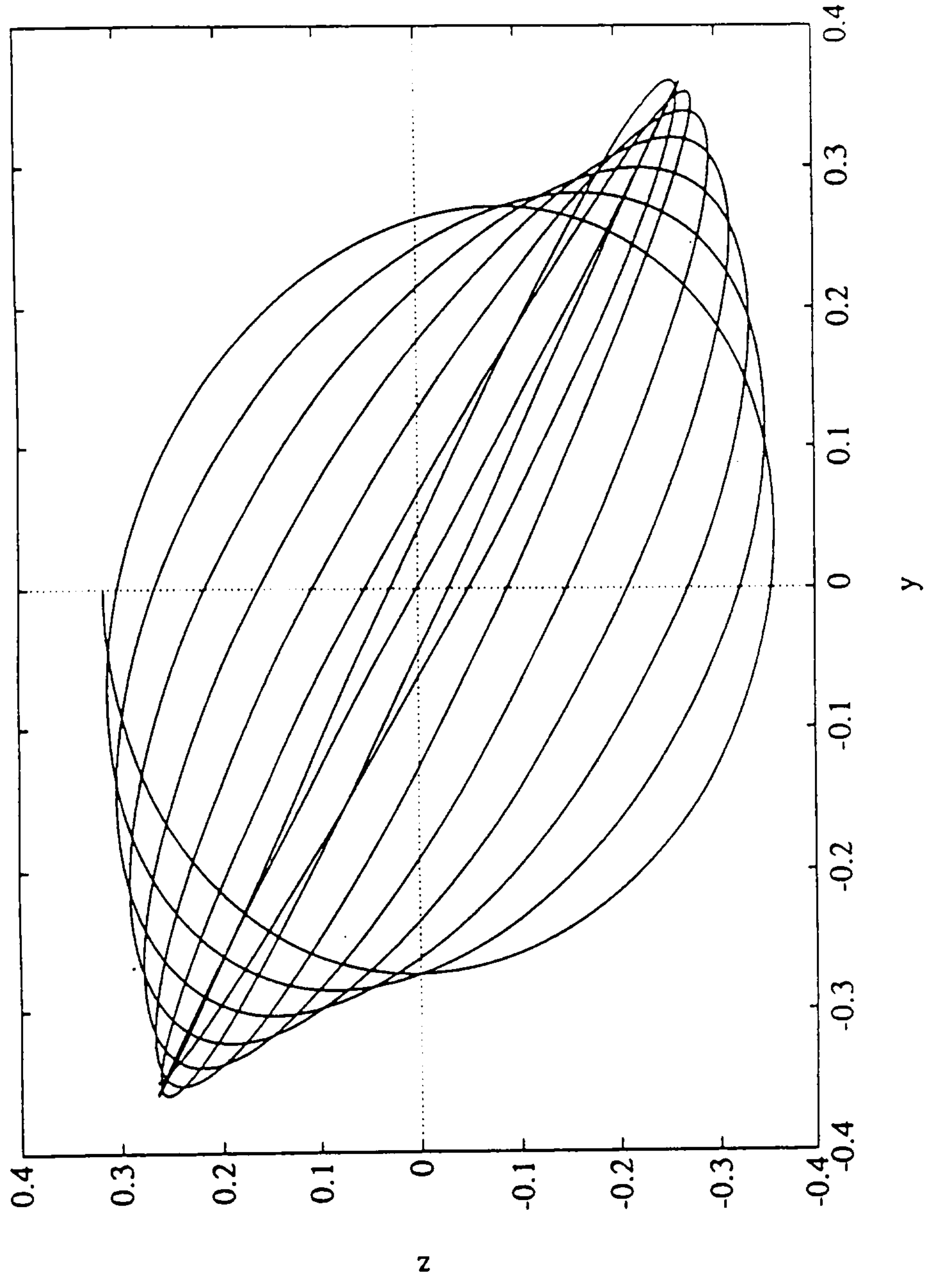


Figure 2.1.b Lissajous orbit for $\alpha = 0.1$, $\beta = 0.3$, $\varphi_1 = \varphi_2 = 0$. Final time $t_f = 25$.

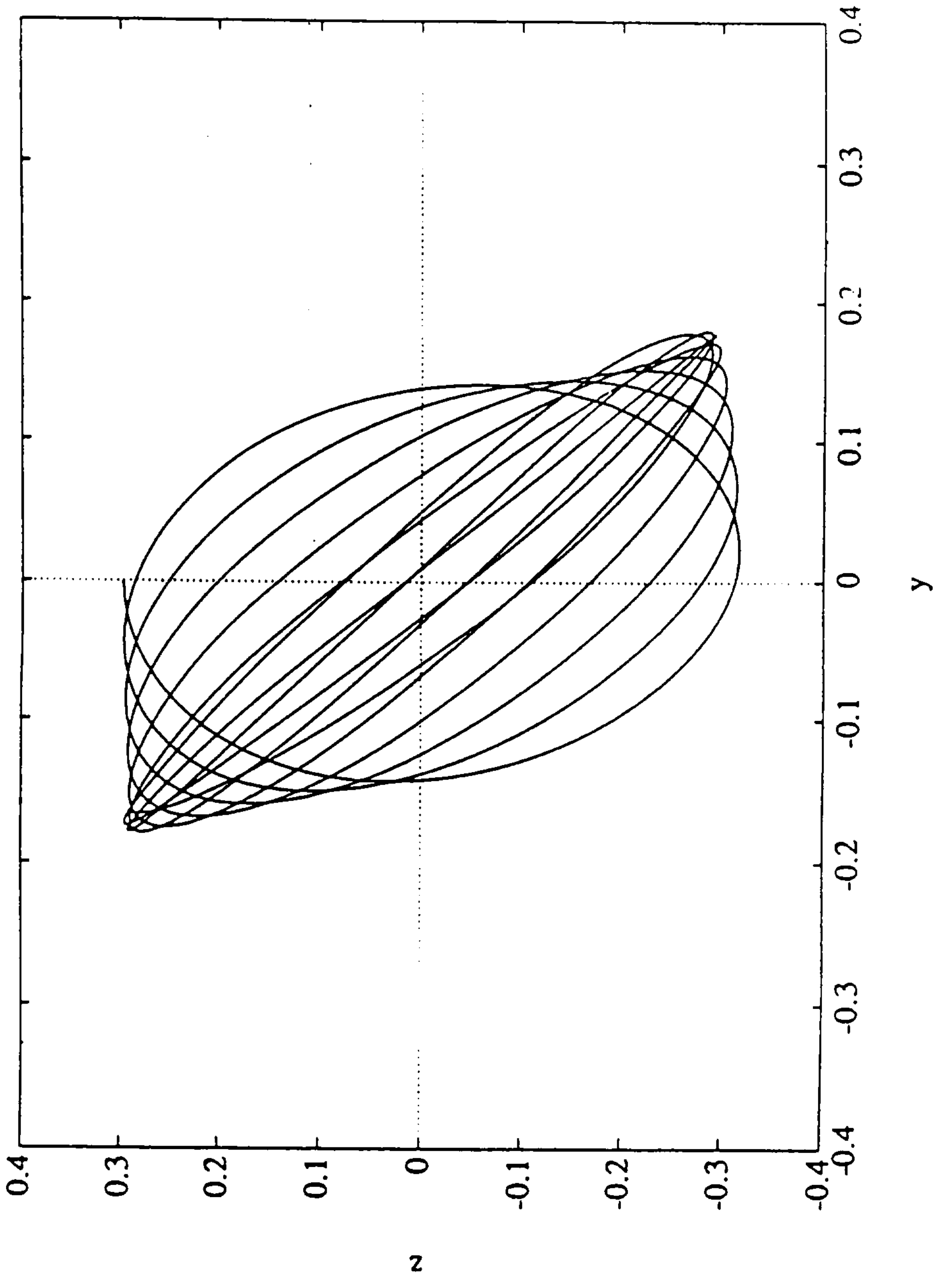


Figure 2.1.c Lissajous orbit for $\alpha = 0.05$, $\beta = 0.3$, $\varphi_1 = \varphi_2 = 0$. Final time $t_f = 25$.

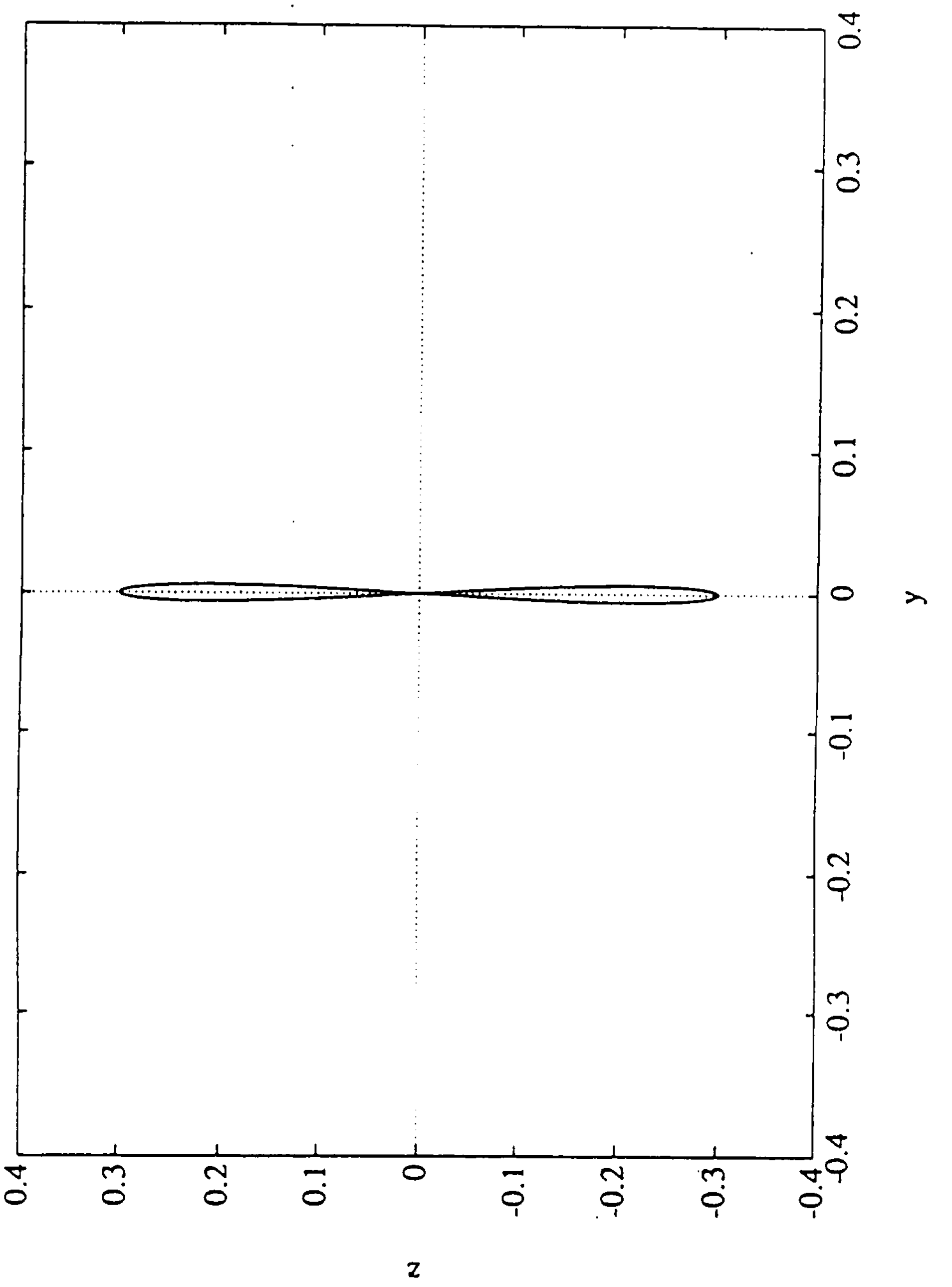


Figure 2.1.d Vertical periodic orbit $\alpha = 0$, $\beta = 0.3$. Period $= 3.14117812$ $\varphi_1 = \varphi_2 = 0$.

### 2.3.3 Numerical Tests

The analytical solutions of (2.1) have been checked against numerical solutions by program TESLNU as follows. Given $\alpha$, $\beta$, $\phi_1$, $\phi_2$ and some order $n$ we obtain the initial conditions for $X$, $Y$, $Z$, $\dot{X}$, $\dot{Y}$, $\dot{Z}$ at $t = 0$. Then the numerical integration has been started and the distance between the numerical solution and the analytical solution is computed (in the phase space) at regular intervals of 0.01 time units ($2\pi$ time units = 1 year). This test has been done for orders 15, 25 and 35, for $\alpha = 0.05$ and 0.10, for $\beta = 0.05(0.05)0.30$ and $\phi_1 = j_1\pi/4$, $j_1 = 0, \ldots, 7$, $\phi_2 = j_2\pi/4$, $j_2 = 0, \ldots, 3$, (the values for $\phi_2 \geq \pi$ are not necessary due to the symmetry). A summary of results can be found in [4].

From the test we can extract the following conclusions:

1. Going from order 15 to order 25 the results improve by a factor of the order of 1000.

2. Going from order 25 to order 35 the results do not improve for $\alpha = 0.05$, $\beta \leq 0.15$. This is due to the fact that the errors of the theory are of the order of magnitude of the rounding error. Otherwise they improve by a factor of 100.

3. For order 35 the errors in the phase space, expressed as a distance ( 1 unit = 1 AU) after 6.28 time units are less than 2Km if $\alpha \leq 0.05$, $\beta \leq 0.25$. For $\alpha = 0.1$ or $\beta = 0.3$ can reach 30000 Km, but only 14 Km after 3.14 time units.

4. The quotients of the distances at t = 6.28 and t = 3.14 ( and also at t = 4.71 and t = 1.57) are between 2000 and 3000 in all cases, at most. This reflects the inherent instability of those orbits due to the instability of $L_1$.

## 2.4 On the Convergence of the Series

If we consider the Hamiltonian formalism, the second order terms, in suitable variables, are of the form

$$H = \lambda x_1 y_1 + \frac{1}{2}\omega_1(x_2^2 + y_2^2) + \frac{1}{2}\omega_2(x_3^2 + y_3^2),$$

where $\lambda x_1 y_1$ is the responsable of the instability and the other two are two harmonic oscillators. Higher order terms couple the three modes. If we skip the instability (see the next section) we have a two degrees of freedom Hamiltonian whose lower order terms are a couple of harmonic oscillators. Then we shall have, close to the origin, a big set of KAM tori[1]. But, between those tori, there are very narrow stochastic zones (because the resonances are dense). Hence we shall have divergence everywhere (except at the origin, but this is not very useful).

However, as said in the Introduction, small divisors shall show up only at high orders (except the one due to the 1 to 1 resonance) because, at the origin $\omega_1/\omega_2$ is close to 29/28. The high order resonances have a very small stochastic zone and the effect is only seen after a big time interval.

Looking for practical applications with the analytical results up to order 35 we did the following test: Let $\alpha = \sigma \cos \epsilon$, $\beta = \sigma \sin \epsilon$. By substituting in $d$ and $f$ we found

$$d = \sum_{j=0, j\,even}^{34} c_{1,j}\sigma^j, \; f = \sum_{j=0, j\,even}^{34} c_{2,j}\sigma^j, \; c_{i,j} = c_{i,j}(\epsilon).$$

Let

$$r(\epsilon) = \frac{0.75}{\sup_{i,j} \mid c_{i,j} \mid^{1/j}}.$$

This gives a "practical" radius of convergence. The factor 0.75 ensures that, if $r(\epsilon)$ were the true radius of convergence, the remainder would be less than $10^{-4}$.

Figure 2.2 shows the "domain of practical convergence" with this criterion when $\epsilon$ ranges from 0 to $\pi/2$.

One can also check that in this domain $d$ decreases and $f$ increases if $\alpha$ increases or if $\beta$ decreases, provided $\beta$ is not too large. Otherwise both $d$ and $f$ decrease when $\alpha$ increases or $\beta$ decreases. Rather than to give the plots of $d$ and $f$ in terms of $\alpha$ and $\beta$ we computed more useful things. Let $\Delta = d - f$ and $q = d/f$. The lines $q = ctant \in \mathbb{Q}$ give resonances, and the curves $\Delta = ctant$ give constant difference of frequencies, to be used in the discussion. The Figure 2.2 also displays curves of constant $q$ (dotted), for $q$ ranging from 809/800 to 839/800, and curves $\Delta = ctant$, for $\Delta$ ranging from 0.025 to 0.095 (solid lines).

Assume that the real problem is the RTBP and we use the formal solution as nominal orbit. Due to the instability one should do manoeuvres from time to time. Hence we can allow for some small error in the nominal orbit. This one will be corrected when the instability is. Hence we can take some values of $\alpha$ and $\beta$. Search for $\phi_1$ and $\phi_2$ such that after 3 months the difference between the analytical and numerical solution is maximal (in the phase space). We decide that if this difference is less than $10^{-7}$ the values of $(\alpha, \beta)$ are admissible. The boundary of the region of admissible parameters is also displayed in Figure 2.2, and it gives

[1] This refers to the invariant tori which persist when we pass from an integrable Hamiltonian system to a small perturbation of it. The existence of these tori is ensured by the celebrated Kolmogorov-Arnold-Moser Theorem. The measure of the set of points, in the phase space, which were in a torus for the integrable system and are not in an invariant torus (but in some "chaotic" region) when the perurbation is added, goes to zero when the size of the perturbation does.

another "practical use of the method" zone. It is slightly smaller than the region of "practical" convergence found before for $d$ and $f$.

As a conclusion, despite the nonconvergent character of the series they can be used, for some practical purposes, in some big region of the $\alpha$, $\beta$ parameter range. They produce approximations to invariant tori: the orbits on those ones are quasiperiodic if $d/f \notin \mathbb{Q}$ and of long period if $d/f \in \mathbb{Q}$. They can be seen as nonlinear Lissajous orbits.

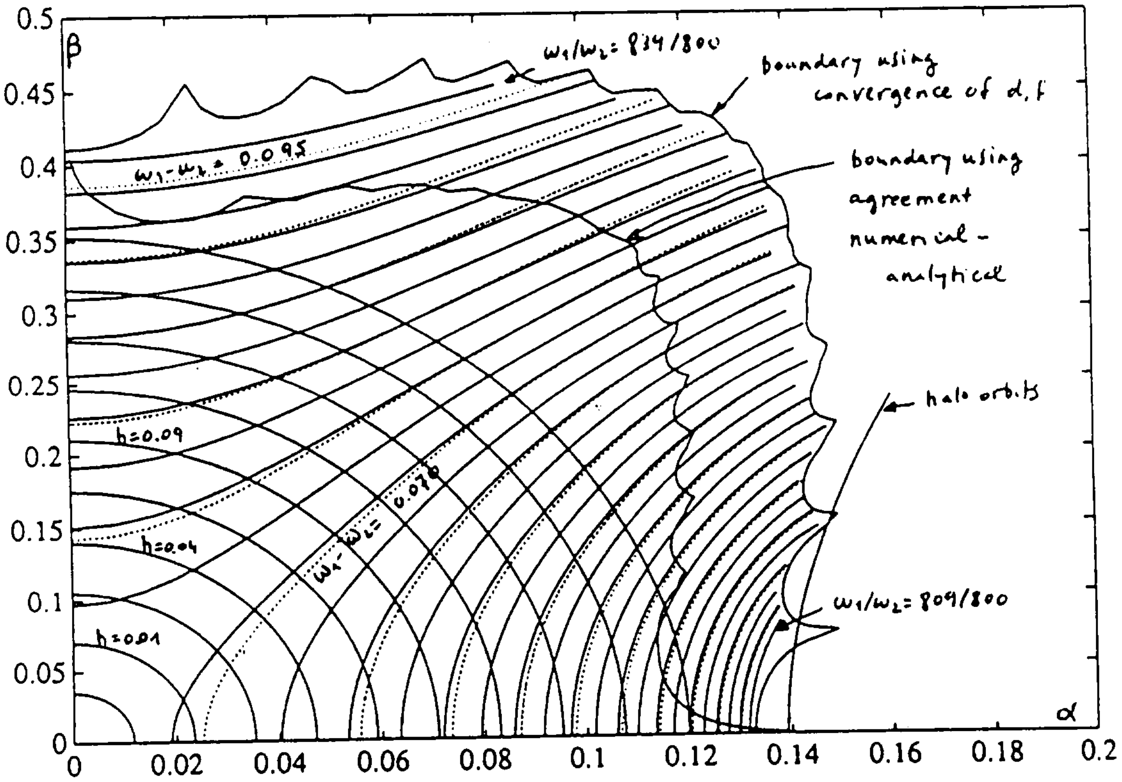


Figure 2.2

Finally we would like to point out that the expressions found for the Lissajous orbits remain valid for $\beta = 0$ (Lyapunov periodic orbits) and for $\alpha = 0$ (vertical periodic orbits). In these cases no problems due to small divisors show up and the restricted series are convergent. Working at order 35, for $\beta = 0$ values of $\alpha$ up to 0.35 give very good approximations. If $\alpha = 0$, $\beta$ can reach values up to 1.13.

## 2.5 Towards a Description of the Neighbourhood of $L_1$

In this section we sketch the ideas to be developed in the subsequent ones. First of all we think about skipping the unpleasant unstable terms of the Hamiltonian. This is accomplished by the reduction to the central manifold $W^c$ of dimension 4. We describe how to carry out the computations. After perfoming a linear change of variables going from $x$, $y$, $p_x$, $p_y$ to $x_1$, $x_2$, $y_1$, $y_2$ and putting $x_3 = z$, $y_3 = p_z$, we have that the Hamiltonian $K$ given by 2.7 can be written as

$$M = \lambda x_1 y_1 + \frac{1}{2}\omega_1(x_2^2 + y_2^2) + \frac{1}{2}\omega_2(x_3^2 + y_3^2) + \sum_{j\geq 3} M_j(x_1, x_2, x_3, y_1, y_2, y_3).$$

Then we pass to an intermediate normal form just trying to kill in $M$ all the terms of the form $x_1^{k_1} y_1^{l_1}$ with $k_1 \neq l_1$. This is possible because there are not problems of small divisors. The denominators appearing in the generating function (using some complexification to reduce the amount of computation) are of the form

$$(k_1 - l_1)\lambda + \sqrt{-1}\left((k_2 - l_2)\omega_1 + (k_3 - l_3)\omega_2\right),$$

with modulus bounded from below by $\lambda$. Let $\widetilde{M}$ be this intermediate Hamiltonian. By rearranging terms it can be written as

$$\widetilde{M} = M^0(x_2, x_3, y_2, y_3) + \sum_{j\geq 1}(x_1 y_1)^j M^j(x_2, x_3, y_2, y_3).$$

It is immediately seen that $I_1 = x_1 y_1$ is a first integral. The reduction to the center manifold is obtained by setting $I_1 = 0$. So it remains a two degrees of freedom Hamiltonian, $M^0$, and furthermore

$$M^0 = \frac{1}{2}\omega_1(x_2^2 + y_2^2) + \frac{1}{2}\omega_2(x_3^2 + y_3^2) + \sum_{j\geq 3} M_j^0(x_2, x_3, y_2, y_3). \qquad (2.18)$$

We point out that if in (2.18) we put $x_3 = y_3 = 0$ we have an invariant set under the flow associated to $M^0$. This is now a 1 degree of freedom Hamiltonian. The orbits are simply the Lyapunov periodic orbits around $L_1$.

The levels of energy of $M^0$ are also represented on Figure 2.2. Each level corresponds to an $S^3$ sphere. When $\beta = 0$ we obtain a Lyapunov orbit. When $\alpha = 0$ an orbit as the one displayed in Figure 2.1 d). For $\alpha \neq 0$ and $\beta \neq 0$ we have either tori or stochastic zones. The orbits in Figure 2.1 a), b) and c) show (approximations of) typical orbits in the tori.

Keeping $M^0 = h$ fixed we can use $z = x_3 = 0$ with $\dot{z} = y_3$ as surface of section. This section, $\Sigma$, is not global because it fully contains the corresponding

Lyapunov orbit. But we can think that all the points of this orbit (in the boundary of the section) are identified. So we should obtain an $S^2$ sphere (the classical Hopf fibration) that can be seen as an space of "osculating orbits". For some value of $h$, $h_H$ say, the level $H^0 = h$ contains the Lyapunov orbit bifurcating to halo orbits. From this value on the section $\Sigma$ contains two fixed points associated to the two (symmetrical) halo orbits. It is known that, at least for moderate values of $h$, they are of elliptic type. At least for values of $h$ not too far from $h_H$ we have in $S^2$ four fixed points. Two of them are the Lyapunov orbit (hyperbolic) and the almost vertical periodic orbit as in Figure 2.1.d (elliptic). The other two are the halo orbits (elliptic). So the sum of the indices (1 for elliptic and -1 for hyperbolic) agrees with the Euler-Poincaré characteristic of $S^2$.

## 2.6 Discussion on the Use of Lissajous Orbits

The Lissajous orbits should avoid them, for practical purposes, a neigbourhood of the Earth-Sun system line (to prevent an spacecraft like SOHO to stay for some time in a region where the radio communications are distorted by the noise coming from the Sun). Let $l$ be the minimum admissible angle between the Earth-Sun and Earth-spacecraft lines. For technical reasons $l$ should be, at least, 3 degrees.

On the other hand orbits making small excursions on the $y$ and $z$ directions are desirable to minimize the cost of the antenna. The admissible halo orbits give excursions of the order of 26° in the $y$ direction to each side of the Earth-Sun line. Hence we can think about the use of Lissajous orbits. However those ones have the problem that they shall pass in front of the solar disk due to the increasing delay between the phases. The standing problems are: How long it takes to pass too close to the Earth-Sun line and, if they do this, which kind of manoeuvre one should do to shift the motion to the admissible region. The answer to those questions is given in what follows for the RTBP.

To avoid too frequent passages near the Sun direction one should require that the two basic frequencies of the Lissajous orbits, $\omega_1$ and $\omega_2$, be as close as possible. In other words, the difference $\Delta = d - f = \Delta(\alpha, \beta)$ should be small. Looking at Figure 2.2 this is seen to be not compatible with the requirement that $\alpha$ and $\beta$ be small. A more detailed discussion follows.

Suppose we start with initial phases $\phi_1 = \phi_2 = 0$ at $t = 0$, with amplitudes $\alpha$ and $\beta$. Then, after some time, $T$, we have that the relation

$$\frac{(x^2 + y^2)^{1/2}}{1 - x} = \tan l$$

holds for the first time, where $x$, $y$, $z$ denote the position of the spacecraft in normalized coordinates. Furthermore we can also look, in the time interval $[0, T]$, for the maximum values of

$$a_y = \tan^{-1}\frac{|\,y\,|}{1-x} \quad \text{and} \quad a_z = \tan^{-1}\frac{|\,z\,|}{1-x}.$$

Keeping $l = 3^o$ we scanned several couples $(\alpha, \beta)$. We keep just two such couples:

1. If $\alpha = 0.0484$ and $\beta = 0.14874$ then $a_y \simeq a_z \simeq 9^o$ and $T = 14.94$ units. (We recall that $2\pi$ units $= 1$ year).
2. If $\alpha = 0.0386$ and $\beta = 0.1228$ then $a_y \simeq a_z \simeq 7^o.25$ and $T = 14.87$ units.

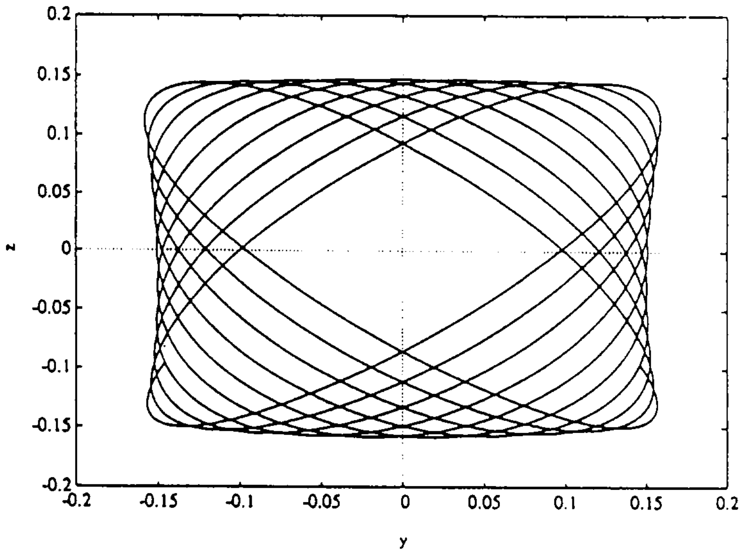


Figure 2.3.a

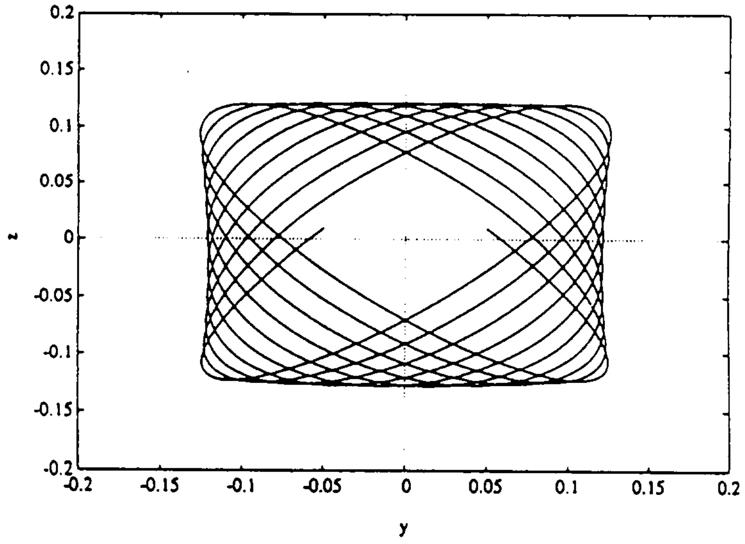


Figure 2.3.b

Going backwards in time we obtain symmetrical results. Hence, in case 1 (resp. 2) we can stay in the admissible region for 29.88 units (resp. 29.74 units), equivalent to 4.75 years.

However it is not good to wait for this time to do a reasonable manoeuvre. This one should be done, in the RTBP, when $y = 0$. Using the symmetry of the solutions we can match an orbit with $(x, 0, z, \dot{x}, \dot{y}, \dot{z})$ to another with $(x, 0, z, \dot{x}, \dot{y}, -\dot{z})$ with $\Delta v = 2 \mid \dot{z} \mid$. Doing this we should do a manoeuvre with $\Delta v \simeq 143$ m/s in case 1 ( 121 m/s in case 2) every 4.31 years (4.30 years in case 2) to keep the spacecraft on the admissible region.

The orbits for cases 1 and 2 are shown in figure 2.3.

In the real problem the quasiperiodic oscillations added to the Lissajous orbit imply that the admissible time between manoeuvres shall be a little bit less than the one in the RTBP and the $\Delta v$ shall be probably slightly larger, but this has to be explored.

As a final comment, it will follow from the remaining part of this Chapter that around the halo orbits there are small invariant tori. This provides also some amount of additional freedom when looking for the RTBP approximation of the nominal orbit for the SOHO mission and the related transfer orbits.

## 2.7 Effective Reduction to the Central Manifold

### 2.7.1 The Method

To start we have the Hamiltonian (2.7). First of all it should be put in the form of 2.5 by a linear canonic change and then to complex form in order to have the quadratic part as

$$\lambda q_1 p_1 + i(w_1 q_2 p_2 + w_2 q_3 p_3).$$

This is done in one step by using

$$\begin{pmatrix} x \\ y \\ y \\ p_x \\ p_y \\ p_z \end{pmatrix} = A \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ p_1 \\ p_2 \\ p_3 \end{pmatrix},$$

where

$$\begin{aligned} a_{11} &= 2\lambda\alpha, \\ a_{12} &= 2w_1 i\beta, \\ a_{13} &= 0, \\ a_{14} &= -2\lambda\alpha, \\ a_{15} &= -2w_1 i\beta, \end{aligned}$$

$$\begin{aligned}
a_{16} &= 0,\\
a_{21} &=( \lambda^2-1-2c_2)\alpha,\\
a_{22} &= (-w_1-1-2c_2)\beta,\\
a_{23} &= 0,\\
a_{24} &= (\lambda^2-1-2c_2)\alpha,\\
a_{25} &= (-w_1^2-1-2c_2)\beta,\\
a_{26} &= 0,\\
a_{31} &= 0,\\
a_{32} &= 0,\\
a_{33} &= \epsilon,\\
a_{34} &= 0,\\
a_{35} &= 0,\\
a_{36} &= 0,\\
a_{41} &= (\lambda^2+1+2c_2)\alpha,\\
a_{42} &= (-w_1+1+2c_2)\beta,\\
a_{43} &= 0,\\
a_{44} &= (\lambda^2+1+2c_2)\alpha,\\
a_{45} &= (-w_1^2+1+2c_2)\beta,\\
a_{46} &= 0,\\
a_{51} &= \lambda(\lambda^2+1-2c_2)\alpha,\\
a_{52} &= w_1 i(-w_1+1-2c_2)\beta,\\
a_{53} &= 0,\\
a_{54} &= -\lambda(\lambda^2+1-2c_2)\alpha,\\
a_{55} &= -w_1 i(-w_1^2+1-2c_2)\beta,\\
a_{56} &= 0,\\
a_{61} &= 0,\\
a_{62} &= 0,\\
a_{63} &= -w_2 i\epsilon,\\
a_{64} &= 0,\\
a_{65} &= 0,\\
a_{66} &= w_2 i\epsilon,
\end{aligned}$$

where $\lambda$, $iw_1$, $iw_2$ are the eigenvalues of $K$ at the origin, $c_2$ is given in (2.3) and $\alpha$, $\beta$, $\epsilon$, are:

$$\begin{aligned}
\alpha &= (2\lambda(\lambda^2(4+3c_2)+4+5c_2-6c_2^2))^{-1/2},\\
\beta &= (2w_1 i(-w_1^2(4+3c_2)+4+5c_2-6c_2^2)^{-1/2},\\
\epsilon &= (2w_2 i)^{-1/2}.
\end{aligned}$$

To generate $K$ as a polynomial up to some degree it is better to work directly with the new complex coordinates $q_j$, $p_j$, j = 1, 2, 3 by using the recurrent relations.

Indeed $x$, $y$, $z$, $p_x$, $p_y$, $p_z$ are homogeneous linear polynomials in the variables $q_j$, $p_j$, j = 1, 2, 3. The quadratic part of $K$ is easily obtained. Then to compute $Q_n(x,y,z) = \rho^n P_n(x/\rho)$, where $\rho = x^2 + y^2 + z^2$ as before, we start with

$$\begin{aligned} Q_1(x,y,z) &= x, \\ Q_2(x,y,z) &= x^2 - \frac{1}{2}y^2 - \frac{1}{2}z^2, \end{aligned}$$

and use the recurrent relation (the one of the Legendre polynomials)

$$Q_n(x,y,z) = \left(2 - \frac{1}{n}\right) x Q_{n-1}(x,y,z) - \left(1 - \frac{1}{n}\right)(x^2 + y^2 + z^2) Q_{n-2}(x,y,z).$$

Then we want to obtain a reduction to a partial normal form (P.N.F.). We can restrict the canonical transformation to one given as the flow time one of a Hamiltonian $G$ starting with terms of degree 3. More concretely, let $M$ (we keep the same name) be the Hamiltonian in the $(q,p)$ variables:

$$M = \lambda q_1 p_1 + i(w_1 q_2 p_2 + w_2 q_3 p_3) + \sum_{k \geq 3} M_k$$

where $M_k$ are homogeneous polynomials in $(q,p)$ of degree $k$. The first part of $M$ will be denoted by $M_2$. On the other hand let

$$G = \sum_{k \geq 3} G_k(q,p),$$

and $\varphi_t^G$ the time $t$ flow using $G$ as Hamiltonian. Define

$$(Q^T, P^T)^T = \varphi_{t=-1}^G((q^T, p^T)^T) = \varphi_{t=1}^{-G}((q^T, p^T)^T).$$

Then, as $G$ is autonomous,

$$(q^T, p^T) = \varphi_{t=1}^G((Q^T, P^T)^T).$$

Obviously the passage from $(q,p)$ to $(Q,P)$ is canonical (Hori's method). Furthermore, if

$$M(q,p) = M(q(Q,P), p(Q,P)) = \widetilde{M}(Q,P),$$

one has

$$\widetilde{M}(q,p) = \varphi_{t=1}^G M(q,p) = M(q,p) + \frac{1}{1!}\frac{d}{dt}M(q,p) + \ldots + \frac{1}{k!}\frac{d^k}{dt^k}M(q,p) + \ldots$$

one has to compute the succesive time derivatives of $M(q,p)$ along the orbits of $G$. This is done by using the Poisson brackets:

$$\frac{d^k}{dt^k}M(q,p) = [\frac{d^{k-1}}{dt^{k-1}}M(q,p), G] \equiv M^{(k)}(q,p).$$

First of all

$$M^{(1)}(q,p) = \sum_{k\geq 2, j\geq 3} [M_k(q,p), G_j(q,p)] = \sum_{r=3} M_r^{(1)}(q,p),$$

where

$$M_r^{(1)} = \sum_{s=2}^{r-1}[M_k, G_{r+2-s}].$$

Then

$$M^{(k)}(q,p) = \sum_{r\geq k+2} M_r^k(q,p),$$

where

$$M_r^k = \sum_{s=k+1}^{r-1} [M_s^{k-1}, G_{r+2-s}].$$

Of course $\widetilde{M}_2 = M_2$. At order 3 we have

$$\widetilde{M}_3 = M_3 + \frac{1}{1!}M_3^{(1)} = M_3 + [M_2, G_3].$$

This is the homological equation to be solved for $\widetilde{M}_3$ and $G_3$. As $M_3$ is of the form

$$\sum m_{3kl} q_1^{k_1} q_2^{k_2} q_3^{k_3} p_1^{l_1} p_2^{l_2} p_3^{l_3},$$

with

$$k_i, l_i \geq 0,\ i = 1,2,3,\ \sum_{i=1}^{3}(k_i + l_i) = 3,$$

we keep in $\widetilde{M}_3$ all the terms with $k_1 = l_1$ and try to cancel the remainder with a suitable $G_3$. We note also that if a homogeneous polynomial, $F$, is of the form

$$F = \sum f_{k,l} q^k p^l,$$

then

$$[M_2, F] = \sum f_{k,l}(\lambda(l_1 - k_1) + iw_1(l_2 - k_2) + iw_2(l_3 - k_3))q^k p^l.$$

Therefore we should choose

$$G_3 = \sum_{l_1 \neq k_1} m_{3kl}(\lambda(l_1 - k_1) + iw_1(l_2 - k_2) + iw_2(l_3 - k_3))^{-1} q^k p^l.$$

Here $q_k$ denotes $q_1^{k_1} q_2^{k_2} q_3^{k_3}$ as usual, and we remark that the divisors have modulus bounded from below by $\mid \lambda \mid$.

Let us describe how to proceed to order four, the other cases being similar. We have

$$\widetilde{M}_4 = M_4 + \frac{1}{1!}M_4^{(1)} + \frac{1}{2!}M_4^{(2)}.$$

As $G_3$ is available we have $M_3^{(1)}$ and, hence, $M_4^{(2)} = [M_3^{(1)}, G_3]$. However $M_4^{(1)} = [M_2, G_4] + [M_3, G_3]$. Putting this together we have

$$\widetilde{M}_4 = M_4 + [M_3, G_3] + M_4^{(2)} + [M_2, G_4].$$

In the right hand side everything is known but the last term. We write (2.7.1) as

$$\widetilde{M}_4 = F_4 + [M_2, G_4], \quad F_4 \text{ known}.$$

This is solved for $\widetilde{M}_4$ and $G_4$ as we did for (2.7.1).

In this way we obtain $\widetilde{M}$ and $G$ to the desired order. Having $G$ we can compute the direct and inverse transformations from $(q, p)$ to $(Q, P)$ if needed. The new Hamiltonian $\widetilde{M}$ is of the form

$$\widetilde{M} = \sum_{s \geq 2} \widetilde{M}_s, \quad \widetilde{M}_s = \sum_{|k|+|l|} = \widetilde{m}_{skl} Q^k P^l, \quad \text{P.N.F.}$$

with $k_1 \neq l_1$. Setting $q_1 = p_1 = 0$ we obtain the reduced Hamiltonian $\widetilde{M}^0$, free from the saddle part. Finally it is convenient to go back to new real coordinates instead of the complex ones $(\overline{Q}, \overline{P})$, $\overline{Q} = (Q_2, Q_3)$, $\overline{P} = (P_2, P_3)$. They are denoted by $(\xi, \eta)$, $\xi = (\xi_1, \xi_2)$ and $\eta = (\eta_1, \eta_2)$.

We shall denote by $H^0$, in accordance with section 9, the final Hamiltonian.

### 2.7.2 The Program and the Results

The previous method has been implemented in the program LISWCH. The program is split in several routines which perform each one of the steps. The only input are the values of $\mu$ and the maximal order $n$. Several dimensions depend on $n$. They are

$$\binom{n+6}{6}, \binom{n+4}{4}, \binom{n+3}{5}, \binom{n+4}{5},$$

$$\binom{n+5}{5}, 6\binom{n+6}{7} + \binom{n+6}{6} - 55.$$

From the input several constants are computed, like $\lambda$, $w_1$, $w_2$, $\gamma$ and the matrix $A$. Also some auxiliar tables are initialized. This is done by routines CONSTA and INICI.

Then the Hamiltonian $M$ is computed by routine GENHAM. This requires in turn the routines SUMH, ADDH, and PRODH performing elementary operations with polynomials. In these routines the homogeneous polynomials are stored in reverse lexicographycal order. The routine GENLEX gives, sequentially, the reverse lexicographycal order and the funtion LLOC says where to store terms with a given set of exponents.

The next step is to compute the P.N.F. This is done by routine FNWCH which requires the additional routine POISH1, doing the Poisson bracket of homogeneous polynomials.

When $G$ and $\widetilde{M}$ are available to order $n$ one can do a check. We should have $\varphi^G_{t=1}M = \widetilde{M}$ up to order $n$. This is done by routine CHECK. If the changes of variables ( $(q,p)$ to $(Q,P)$ and viceversa) are required, this can be done by routine CHANGE.

Finally the routines REDHAM and AREAL perform the reduction to the central manifold and the passage to real variables.

The Table 2.5 of [4] shows the coefficients of the final reduced Hamiltonian up to order 8. The final expression is quite moderate. Only 130 terms appear. To order 16 the final number of terms is 1240, but one should be aware that in the initial Hamiltonian, $M$, there are 38034 complex coefficients and even in the P.N.F., $\widetilde{M}$, 5834 terms remain. The computing time including checks and the changes of variables takes $2^m$ at order 8, $1^h$ at order 12 and $16^h$ at order 16 (running on HP9000/835S). If we are only interesed in the reduced Hamiltonian only 1/5 of that time is needed. We also remark that if the computations are done to order $n$, the changes are available at order $n-1$.

### 2.7.3 Numerical Simulations and Tests

When $H^0(\xi,\eta)$ is available we can do several simulations. Usually order 8 has been used for $H^0$. Even so to compute several thousands of iterates of the Poincaré map through $\xi_2 = 0$ (equivalent to $z = 0$ in the initial variables) is a big task. This has been done for a limited set of values of the energy $h$ ($H^0 = h$). The Figures 2.4.1 to to 2.4.7 show the plots of that surface of section for the energies $h = 0.1$, 0.3, 0.4, 0.5, 0.7, 1.0 and 1.5 respectively. The boundary is always the Lyapunov periodic orbit. Figure 2.4.8 shows the difference between the Lyapunov orbit computed on $h = 1.5$ for order 8 (continuous curve) and order 16 (dotted curve). The half widths of the Figures (both in $\xi_1$ and $\eta_1$) are 0.36, 0.60, 0.68, 0.76, 0.92, 1.12, 1.44 and 1.44 respectively. In all those Fifures we can consider the Earth sitting on the negative part of the $\eta_1$ axis and the Sun, far away, on the positive part of the same axis. For $h < h_H \simeq 0.332820$, the Lyapunov orbit is stable. From this value on the halo orbits can be seen on the Figures. It is remarkable that even for $h = 1.5$ it is hard to see stochastic zones. One can see some very small islands but the motion is quite ordered. The lines (not drawn) which separate the zone of tori (or Lissajous orbits) around the "vertical orbit" (sitting near the center of the Figure) and the zones of tori around the halo orbits are the traces, on this surface of section, of the invariant manifolds of the Lyapunov orbit.

To relate the used levels of energy with the $\beta$ parameter of the halo orbits one can see the next Table.

| | |
|---|---|
| 0.02 | 0.334378 |
| 0.04 | 0.339052 |
| 0.06 | 0.346846 |
| 0.08 | 0.357762 |
| 0.10 | 0.371805 |
| 0.12 | 0.388981 |
| 0.14 | 0.409300 |
| 0.16 | 0.432770 |
| 0.18 | 0.459404 |
| 0.20 | 0.489217 |

Table 2.2

Also to present some idea of the goodness of the results from a quantitative point of view we present additional results on Table 2.3. The orbits 1 to 9 have been selected by groups of 3 in each one of the levels $h = 0.5$, $h = 1.0$, $h = 1.5$. Inside each level we give the initial conditions for periodic orbits of the following types: Lyapunov (orbits 1, 4, 7), vertical (2, 5, 8) and halo (3, 6, 9). All the computations have been done at order 8. Then we transform back to cartesian coordinates and we compare with numerical values. The maximal errors in the space variables are, in Km, 16, 12, 700, 313, 238, 3180, 1875, 1465 and 9705 for orbits 1 to 9, respectively. This is improved by a factor larger than 100 for $h \leq 0.5$ and near 10 for $h \simeq 1.5$ if order 16 is used. We should pay a greater computing time and 10 Mb of memory to store the coefficients of the changes.

| Orbit | Reduced | Cartesian | Numerical |
|---|---|---|---|
| 1 | 0.0 | −0.9886192288 | −0.9886191198 |
| | 0.0 | 0.0 | 0.0 |
| | 0.686159 | 0.0 | 0.0 |
| | 0.0 | 0.0 | 0.0 |
| | | −0.0107659711 | −0.0107660492 |
| | | 0.0 | 0.0 |
| 2 | 0.0 | −0.9903243964 | −0.9903243149 |
| | 0.0 | 0.0 | 0.0 |
| | 0.088866 | 0.0 | 0.0 |
| | 0.698586 | 0.0 | 0.0 |
| | | −0.0007133605 | −0.0007138474 |
| | | 0.0100388371 | 0.0100387530 |
| 3 | 0.633035 | −0.9911031366 | −0.9911074139 |
| | 0.0 | −0.0048042722 | −0.0047996050 |
| | −0.015524 | 0.0 | 0.0 |
| | 0.292145 | −0.0029410293 | −0.0029333451 |
| | | 0.0017601018 | 0.0017917750 |
| | | 0.0042457275 | 0.0042520908 |
| 4 | 0.0 | −0.9881865512 | −0.988184666 |
| | 0.0 | 0.0 | 0.0 |
| | 0.968531 | 0.0 | 0.0 |
| | 0.0 | 0.0 | 0.0 |
| | | −0.0150668946 | −0.0150683874 |
| | | 0.0 | 0.0 |
| 5 | 0.0 | −0.9906603953 | −0.9906588061 |
| | 0.0 | 0.0 | 0.0 |
| | 0.179863 | 0.0 | 0.0 |
| | 0.979163 | 0.0 | 0.0 |
| | | −0.0015091839 | −0.0015181117 |
| | | 0.01423098598 | 0.0142287520 |
| 6 | 0.806790 | −0.9919717435 | −0.9919730051 |
| | 0.0 | −0.0059292421 | −0.0059080613 |
| | 0.017889 | 0.0 | 0.0 |
| | 0.572802 | −0.0037342856 | −0.0037259701 |
| | | 0.0026423380 | 0.0026711102 |
| | | 0.0085015587 | 0.0085389294 |

| Orbit | Reduced | Cartesian | Numerical |
|---|---|---|---|
| 7 | 0.0 | −0.9878719651 | −0.9878594540 |
| | 0.0 | 0.0 | 0.0 |
| | 1.184712 | 0.0 | 0.0 |
| | 0.0 | 0.0 | 0.0 |
| | | −0.0183518934 | −0.0183610059 |
| | | 0.0 | 0.0 |
| 8 | 0.0 | −0.99909942212 | −0.9909844530 |
| | 0.0 | 0.0 | 0.0 |
| | 0.271440 | 0.0 | 0.0 |
| | 1.188193 | 0.0 | 0.0 |
| | | −0.0023749889 | −0.0024266444 |
| | | 0.0174669454 | 0.0174506786 |
| 9 | 0.963854 | −0.9929400824 | −0.9929183465 |
| | 0.0 | −0.0068042692 | −0.0067395456 |
| | 0.050572 | 0.0 | 0.0 |
| | 0.739605 | −0.0043961422 | −0.0044301972 |
| | | 0.0037697329 | 0.0037048142 |
| | | 0.0112519581 | 0.0113758971 |

Table 2.3

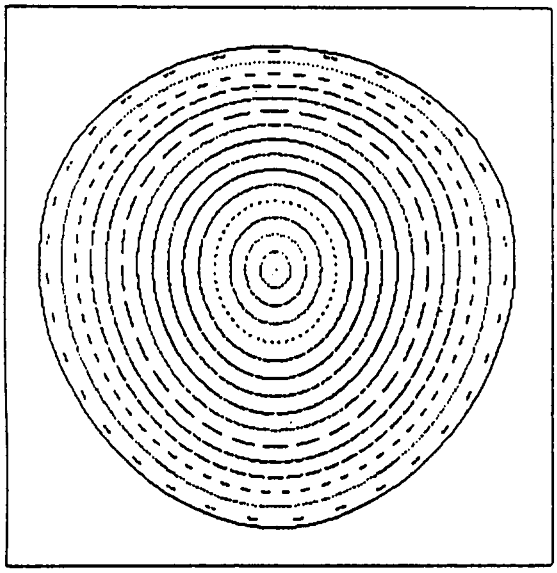


Figure 2.4.1

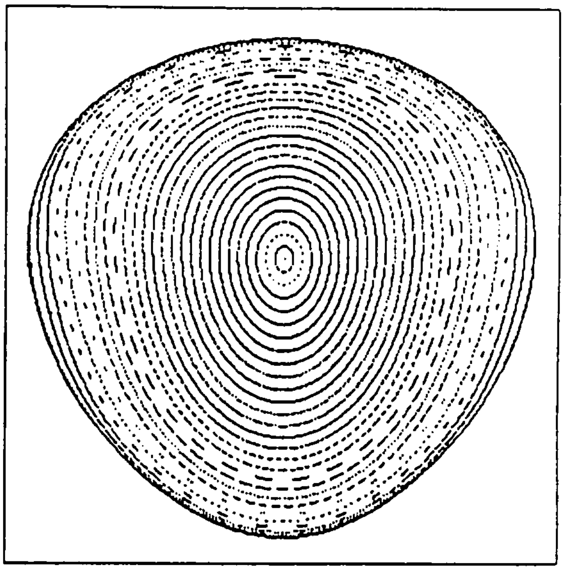


Figure 2.4.2

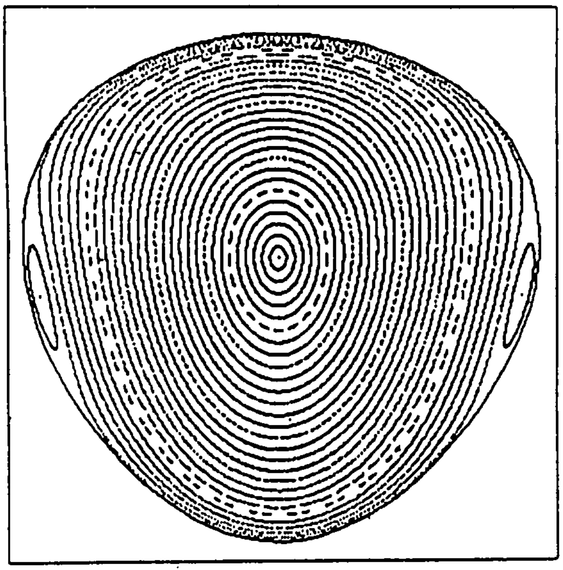


Figure 2.4.3

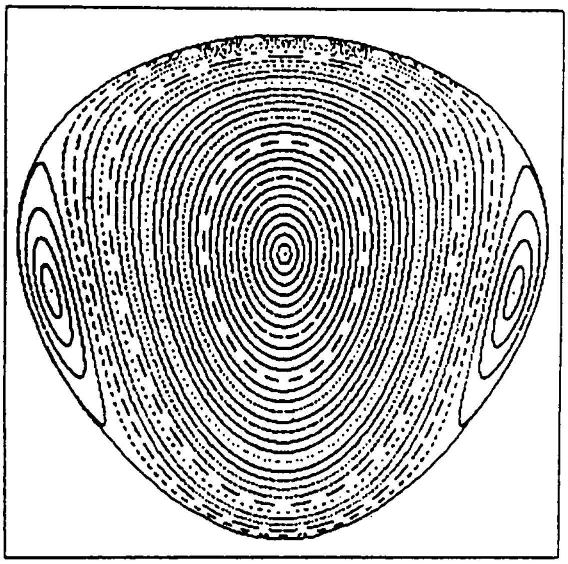


Figure 2.4.4

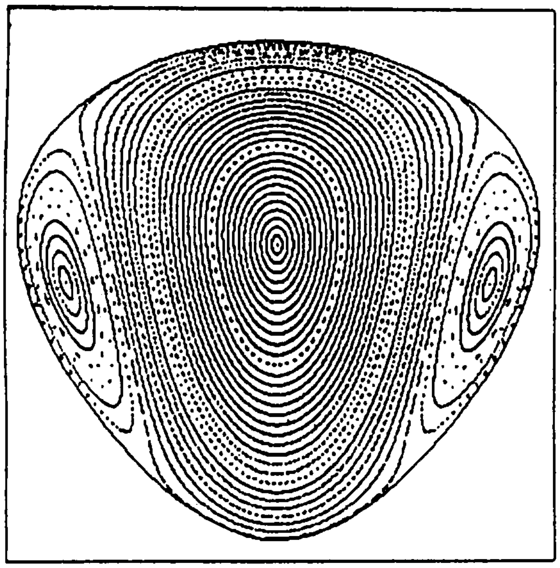


Figure 2.4.5

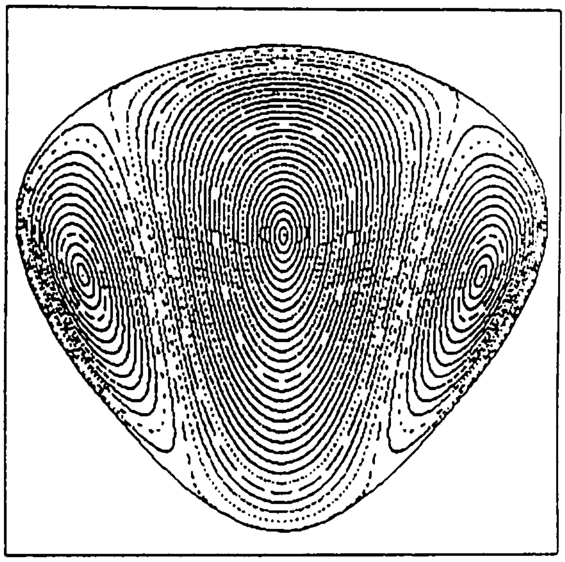


Figure 2.4.6

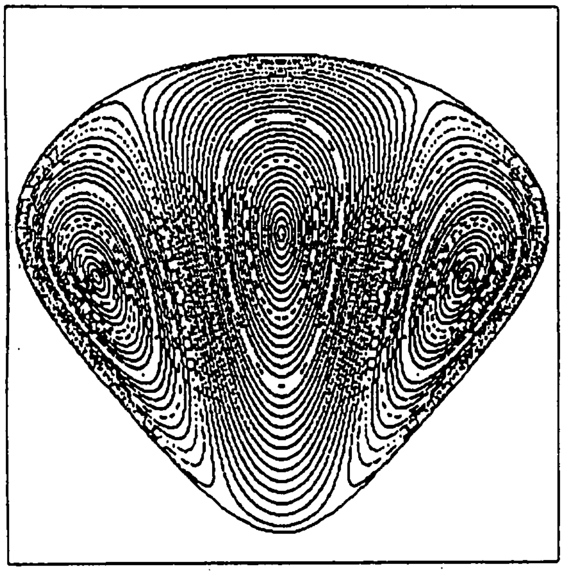


Figure 2.4.7

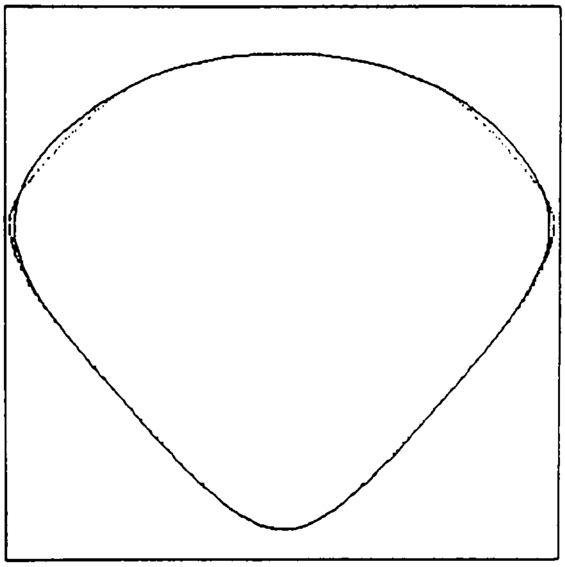


Figure 2.4.8

## 2.8 Conclusions

So far we have seen that it is possible to describe quite precisely by analytical means the neighbourhood of $L_1$ for the Earth-Moon problem as a RTBP skipping the unstable/stable directions up to distances less than $0.9 \times 10^6$ Km from the Earth. Indeed the Lyapunov orbit for $h$ =1.5 ranges from less than $0.9 \times 10^6$ to more than $1.85 \times 10^6$ Km from the Earth and its range in the $y$ variable attains $2.9 \times 10^6$ Km. The use of the central manifold to high order is much better than the methods used before to obtain Lissajous orbits. It is not affected by small divisors problems. It has the problem that it does not produce explicit solutions. The description of the neighbourhood could not be done without skipping the instability.

To go back to the real problem one should refer again to parallel shooting including the perturbations starting with a numerical solution of $M^0$. Eventually continuation should be used for large amplitudes.

# Chapter 3

# Quasiperiodic Halo Orbits

## 3.1 Numerical Refinement

Taking as an starting point the analytical quasiperiodic halo orbit obtained by program QPO, in the way described in Chapter VII of [2], a quasiperiodic orbit of the real solar system is computed, using the JPL ephemeris, by means of a parallel shooting procedure.

In the normalized reference system, see Appendix B of [4] or Chapter V of [2], we say that a qpo has z-amplitude $\beta$ if it comes from improving a halo orbit of z-amplitude $\beta$.

If $(t, x, y, z, \dot{x}, \dot{y}, \dot{z})$ denote the time, position and velocity in the normalized system, we define the Poincaré map, $\mathcal{P}$, with the surface of section $y = 0$, following the flow when $t$ increases:

$$\begin{aligned}\mathcal{P}(t, x, y = 0, z, \dot{x}, \dot{y}, \dot{z}) &= (\overline{t}, \overline{x}, \overline{z}, \dot{\overline{x}}, \dot{\overline{y}}, \dot{\overline{z}}) = \\ &= (\mathcal{P}_1(t, x, z, \dot{x}, \dot{y}, \dot{z}), \ldots, \mathcal{P}_6(t, x, z, \dot{x}, \dot{y}, \dot{z})).\end{aligned}$$

The surface of section $y = 0$ gives the intervals at which the parallel shooting is defined, in the sense that each interval of the procedure is comprised between two crosses with this surface of section.

We denote by $Q_i = (t_i, x_i, z_i, \dot{x}_i, \dot{y}_i, \dot{z}_i)$ the $i-$th point in the partition of the full interval at which the qpo shall be computed. If the parallel shooting is splitted in $N$ subintervals then $i = 0, \ldots, N$.

So, in principle, we have $6(N + 1)$ free variables. One should take into account that we choose the initial epoch at which the qpo must be computed, and, in addition, we improve a qpo with a certain z-amplitude. So we fix $t_0$ and $z_0$ at the beginning of the procedure and the number of free variables (the components of $Q$) is reduced to $6N + 4$.

In this way we need $6N + 4$ equations for the determination of these variables.

Any parallel shooting must satisfy the matching conditions:

$$F_{6i+j}(Q) = \mathcal{P}_j(Q_i) - (Q_{i+1})_j = 0, \quad j = 1, \dots, 6, \quad i = 0, \dots, N-1,$$

where $F$ is the vector of $6N+4$ equations to be fulfilled by the variable $Q$.

The matching conditions give us $6N$ equations. The remaining four ones can be imposed in several ways obtaining different approaches of the qpo.

The ones that we have taken are:

$$\begin{array}{lll}
F_{6N+1} & = x_0 - {x^t}_0 - (x_N - {x^t}_N) & = 0, \\
F_{6N+2} & = z_N - {z^t}_N & = 0, \\
F_{6N+3} & = \pi_4(\dot{x}_0 - \dot{x}_0^t) + \pi_5(\dot{y}_0 - \dot{y}_0^t) + \pi_6(\dot{z}_0 - \dot{z}_0^t) & = 0, \\
F_{6N+4} & = \pi_4(\dot{x}_N - \dot{x}_N^t) + \pi_5(\dot{y}_N - \dot{y}_N^t) + \pi_6(\dot{z}_N - \dot{z}_N^t) & = 0,
\end{array}$$

Here the superindex $t$ denotes a value taken from the analytical quasiperiodic orbit and $\pi_4$, $\pi_5$, $\pi_6$ are the three last components of the projection factors in the unstable component of the quasiperiodic orbit (see [2]).

As a summary, the system of equations to be solved is:

$$\begin{array}{rll}
F_1(Q) & = \mathcal{P}_1(t_0, x_0, z_0, \dot{x}_0, \dot{y}_0, \dot{z}_0) - t_1 & = 0, \\
F_2(Q) & = \mathcal{P}_2(t_0, x_0, z_0, \dot{x}_0, \dot{y}_0, \dot{z}_0) - x_1 & = 0, \\
\dots & \dots\dots & \\
F_6(Q) & = \mathcal{P}_6(t_0, x_0, z_0, \dot{x}_0, \dot{y}_0, \dot{z}_0) - \dot{z}_1 & = 0, \\
F_7(Q) & = \mathcal{P}_1(t_1, x_1, z_1, \dot{x}_1, \dot{y}_1, \dot{z}_1) - t_2 & = 0, \\
\dots & \dots\dots & \\
F_{6N}(Q) & = \mathcal{P}_6(t_{N-1}, x_{N-1}, z_{N-1}, \dot{x}_{N-1}, \dot{y}_{N-1}, \dot{z}_{N-1}) - \dot{z}_N & = 0, \\
F_{6N+1}(Q) & = x_0 - {x^t}_0 - (x_N - {x^t}_N) & = 0, \\
F_{6N+2}(Q) & = z_N - {z^t}_N & = 0, \\
F_{6N+3}(Q) & = \pi_4(\dot{x}_0 - \dot{x}_0^t) + \pi_5(\dot{y}_0 - \dot{y}_0^t) + \pi_6(\dot{z}_0 - \dot{z}_0^t) & = 0, \\
F_{6N+4}(Q) & = \pi_4(\dot{x}_N - \dot{x}_N^t) + \pi_5(\dot{y}_N - \dot{y}_N^t) + \pi_6(\dot{z}_N - \dot{z}_N^t) & = 0,
\end{array}$$

where

$$Q = (t_0, x_0, \dot{x}_0, \dot{y}_0, \dot{z}_0, t_1, x_1, z_1, \dot{x}_1, \dot{y}_1, \dot{z}_1, \cdots, t_N, x_N, z_N, \dot{x}_N, \dot{y}_N, \dot{z}_N).$$

Denoting the equations by $F(Q) = 0$ and taking $Q^{(0)}$ as the initial values of the variables, the improved values are obtained by means of a Newton procedure:

$$DF(Q^{(j)})(Q^{(j+1)} - Q^{(j)}) = -F(Q^{(j)}).$$

### 3.1.1 Computation of the Initial Values

The initial values of the components of $Q^{(0)}$, $t_0$ and $z_0$ can be obtained as follows. For a halo orbit it is clear that, belonging to a one parameter family, it is enough to fix a value of $z$ when $y = 0$ at a given epoch. From the characteristic curve of the family, analytically expressed by $\Delta(\alpha, \beta) = 0$, the x-amplitude, $\alpha$, can be obtained from the z-amplitude, $\beta$. It is true that $\beta$ does not coincide with $z_0$, but the expression of $z$ when the halo angle is zero: $z = \beta + O_2(\alpha, \beta)$, where $O_2$ means terms of second order in $\alpha, \beta$, shows that $z$ and $\beta$ are close. An iterative procedure can be used to determine $\beta$.

The terms added to the halo part of the orbit by program QPO (see Chapter VII of [2]) depend on $\alpha$, $\beta$ and $t$.

Hence, for a given $t_0$, the values of $x_0$, $y_0$, $z_0$, $\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$ depend on $\alpha$ and $\beta$, and $\alpha$ and $\beta$ should satisfy the $\Delta$ condition. They also depend on the epoch, $t^*$, at which the halo part of the qpo crosses the $y = 0$ hyperplane with $z$ having the sign assigned to $\beta$. If the quasiperiodic perturbations where zero, the time $t^*$ will coincide with $t_0$ (modulus the period of the halo orbit) if the sign of $\beta$ and $z_0$ agree. Othewise $t_0$ and $t^*$ differ in a half period (also modulus the period).

When the quasiperiodic part of the orbit is included $t_0$ and $t^*$ are close but they are not forcely equal (or they differ roughly in a half period of the halo part).

First of all we have used an iterative procedure to determine $\beta$ and $t^*$ from $z_0$ and $t_0$. This is implemented in routine ESTQPO. We proceed as follows:

- We start with an initial value of the parameters $\beta$ and $t^*$. If nothing better is available we use $\beta = \pm z_0$, $t^* = t_0 + \delta \cdot T/2$ where the sign of $\beta$ is chosen in order to agree with the previously assigned sign. In the expression of $t^*$ the parameter $\delta$ is set equal to zero if $\beta$ and $z_0$ have the same sign, and $\delta = 1$ otherwise. $T$ refers to the aproximate period of the halo orbit. As $\alpha$ and $\beta$ are unknown, this period is not available, however taking into account that we deal with small and medium size halo orbits and its period has not large changes along the family, the approximate value of $T$ is the one which corresponds to $\beta = 0.08$.

- When an approximation of $(\beta, t^*)$ is available, we compute (using routine ANAQPO) the corresponding values of $y$ and $z$ (and in fact all the coordinates). The standing equations are $y(\beta, t^*, t_0) = 0$ and $z(\beta, t^*, t_0) = z_0$.

- The previous equations are solved by Newton's method. For this purpose we need the partial derivatives: $\frac{\partial y}{\partial \beta}$, $\frac{\partial y}{\partial t^*}$, $\frac{\partial z}{\partial \beta}$ and $\frac{\partial z}{\partial t^*}$. They are computed using numerical differentiation with an step $\varepsilon$, selected equal to $10^{-6}$. Using this partial derivatives the Newton procedure is iterated till convergence. The value $10^{-11}$ in the corrections of $\beta$ and $t^*$ have been used as stopping criteria.

The full initial conditions, $(x_0, y_0, z_0, \dot{x}_0, \dot{y}_0, \dot{z}_0)$ at $t = t_0$, are known once the values of $\beta$ and $t^*$ have been determined. Now we need to determine, for the analytical qpo, the epochs $t_1, \ldots, t_N$ of the successive passages through $y = 0$. The separation $t_{i+1} - t_i$ is a "half revolution", so when $t_i$ is known, we determine an approximate value of $t_{i+1}$ adding $T/2$ to $t_i$. Then the routine PREDYO determines the corrections to be applied to $t_{i+1}$ in order to get $y = 0$. This is done using again Newton's method. In this case the required derivative, $\dot{y}$, is available from routine ANAQPO. By iteration (up to an error less than $10^{-10}$) of $\Delta t_{i+1} = -y(t_{i+1})/\dot{y}(t_{i+1})$ we obtain the value $t_{i+1}$.

### 3.1.2 The Differential Matrix $DF(Q)$

The differential matrix $DF(Q)$ has the following structure:

$$DF(Q) = \begin{pmatrix} A_0 & -I & & & & & \\ & A_1 & -I & & & & \\ & & A_2 & -I & & & \\ & & & \cdots & \cdots & & \\ & & & & \cdots & \cdots & \\ & & & & & A_{N-1} & -I \\ B_N & & & & & & A_N \end{pmatrix},$$

where $A_i$, $i = 1, 2, \ldots, N$ denote the $6 \times 6$ matrix $D\mathcal{P}(Q_i)$, and $A_0$ is the $6 \times 4$ matrix obtained from $D\mathcal{P}(Q_0)$ skipping the columns 1 and 3 (related to $t_0$ and $z_0$ that do not change).

The matrix $B_N$ is $4 \times 4$, with all entries equal zero except the element in the first row and column which is set equal to 1.

The matrix $A_N$ is a $4 \times 6$ matrix with the following structure:

$$A_N = \begin{pmatrix} a_1 & -1 & a_5 & 0 & 0 & 0 \\ a_2 & 0 & a_6 & -1 & 0 & 0 \\ a_3 & 0 & a_7 & 0 & -1 & 0 \\ a_4 & 0 & a_8 & 0 & 0 & -1 \end{pmatrix},$$

where $a_1 = \frac{\partial x_N^t}{\partial t_N}$, $a_2 = \frac{\partial \dot{x}_N^t}{\partial t_N}$, $a_3 = \frac{\partial \dot{y}_N^t}{\partial t_N}$, $a_4 = \frac{\partial \dot{z}_N^t}{\partial t_N}$, $a_5 = \frac{\partial x_N^t}{\partial z_N}$, $a_6 = \frac{\partial \dot{x}_N^t}{\partial z_N}$, $a_7 = \frac{\partial \dot{y}_N^t}{\partial z_N}$ and $a_8 = \frac{\partial \dot{z}_N^t}{\partial z_N}$.

The derivatives $a_1, \ldots, a_8$ have been computed using numerical differentiation.

### 3.1.3 Computing the Poincaré Map and its Differential

Let $\vec{r}, \vec{r}_1, \ldots, \vec{r}_k$ the ecliptic coordinates (centered at the center of masses of the solar system) of the spacecraft and of the bodies number 1 to k of the solar system. The lack of coherence and radiation pressure are included in the masses of the corresponding bodies.

Then the equations of motion are:

$$\ddot{\vec{r}} = -\sum_{i=1}^{k} Gm_i \frac{\vec{r}-\vec{r_i}}{d_i^3},$$

where $m_i$ is the (total) mass of the i-th body, $G$ is the gravitational constant and $d_i = \mid \vec{r}-\vec{r_i} \mid_2$.

The Poincaré map is expressed in normalized coordinates. Hence, given initial conditions $t$, $x$, $y=0$, $z$, $\dot{x}$, $\dot{y}$, $\dot{z}$, they are converted to days and ecliptic position and velocity. Then the numerical integration of Newton's equations is started. First we perform the integration for a time interval which is near to, but a little bit less than, the half period or the full period of the halo part of the analytical qpo, depending of the type of intervals that we have chosen. Then, after each step in the numerical integration, the point is transformed back to normalized time and coordinates and the condition $y = 0$ is searched. When, at some step, a change of sign of $y$ is detected, a Newton's method is used through $\Delta t = -y/\dot{y}$ to obtain the final time, $\mathcal{P}_1(t, x, z, \dot{x}, \dot{y}, \dot{z})$, under the Poincaré map. The remaining variables, $x, z, \dot{x}, \dot{y}, \dot{z}$ give the other five components of the image of the Poincaré map.

To obtain the differential, $D\mathcal{P}$, of the Poincaré map, the equations of motion should be integrated simultaneously with the variational equations. As the system is non autonomous we have also variations with respect to the initial time. First we look in ecliptic coordinates. Let $x_1$, $x_2$, $x_3$, $x_4$, $x_5$, $x_6$ denote ecliptic position and velocity.

The equations of the autonomized system are:

$$\begin{aligned}
\dot{x}_0 &= 1, \\
\dot{x}_1 &= x_4, \\
\dot{x}_2 &= x_5, \\
\dot{x}_3 &= x_6, \\
\dot{x}_4 &= -\sum_{i=1}^{k} Gm_i \frac{x_1 - x_{i,1}}{d_i^3}, \\
\dot{x}_5 &= -\sum_{i=1}^{k} Gm_i \frac{x_2 - x_{i,2}}{d_i^3}, \\
\dot{x}_6 &= -\sum_{i=1}^{k} Gm_i \frac{x_3 - x_{i,3}}{d_i^3}.
\end{aligned}$$

The variational equations can be written in the form $\dot{V} = D \cdot V$, where $V$ and $D$ are $7 \times 7$ matrices and $V$ is initialized to the identity matrix. The matrix $D$ has the block structure:

$$D = \left(\begin{array}{c|ccc|ccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & & & & 1 & 0 & 0 \\ 0 & & \bigcirc & & 0 & 1 & 0 \\ 0 & & & & 0 & 0 & 1 \\ \hline g_1 & h_{11} & h_{12} & h_{13} & 0 & 0 & 0 \\ g_2 & h_{21} & h_{22} & h_{23} & 0 & 0 & 0 \\ g_3 & h_{31} & h_{32} & h_{33} & 0 & 0 & 0 \end{array}\right),$$

where:

$$\begin{aligned}
g_1 &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(-x_{i,1} + \frac{3q_i\,(x_1 - x_{i,1})}{d_i^2}\right), \\
g_2 &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(-x_{i,2} + \frac{3q_i\,(x_2 - x_{i,2})}{d_i^2}\right), \\
g_3 &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(-x_{i,3} + \frac{3q_i\,(x_3 - x_{i,3})}{d_i^2}\right), \\
q_i &= (x_1 - x_{i,1})\dot{x}_{i,1} + (x_1 - x_{i,2})\dot{x}_{i,2} + (x_1 - x_{i,3})\dot{x}_{i,3}, \\
h_{11} &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(1 - \frac{3\,(x_1 - x_{i,1})^2}{d_i^2}\right), \\
h_{12} &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(-\frac{3\,(x_1 - x_{i,1})\,(x_2 - x_{i,2})}{d_i^2}\right), \\
h_{13} &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(-\frac{3\,(x_1 - x_{i,1})\,(x_3 - x_{i,3})}{d_i^2}\right), \\
h_{21} &= h_{12}, \\
h_{22} &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(1 - \frac{3\,(x_2 - x_{i,2})^2}{d_i^2}\right), \\
h_{23} &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(-\frac{3\,(x_2 - x_{i,2})\,(x_3 - x_{i,3})}{d_i^2}\right), \\
h_{31} &= h_{13}, \\
h_{32} &= h_{23}, \\
h_{33} &= -\sum_{i=1}^{k} \frac{Gm_i}{d_i^3}\left(1 - \frac{3\,(x_3 - x_{i,3})^2}{d_i^2}\right).
\end{aligned}$$

In principle the variational equations would be 49 but, due to the fact that the first row of $D$ is zero, the full first row of the variational variables must remain constant during the integration. We can skip seven equations plus the one: $\dot{t} = \dot{x}_0 = 1$. So the variational equations can be written as:

$$\begin{pmatrix} x_7 \\ x_8 \\ \cdots \\ \cdots \\ x_{12} \end{pmatrix}^{\bullet} = \tilde{D} \begin{pmatrix} x_7 \\ x_8 \\ \cdots \\ \cdots \\ x_{12} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ g_1 \\ g_2 \\ g_3 \end{pmatrix},$$

$$\begin{pmatrix} x_{13} & x_{19} & \cdots & \cdots & x_{43} \\ x_{14} & x_{20} & \cdots & \cdots & x_{44} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{18} & x_{24} & \cdots & \cdots & x_{48} \end{pmatrix}^{\bullet} = \tilde{D} \begin{pmatrix} x_{13} & x_{19} & \cdots & \cdots & x_{43} \\ x_{14} & x_{20} & \cdots & \cdots & x_{44} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{18} & x_{24} & \cdots & \cdots & x_{48} \end{pmatrix},$$

where:

$$\tilde{D} = \left( \begin{array}{ccc|ccc} & & & 1 & 0 & 0 \\ & \bigcirc & & 0 & 1 & 0 \\ & & & 0 & 0 & 1 \\ \hline h_{11} & h_{12} & h_{13} & 0 & 0 & 0 \\ h_{21} & h_{22} & h_{23} & 0 & 0 & 0 \\ h_{31} & h_{32} & h_{33} & 0 & 0 & 0 \end{array} \right).$$

The initial conditions are $x_j = 0$ for $j = 1$ to 48 except $x_{13} = x_{20} = \ldots = x_{48} = 1$. When the integration of the variational matrix, $V$, is finished one should do two things:

1. To obtain the differential matrix of the Poincaré map, $\mathcal{P}$, in normalized coordinates.

2. To obtain, also in normalized coordinates, the matrix of the partial derivatives of $x$, $y$, $z$, $\dot{x}$, $\dot{y}$ and $\dot{z}$ at the time $t_{i+1}$ with respect to the values of $x$, $y$, $z$, $\dot{x}$, $\dot{y}$, $\dot{z}$ at $t = t_i$. In the case of a halo orbit this matrix would be the monodromy matrix.

The second point is not used by the parallel shooting algorithm, but will be used in the course of the determination of the numerical projection factors.

First of all we require to express the 7 × 7 matrix $V$ in normalized coordinates, $V_n$. We have $V_n = DT_2 \cdot V \cdot DT_1$, where $DT_1$ is the differential of the transformation, $T_1$, from normalized variables to ecliptic ones,

$$T_1(t, x, y, z, \dot{x}, \dot{y}, \dot{z}) = (t_d, x_1, x_2, x_3, x_4, x_5, x_6),$$

evaluated at the initial point, and $DT_2$ is the differential of the inverse transformation, $T_2$, at the final time,

$$T_2(\bar{t}_d, \bar{x}_1, \bar{x}_2, \bar{x}_3, \bar{x}_4, \bar{x}_5, \bar{x}_6) = (\bar{t}, \bar{x}, \bar{y}, \bar{z}, \dot{\bar{x}}, \dot{\bar{y}}, \dot{\bar{z}}).$$

The overlined variables mean the image variables, and $t_d$ refers to the time expressed in julian days instead of adimensional (normalized) time units.

Using the routines TRANSLI, to go from normalized to adimensional anu TRANS, to go from adimensional to ecliptic, the transformation $T_1$ is obtained. $T_2$ is given by the inverse.

The differentials $DT_1$, $DT_2$ have been evaluated by numerical differentiation. The step used in the variables is a magnitude given by the user. We have used the value $10^{-6}$ as more convenient.

When $V_n$ is available, the matrix refered to in 2) is obtained easily deleting the first row and column. The matrix of 1) requires a little more work. We put

$$V_n \begin{pmatrix} \Delta t \\ \Delta x \\ 0 \\ \Delta z \\ \Delta \dot{x} \\ \Delta \dot{y} \\ \Delta \dot{z} \end{pmatrix} + \delta t \begin{pmatrix} 1 \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix}_f = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \end{pmatrix}.$$

The amount $\delta t$ is left free in order to have $c_2 = 0$. The components of the vectorfield at the last point, in normalized coordinates, $(\dot{x}, \dot{y}, \dot{z}, \ddot{x}, \ddot{y}, \ddot{z})_f$, are obtained again from routines TRANS and TRANSLI.

If the element $(i, j)$ of the matrix $V_n$ is denoted by $V_{nij}$, $i, j = 0, \ldots, 6$, $\delta t$ is determined through:

$$V_{n20}\Delta t + V_{n21}\Delta x + V_{n23}\Delta z + V_{n24}\Delta \dot{x} + V_{n25}\Delta \dot{y} + V_{n26}\Delta \dot{z} + \delta t \cdot \dot{y} = 0.$$

If this value of $\delta t$ is substituted in the expressions of $c_0, c_1, \ldots, c_6$, we get:

$$\begin{aligned}
c_0 &= V_{n00}\Delta t + V_{n01}\Delta x + V_{n03}\Delta z + V_{n04}\Delta \dot{x} + V_{n05}\Delta \dot{y} + V_{n06}\Delta \dot{z} - \\
&\quad - \left(\frac{1}{\dot{y}}\right)_f (V_{n20}\Delta t + V_{n21}\Delta x + V_{n23}\Delta z + V_{n24}\Delta \dot{x} + V_{n25}\Delta \dot{y} + \\
&\quad + V_{n26}\Delta \dot{z}) \\
&= \left(V_{n00} - \frac{V_{n20}}{\dot{y}}\right)_f \Delta t + \ldots + \left(V_{n06} - \frac{V_{n26}}{\dot{y}}\right)_f \Delta \dot{z}, \\
\ldots &\quad \ldots \\
c_6 &= V_{n60}\Delta t + V_{n61}\Delta x + V_{n63}\Delta z + V_{n64}\Delta \dot{x} + V_{n65}\Delta \dot{y} + V_{n66}\Delta \dot{z} - \\
&\quad - \left(\frac{\ddot{z}}{\dot{y}}\right)_f (V_{n20}\Delta t + V_{n21}\Delta x + V_{n23}\Delta z + V_{n24}\Delta \dot{x} + V_{n25}\Delta \dot{y} +
\end{aligned}$$

$$
\begin{aligned}
& +V_{n26}\Delta\dot{z}) \\
= & \left(V_{n60} - \frac{\ddot{z}V_{n20}}{\dot{y}}\right)_f \Delta t + \ldots + \left(V_{n66} - \frac{\ddot{z}V_{n26}}{\dot{y}}\right)_f \Delta\dot{z}.
\end{aligned}
$$

The components of $c_0, c_1, \ldots, c_6$ (seen as linear forms) give the matrix $D\mathcal{P}$:

$$
\begin{aligned}
(D\mathcal{P})_{11} &= \left(V_{n00} - \frac{V_{n20}}{\dot{y}}\right)_f, \\
\ldots & \quad \ldots \\
(D\mathcal{P})_{66} &= \left(V_{n66} - \frac{\ddot{z}V_{n26}}{\dot{y}}\right)_f.
\end{aligned}
$$

The computation of the image under the Poincaré map and its differential is done in the routine POINC2. The routine DERIVP contains the vectorfield in dimension 48 obtained by adding to the equations of motion the variational equations (we remark that the equation $\dot{x}_0 = 1$ and the related variational equations, $\dot{V}_{0i} = 0$, $i = 0, \ldots, 6$ can be skipped).

With the initial assignations of values to $Q$ and the iterative procedure, the parallel shooting program GQPTPS finds a final value of $Q$.

## 3.2 Main Program and Basic Routines

The program which performs the computations of the quasiperiodic halo orbits makes use of a large set of auxiliar routines, numerical integrator, manipulation of formal series, changes of coordinates, routines for the use of the JPL ephemeris, etc. which are not explained here (see Appendix A of [4] for its description). Only the ones that are closely related to the problem are described below.

**Program GQPTPS** This program produces the initial conditions, at a selected number of points, of the motion of the spacecraft in the solar system near a quasiperiodic halo orbit. The starting data are obtained from the output of programs ANACOM, QPO and CONA. (See [2] for these last three programs).

**Subroutine ANAQPO** Computes the coordinates (position, velocity and acceleration) of a quasiperiodic halo orbit according to the analytical theory using the output of program QPO.

**Subroutine DERIVP** This routine computes the ecliptic vectorfield defining the equations of motion and the variational equations, with respect to time and to coordinates in the phase space.

**Subroutine ESTQPO** When the quasiperiodic part of the halo orbit is included, this routine performs the iterative procedure to compute $\beta$ and $t^*$ from $z_0$ and $t_0$.

**Subroutine MAGCON** This routine computes analytical approximations, by using the RTBP model, of the normalized projection factors on the unstable direction which appear in the last three equations that define the parallel shooting procedure. For its computation it uses the output of program `CONA`.

**Subroutine POINC2** This routine computes explicitely the Poincaré map of the flow with the surface of section $y = 0$. The differential of the Poincaré map can also computed making use of numerical differentiation.

**Subroutine PREDY0** When the values of $\beta$ and $t^*$ have been obtained, this routine computes the epochs $t_1, \ldots, t_N$ of the succesive passages of the quasiperiodic halo orbit through $y = 0$.

**Subroutine TPSS2** This is one of the main routines of the program. Making use of the Poincaré map (routine `POINC2`) it performs the computations of the parallel shooting algorithm.

### 3.2.1 Numerical Results

An extensive use of this program has been done in the study of the transfer problem. Here we present, as a sample of the typical output of the program, the results corresponding to a halo orbit of z-amplitude $\beta$=0.08.

The numerical quasiperiodic orbit has been refined starting from a periodic halo orbit with these characteristics:

- The spacecraft section divided by the mass (`SPCSM` = 0.01).
- The full solar system is considered (`IPA`).
- The orbit is computed in the Earth+Moon-Sun System (`ICO=2`).
- The z-amplitude (`ZO`) is 0.08.
- The initial epoch, in modified julian days starting at 1950.0, (`TO`) at which the integration is started is 16587.0 (1-VI-1995).
- The number of half revolutions (`NIT`) for the parallel shooting is 40.
- The parallel shooting was performed using half revolutions instead of full revolutions (`NHR=1`). It is possible to do the computations of the parallel shooting using full revolutions (`NHR=2`) but some care must be taken when using the output of this program to do the transfer computations.

- The parameter $\beta$ has been chosen with the same sign as $z_0$ (`IBETA=1`), it can be chosen with the opposite sign taking (`IBETA=-1`).
- The bound for the error in the equations for a stopping criteria in the parallel shooting procedure (`BOUND1`) is $2 \times 10^{-9}$.
- The bound for the error in the corrections for a stopping criteria in the parallel shooting procedure (`BOUND2`) is $3 \times 10^{-10}$.
- The selected number of maximum iterations of the procedure (`NUMITM`) is 12.
- The step for the numerical differentiation for computing the Poincaré map (`EPS`) is $10^{-6}$.

With these data program `GQPTPS` has needed 6 iterations having as final error in the equations of the parallel shooting the value $2.8927 \cdot 10^{-10}$.

Only a sample of the output is given here. It has been splitted in two parts. In the first one we give the normalized coordinates of the intersections of the numerical quasiperiodic orbit with the surface of section $y = 0$ (`VECTOR XS`). In the second part we give the epochs at which these intersections take place in different units: normalized, modified julian dates and calendar dates.

The output of the program gives also the monodromy matrix related to each interval of the parallel shooting. This part of the output is not included here because of its large extension.

Finally some plots of the projections of the qpo on the coordinate planes are included.

```
SPCSM = .10000000000E-01  IPA =  1  1 11  1  1  1  1  1  1  1  1  1  1  0  0  0  ICO = 2
Z0 =  .080000 T0 = 16587.00000 NIT = 40 NHR = 1 IBETA = 1
BOUND1 = .2000E-08  BOUND2 = .3000E-09  NUMITM = 12  EPS = .1000E-05

VECTOR XS (normalized coodinates)
 0  -.11600476054E+00   .00E+00   .80000000002E-01   .10826885389E-01   .86029830892E+00  -.69048017066E-02
 1   .15892844037E+00   .71E-11  -.65113375545E-01   .22149457655E-02  -.94301403284E+00   .69486923151E-02
 2  -.11488379204E+00  -.47E-11   .81450254119E-01  -.89861259915E-03   .90153843583E+00  -.52032939411E-02
 3   .16133690121E+00   .71E-11  -.65702194104E-01  -.10612061403E-02  -.98132463978E+00   .65523734005E-02
 4  -.11210263903E+00  -.47E-11   .81319829491E-01   .10077074844E-02   .85837949759E+00  -.51095675728E-02
 5   .15875780152E+00   .71E-11  -.66154478048E-01   .17944442095E-02  -.93902846371E+00   .57145011265E-02
 6  -.11456965483E+00  -.47E-11   .82684525140E-01  -.10718417334E-02   .89780324139E+00  -.40709123115E-02
 7   .16152773706E+00   .71E-11  -.66687324460E-01  -.10919297659E-02  -.98476483863E+00   .51099377425E-02
 8  -.11221793799E+00  -.47E-11   .82465369173E-01   .10921878703E-02   .86124584359E+00  -.39985627301E-02
 9   .15859763506E+00   .71E-11  -.66956720297E-01   .13206980203E-02  -.93574644182E+00   .42243943161E-02
10  -.11426249898E+00  -.47E-11   .83604828979E-01  -.11888138236E-02   .89369090144E+00  -.27700318156E-02
11   .16167939178E+00   .71E-11  -.67403115761E-01  -.96095979396E-03  -.98751665718E+00   .35238311800E-02
12  -.11240079010E+00  -.47E-11   .83282818787E-01   .11730051798E-02   .86490253561E+00  -.27747480684E-02
13   .15845038527E+00   .71E-11  -.67490295341E-01   .65950783055E-03  -.93286474582E+00   .26160143185E-02
14  -.11394377662E+00  -.47E-11   .84166673428E-01  -.13230253833E-02   .88915157632E+00  -.14127633162E-02
15   .16176834282E+00   .71E-11  -.67833447511E-01  -.54665877571E-03  -.98956041436E+00   .18924410125E-02
16  -.11264708626E+00  -.47E-11   .83750985460E-01   .12530383237E-02   .86910863613E+00  -.15019312234E-02
17   .15834003364E+00   .71E-11  -.67740900235E-01   .14091935384E-03  -.93074579702E+00   .98966776381E-03
18  -.11363714076E+00  -.47E-11   .84342373598E-01  -.14447564886E-02   .88471692457E+00  -.60403509874E-04
19   .16189117103E+00   .71E-11  -.67953551019E-01  -.23254589803E-03  -.99130021513E+00   .18763826273E-03
20  -.11294874492E+00  -.48E-11   .83848304224E-01   .13334295927E-02   .87373859269E+00  -.17858473250E-03
21   .15827977856E+00   .71E-11  -.67692637286E-01  -.20529937839E-03  -.92968695079E+00  -.60153781054E-03
22  -.11336254383E+00  -.47E-11   .84159845749E-01  -.14679523051E-02   .88030368191E+00   .12125124118E-02
23   .16199059998E+00   .71E-11  -.67809710392E-01  -.17326171870E-03  -.99221367929E+00  -.15422177274E-02
24  -.11325912147E+00  -.47E-11   .83628267463E-01   .13800825620E-02   .87850081443E+00   .11556109430E-02
25   .15829291749E+00   .71E-11  -.67392092348E-01  -.34437214383E-03  -.92969957462E+00  -.21126756444E-02
26  -.11312447640E+00  -.47E-11   .83657039284E-01  -.13946197659E-02   .87600979951E+00   .24228106890E-02
27   .16199461490E+00   .71E-11  -.67419585262E-01  -.95484639992E-04  -.99191514333E+00  -.31876596287E-02
28  -.11363627707E+00  -.47E-11   .83105948375E-01   .13832765905E-02   .88340121432E+00   .24447722180E-02
29   .15834192342E+00   .71E-11  -.66846833223E-01  -.38726443234E-03  -.93059117942E+00  -.35359927912E-02
30  -.11289069014E+00  -.47E-11   .82859357966E-01  -.12253868661E-02   .87172823008E+00   .35354637929E-02
31   .16187291785E+00   .71E-11  -.66798036112E-01   .31896237576E-03  -.99042237417E+00  -.46382090325E-02
32  -.11397232338E+00  -.47E-11   .82325376017E-01   .13143150340E-02   .88813676070E+00   .36581376418E-02
33   .15841542208E+00   .71E-11  -.66087227991E-01  -.54862858291E-03  -.93223519971E+00  -.48791893586E-02
34  -.11268585717E+00  -.47E-11   .81811500073E-01  -.10822256651E-02   .86753930897E+00   .45574443249E-02
35   .16174059152E+00   .71E-11  -.65941687626E-01   .10471959547E-02  -.98856658345E+00  -.59478554518E-02
36  -.11433852262E+00  -.48E-11   .81277120631E-01   .12368362067E-02   .89298670732E+00   .48000109949E-02
37   .15860594388E+00   .71E-11  -.65112743966E-01  -.86170430057E-03  -.93502518785E+00  -.61731643844E-02
38  -.11254651941E+00  -.48E-11   .80506809736E-01  -.92708558033E-03   .86399706108E+00   .54812308244E-02
39   .16161236955E+00   .71E-11  -.64874505062E-01   .16604680248E-02  -.98646696093E+00  -.71208634732E-02
40  -.11600476054E+00  -.47E-11   .80000000002E-01  -.22173879049E-02   .89901711612E+00   .61300314700E-02
```

Table 3.1 Normalized coordinates at the intersections of the refined QPO

```
SPCSM = .10000000000E-01  IPA =  1  1 11  1  1  1  1  1  1  1  1  1  1  0  0  0  ICO = 2
ZO =  .080000 TO = 16587.00000 NIT = 40 NHR = 1 IBETA = 1
BOUND1 = .2000E-08  BOUND2 = .3000E-09  NUMITM = 12  EPS = .1000E-05

VECTOR TS (normalized time)
 -.28814675855E+02  -.27249976526E+02  -.25725649263E+02  -.24233673082E+02  -.22702843362E+02  -.21135813058E+02
 -.19605457567E+02  -.18113754547E+02  -.16589230332E+02  -.15022596634E+02  -.13486096560E+02  -.11993584104E+02
 -.10475312405E+02  -.89101529894E+01  -.73675809736E+01  -.58731096759E+01  -.43606367621E+01  -.27979404526E+01
 -.12497033589E+01   .24745095331E+00   .17546158045E+01   .33140539968E+01   .48674935968E+01   .63682206643E+01
  .78707272214E+01   .94260777252E+01   .10983921244E+02   .12489153316E+02   .13987714326E+02   .15538202655E+02
  .17099717800E+02   .18610076483E+02   .20105502799E+02   .21650552212E+02   .23214889606E+02   .24730968285E+02
  .26224257154E+02   .27763417200E+02   .29329508167E+02   .30851624005E+02   .32343246656E+02

VECTOR TS (in mod. jul. days (1950.0))
  .16587000000E+05   .16677956129E+05   .16766565426E+05   .16853294151E+05   .16942281436E+05   .17033373065E+05
  .17122332784E+05   .17209045630E+05   .17297666375E+05   .17388734950E+05   .17478051853E+05   .17564811752E+05
  .17653069038E+05   .17744051912E+05   .17833721778E+05   .17920595545E+05   .18008515747E+05   .18099355441E+05
  .18189354618E+05   .18276384349E+05   .18363995993E+05   .18454646292E+05   .18544947891E+05   .18632185306E+05
  .18719526163E+05   .18809938844E+05   .18900496445E+05   .18987995736E+05   .19075107238E+05   .19165237280E+05
  .19256008312E+05   .19343805614E+05   .19430734896E+05   .19520548773E+05   .19611483863E+05   .19699613669E+05
  .19786418701E+05   .19875890229E+05   .19966927254E+05   .20055408000E+05   .20142116175E+05

VECTOR TS (calendar dates)
     1.00-JUN-1995      30.96-AGO-1995      27.57-NOV-1995      22.29-FEB-1996      21.28-MAI-1996      20.37-AGO-1996
    17.33-NOV-1996      12.05-FEB-1997      11.67-MAI-1997      10.73-AGO-1997       8.05-NOV-1997       2.81-FEB-1998
     2.07-MAI-1998       1.05-AGO-1998      29.72-OCT-1998      24.60-GEN-1999      22.52-ABR-1999      22.36-JUL-1999
    20.35-OCT-1999      15.38-GEN-2000      12.00-ABR-2000      11.65-JUL-2000       9.95-OCT-2000       5.19-GEN-2001
     2.53-ABR-2001       1.94-JUL-2001      30.50-SET-2001      27.00-DES-2001      24.11-MAR-2002      22.24-JUN-2002
    21.01-SET-2002      17.81-DES-2002      14.73-MAR-2003      12.55-JUN-2003      11.48-SET-2003       8.61-DES-2003
     4.42-MAR-2004       1.89-JUN-2004      31.93-AGO-2004      28.41-NOV-2004      23.12-FEB-2005
```

Table 3.2 Time vectors with the epochs of intersections of the refined QPO

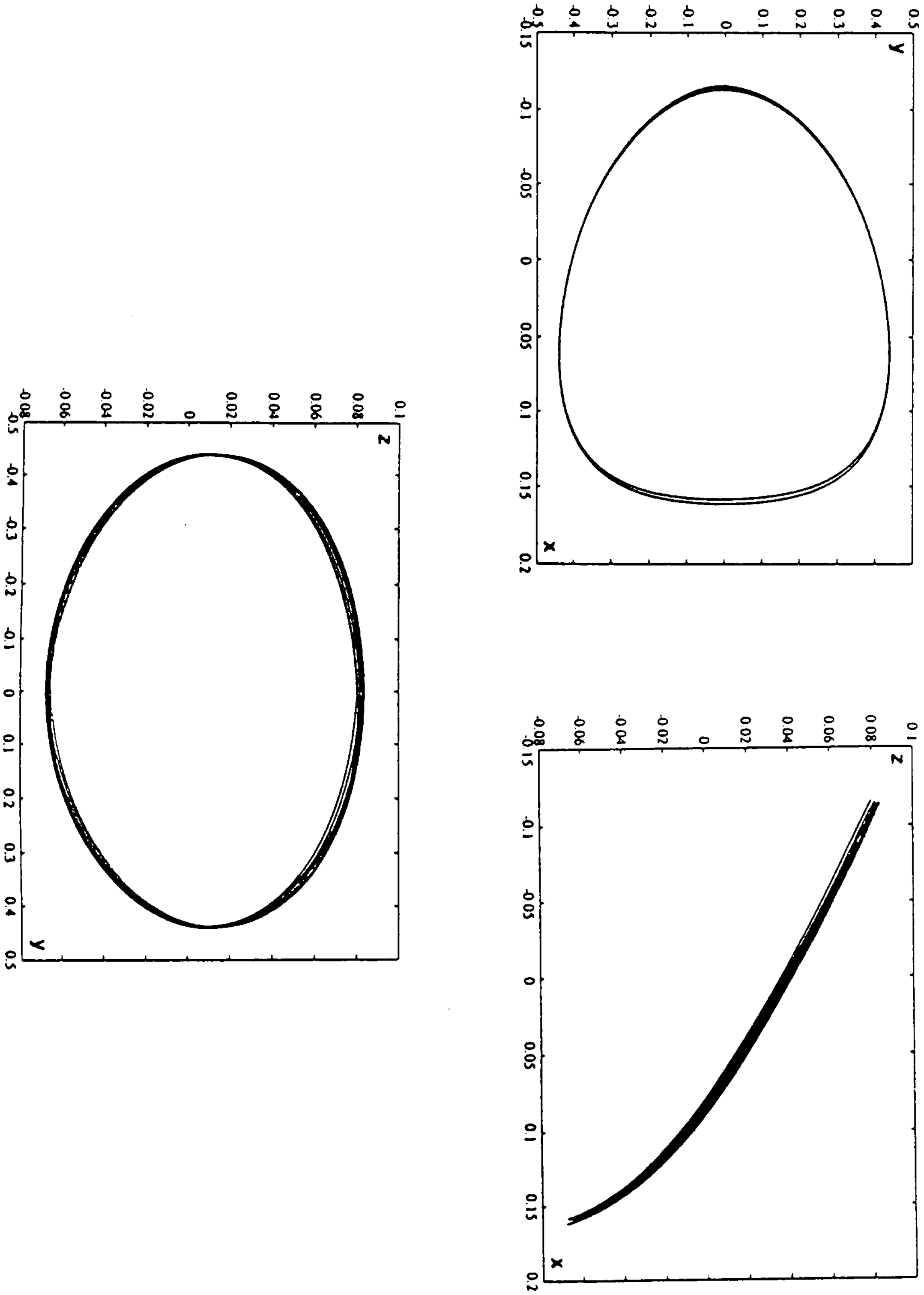


Figure 3.1 Projections of the refined quasiperiodic halo orbit

## 3.3 The Equations of Motion for the Simulations of the Control

The purpose of the next sections of this Chapter is the discussion of the numerical results obtained for the control of an spacecraft in the vicinity of a halo orbit in the Earth+Moon-Sun system. The methodology used for the definition of the control strategy is the same one that was employed in a previous work [2]. The only difference is that now all the computations have been done ussing the JPL ephemeris for the computation of the motion of the bodies of the solar system. Nevertheless the main ideas of the method have been included in the text.

The on/off controls are supposed to act in a very short time. Then, when a control is applied we merely shift the velocities by the suitable amount.

The simulation program is suited to this approach because we have used a one step method (a Runge-Kutta-Fehlberg algorithm of order 7/8 and automatic step size control).

The mass of the Sun has been altered, as explained in Appendix B of [4], to account for the radiation pressure effects.

As the effects of the radiation pressure have some uncertainties, we have added a random contribution to the acceleration given above. In order to avoid conflict with the numerical integration scheme, the random forces have been kept fixed at each step of the integration.

The "standard" random errors in the radiation pressure consist in a relative error, in the modulus of the radiation pressure acceleration, which is supposed distributed according to a centered normal law with typical deviation of 5%. Normally distributed errors with zero mean and typical deviation equal to 0.05 rad. have been used for both ecliptical latitude and longitude of the radiation pressure acceleration.

These standard values can be modified using an amplification (or reduction if less than 1) factor.

A systematic noise can be introduced in the following way. The nominal orbit is computed with a given value of s/m (spacecraft section facing to the Sun / mass). If the simulation is done with a value of s/m greater (less) than the nominal one, a net acceleration outwards (towards) the Sun will result.

In all the computations the adopted value of s/m has been 0.01. This is pessimistic because, probably, the real figure will be 1/3 of that one.

## 3.4 The Effect of Errors

The current point will be known, in the applications, through tracking. As specified by the Agency, for the $L_1$ case in the Earth+Moon-Sun problem the tracking errors have been supposed to be distributed according to centered normal laws with $2\sigma$

deviations equal to 3, 5 and 30 km, 2, 2 and 6 mm/sec in $x$, $y$, $z$, $\dot{x}$, $\dot{y}$, $\dot{z}$, respectively. Here the coordinates refer to the adimensional system.

When the simulation is started, at a given epoch, the nominal point is computed. Then, random variations in the 6 coordinates are introduced due to tracking errors according to the given laws. The integration of the motion of the spacecraft in the solar system field is done with the random errors specified in the preceeding section.

At every step the unstable component is computed, but to this computation (not to the current point) again tracking errors are applied. Hence, for a given epoch we have 3 different points: The nominal one, the real one and the estimated point.

When a too large unstable component forces to the execution of a manoeuvre the $x$ and $y$ components of the jump in velocity are computed. Then, again some execution errors are introduced. They can be due to a wrong direction for the manoeuvre or wrong modulus of the same.

The "standard" assumptions on those errors are the following. We have supposed that, if an $(x, y)$ manoeuvre should be done, the $x$ and $y$ components are modified with a centered normal deviation with $2\sigma$ value equal to 5% of the modulus of the scheduled manoeuvre. For the $z$ component (theoretically zero) we have used a 2% of the same amount. If (x) only or (y) only manoeuvres are done it is supossed, also, that the errors are of 5% in the active component and 2% in the non active ones.

Both the standard values of the tracking and the execution of the manoeuvre errors, can be modified using amplifying factors.

## 3.5 When a Control is Applied

It is clear that to do a manoeuvre when the unstable component is too small is nonsense. The small unstability can be due to tracking errors. In Chapter III of [2] it was found that, taking a box, centered at the origin, with $2\sigma$ half sides in each one of the six variables, the maximal unstable component, changing from one point in the orbit to another one, is near $2 \cdot 10^{-7}$ in adimensional units. Furthermore, the random errors in the acceleration should cancel on the average, but locally can have some importance. With the used values their amount can be estimated in $1 \cdot 10^{-7}$, approximately, in the unstable component (u.c. for shortness). Hence, manoeuvres with an u.c. less than 2, 3 or 4 times $10^{-7}$ should not be done.

A value is given by the user of the simulation program for this lower bound of the u.c. in modulus. Below this value no manoeuvre should be done.

When the u.c. is greater than the given value a manoeuvre can be useful. However, too frequent manoeuvres should be avoided because to settle down the tracking errors, some time interval is necessary. Typically one month, at least, has been asked, between two consecutive manoeuvres.

If the u.c. is greater than the lower bound and the time since last control is big enough we must be sure that the u.c. is increasing in an exponential way. If this is not so, the deviations can be due to the presence of small oscillations in the motion around the nominal orbit, as happens when an analytical one is taken. If the u.c. increases by a factor $\lambda_1$ (the dominant eigenvalue) in one period, $T$, in a step of integration, $\Delta t$, should increase, in theory by a factor $\exp(\Delta t \frac{\ln \lambda_1}{T})$. A value $m$, slightly smaller than $\frac{\ln \lambda_1}{T}$ has been used. If for three consecutive steps of integration the quotient of succesive u.c. is greater than $\exp(m\Delta t)$, then a clear exponential behavior is present. In any case an upper bound of the u.c. must be given such that, if the u.c. is greater than this upper bound, a manoeuvre is done unconditionally.

From the unstable character of the behavior it follows that the best choice (from the optimal point of view, i.e. the one such that the foreseen fuel used divided by the time interval between two consecutive manoeuvres reaches a minimum) is to perform manoeuvres at a value of the u.c. roughly equal to $e = \exp(1)$ times the sum of the total effect of the diverse errors. Using a lower bound $0.4 \cdot 10^{-6}$, a value of the upper bound of $10^{-6}$ is suitable.

## 3.6 Magnitudes Related to the Control

The control algorithm requires several magnitudes mainly related to the behaviour of the unstable manifold. These magnitudes, as well as a suitable form of the nominal orbit, are computed in a program independent of the one which performs the simulations of the control.

First of all program NUNOPF computes, for given fixed time intervals, the normalized coordinates of the points of the qpo. It also computes the values of the projection factors related to the unstable manifold from the eigendirections given by routine SDIREC. In fact this last routine computes for the initial points in each revolution the unstable and stable directions as well as a basis for the remaining four directions related to the central part. These computations are performed in such a way that under the differential of the flow, from one revolution to the next one the directions are mapped in a continous way. This is acomplished in the following way:

Asume that the qpo is splitted in N revolutions. Associated to the $i$-th revolution we have the variational matrix $A_i$ in normalized coordinates. The individual use of each one of these matrices in order to get the above directions implies a certain discontinuity when we change from one revolution to another. Nevertheless this was the approach used in [2]. In order to avoid this discontinuity it is better to use the composed matrix over the whole revolutions.

This implies some tecnichal difficulties, for example:

Due to the large value of the unstable eigenvector of each one of the matrices $A_i$ (roughly 1700) it is not possible to perform a direct computation of the eigenvalues of the matrix $A = A_N \times A_{N-1} \times \ldots \times A_1$ because of the possible overflows during the computation of $A$ as well as the big rounding errors. We must have into account that the dominant eigenvalue of $A$ is of the order of $1700^N$.

To avoid this difficulty we have proceeded as follows. For the computation of the unstable and the stable eigenvectors we have used the power method applied to the matrix $A$ and its inverse, respectively. For the remaining ones the procedure is a little bit more tricky. We start by taking four linearly independent vectors.

Assume that $u_0$ is the unstable vector of $A$. Let

$$\bar{u}_i = A_i \times A_{i-1} \times \ldots \times A_1 u_0$$

and $u_i = \frac{\bar{u}_i}{|\bar{u}_i|}$.

We define $\delta_i =\mid A_i u_{i-1} \mid$. We note that

$$u_i = \frac{A_i u_{i-1}}{\delta_i}.$$

Let $w_0^0$ be some arbitrary vector. We introduce $\bar{w}_1^0 = A_1 w_0^0$. The initial vector $w_0^0$ is modified in the following way:

$$w_0^1 = w_0^0 - \frac{\lambda_1}{\delta_1} u_0,$$

where $\lambda_1 = (\bar{w}_1^0, u_1)$. Note that $w_0^1$ has no unstable component under $A_1$. This is the first step of the procedure. In the second step we take $w_1^0 = \bar{w}_1^0 - \lambda_1 u_1$. That is: $w_1^0 = A_1 w_0^0$. Now we can compute $\bar{w}_2^0 = A_2 w_1^0$. The vector $w_1^0$ is modified in the following way:

$$w_1^1 = w_1^0 - \frac{\lambda_2}{\delta_2} u_1,$$

where $\lambda_2 = (\bar{w}_2^0, u_2)$. Note that $w_1^1$ has no unstable component under $A_2$. Then we perform another correction in the following way:

$$w_0^2 = w_0^1 - \frac{\lambda_2}{\delta_1 \delta_2} u_0,$$

Note that $w_0^2$ has no unstable component under $A_2 A_1$.

Iterating the procedure till obtaining $w_0^n$, we arrive to a vector without unstable component under $A$. If this is done with four linearly independent vectors we get a basis of the central part.

It must be noted that the stable components have not been taken into account so the procedure works only for large values of $N$, say for instance $N \geq 5$ in the problem considered.

## 3.7 Description of the Program

The program CONSIM developed in [2] has been adapted to use the JPL numerical ephemeris in order to perform the simulation of the control according to what it has been said in the preceeding sections.

For the nominal orbit and for the projection factors several possibilities are offered:

1. Numerical halo orbit (in this case the projection factors on the Floquet modes 3 and 4 are also available optionally).
2. Analytically quasiperiodic orbit (generated by routine ANAQPO from the output of programs ANACOM and QPO) and analytical parameters of control (it is enough to use the ones of the RTBP) generated by program CONA from the output of ANAVAR. See [2] for the above programs.
3. Refined numerical quasiperiodic orbit and related normalized projection factors. They are generated by programs GQPTPS and NUNOPF.

To perform checks, a nominal orbit of any kind can be used with projection factors of any kind. To allow for this possibility several additional parameters can be requested.

The determination of a point in the orbit can be done in the following ways:

1. The point in the nominal orbit is computed at the current time.
2. It is computed at a time near the current one such that the distance from the actually computed (by integration) point to the selected point in the nominal path be a local minimum. This is done writting

   $$d^2 = (x - x_n)^2 + (y - y_n)^2 + (z - z_n)^2,$$

   where $x_n$, $y_n$, $z_n$ are the nominal values and $x$, $y$, $z$ the real ones. Then the nominal ones are considered functions of time and the other ones are constant. By time differentiation

   $$\frac{1}{2}(d^2)^{\cdot} = (x - x_n)\dot{x}_n + (y - y_n)\dot{y}_n + (z - z_n)\dot{z}_n = 0.$$

   This equation is solved for $t$ using Newton's method.

The nominal orbit is computed by routine NOMORB. The type of the manoeuvres can also be selected:

1. Full optimal control ( $(x, y)$-plane manoeuvres).
2. $x$-axis manoeuvres.
3. $y$-axis manoeuvres.
4. Radiation pressure manoeuvres.

The random generator used to obtain normally distributed points (for the noise in acceleration and also for tracking and manoeuvres errors) is called a given (by the user) number of times before starting the proper computations. This allows to do statistics with all the other data fixed.

Each time a manoeuvre is done, written output is produced related to it. At the end of the run a statistics of the manoeuvres is given. The available information is:

- Number of manoeuvres.
- Total amount of $\Delta v$ required (in adimensional and physical units).
- Minimum and maximum observed time between two consecutive manoeuvres (in julian days).
- Maximal and minimal manoeuvre (in adimensional and physical units).

The normalized projection factors are computed by the routine MAGCON, and from them the unitary controls can be computed in any case.

The numerical RTBP values are read from the output of program PAPUS. They are given as Fourier series and the only thing to do is the evaluation of the series at the given angle.

The analytical values are obtained from the output of CONA. From this output, and with the $x$ and $z$ amplitudes determined when the nominal orbit is computed, a Fourier series with numerical coefficients is obtained for the projection factors, determinant and first Floquet mode. From a given angle (note that the period should be given if the orbit chosen is the numerical one) the three magnitudes can be determined and from them the normalized projection factors.

Finally, if numerically refined projection factors are used, the only thing to do is to obtain them at a given epoch using the Lagrange interpolation routine LAG.

### 3.7.1 Numerical Results

In the Table of the next page we present a summary of the results obtained for the refined halo orbit of amplitude $z_0 = 0.08$ and initial epoch $t_0 = 16587$ mjd (since 1950.0). In all the simulations the $(x, y)$ control has been used. The time interval used in the simulations is of roughly 9 years, and goes from 16677.0 mjd to 19966.0 mjd (we have skipped the first half revolution and the last revolution of the refined halo orbit).

The results are not sensitive to the value of $\beta$ (within moderate values) and are quite similar to the ones obtained in [2] when an analytical model of solar system was used.

A first look to the Table shows that, for the standard values of the different errors, at most 18.6 cm/sec per year are required. The time interval between manoeuvres ranges between 1.5 and 5 months. This is changed if the standard errors are altered. In the most pessimistic case (errors 2, 2 and 4 times the standard ones) at most 28 cm/sec per year are required.

All the computations have been done with the option: minimal distance from the nominal to the current point. In the interactive runs of CONSIM additional information appears which allows to check the values of the changes of the nominal point in time. The runs rarely exceed $2^m$ in runs covering 9 years controlled by on/off algorithm (on a HP 9000/835S).

In the table the following notation has been used:

$W^u_{max,min}$ refer to the upper and lower bounds for the unstable component.

$\Delta t$: to the desired minimum time interval between manoeuvres.

$a_i$, i=1, 2, 3: to the amplifying factors of errors in tracking, execution of manoeuvres and radiation pressure.

calls: to the number of previous calls to the random number generator.

man: to the number of manoeuvres done.

$\Delta v$: to the total amount of increment of velocity required for the time span of 9 years considered. (cm/sec).

$t_{min,max}$: to the minimum and maximum experimental values of the time interval between manoeuvres (jd).

$v_{min,max}$: to the minimum and maximum values of the manoeuvres (cm/sec).

| $W^u_{max}$ | $W^u_{min}$ | $\Delta t$ | $a_1$ | $a_2$ | $a_3$ | *calls* | *man* | $\Delta v$ | $t_{min}$ | $t_{max}$ | $v_{min}$ | $v_{max}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 1 | 100 | 40 | 161 | 47 | 150 | 1.8 | 5.9 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 1 | 400 | 41 | 170 | 45 | 149 | 2.1 | 5.9 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 1 | 700 | 41 | 163 | 50 | 150 | 2.0 | 6.3 |
| $2.D-6$ | $.8D-6$ | 60 | 1 | 1 | 1 | 101 | 34 | 233 | 60 | 264 | 4.4 | 11.9 |
| $2.D-6$ | $.8D-6$ | 60 | 1 | 1 | 1 | 401 | 36 | 230 | 63 | 165 | 3.8 | 11.4 |
| $2.D-6$ | $.8D-6$ | 60 | 1 | 1 | 1 | 701 | 37 | 232 | 62 | 161 | 3.8 | 10.8 |
| $.5D-6$ | $.2D-6$ | 60 | 1 | 1 | 1 | 102 | 47 | 106 | 31 | 117 | 0.8 | 3.0 |
| $.5D-6$ | $.2D-6$ | 60 | 1 | 1 | 1 | 402 | 47 | 106 | 31 | 117 | 0.8 | 3.0 |
| $.5D-6$ | $.2D-6$ | 60 | 1 | 1 | 1 | 702 | 43 | 98 | 39 | 129 | 0.9 | 2.9 |
| $1.D-6$ | $.4D-6$ | 30 | 1 | 1 | 1 | 103 | 40 | 156 | 46 | 167 | 2.0 | 5.9 |
| $1.D-6$ | $.4D-6$ | 30 | 1 | 1 | 1 | 403 | 38 | 140 | 38 | 177 | 1.9 | 5.6 |
| $1.D-6$ | $.4D-6$ | 30 | 1 | 1 | 1 | 703 | 40 | 152 | 42 | 175 | 2.0 | 5.6 |
| $1.D-6$ | $.4D-6$ | 60 | 2 | 1 | 1 | 104 | 46 | 197 | 36 | 149 | 1.6 | 6.0 |
| $1.D-6$ | $.4D-6$ | 60 | 2 | 1 | 1 | 404 | 40 | 172 | 56 | 150 | 1.7 | 6.6 |
| $1.D-6$ | $.4D-6$ | 60 | 2 | 1 | 1 | 704 | 48 | 213 | 30 | 148 | 2.1 | 6.4 |
| $1.D-6$ | $.4D-6$ | 60 | 4 | 1 | 1 | 105 | 65 | 326 | 20 | 101 | 2.2 | 7.0 |
| $1.D-6$ | $.4D-6$ | 60 | 4 | 1 | 1 | 405 | 60 | 303 | 11 | 156 | 1.6 | 6.8 |
| $1.D-6$ | $.4D-6$ | 60 | 4 | 1 | 1 | 705 | 49 | 229 | 33 | 176 | 1.8 | 6.1 |
| $1.D-6$ | $.4D-6$ | 60 | .5 | 1 | 1 | 106 | 40 | 150 | 60 | 139 | 2.0 | 5.4 |
| $1.D-6$ | $.4D-6$ | 60 | .5 | 1 | 1 | 406 | 37 | 135 | 60 | 193 | 2.0 | 5.5 |
| $1.D-6$ | $.4D-6$ | 60 | .5 | 1 | 1 | 706 | 39 | 136 | 54 | 183 | 2.0 | 5.7 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 2 | 1 | 107 | 42 | 177 | 42 | 163 | 2.2 | 6.3 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 2 | 1 | 407 | 41 | 180 | 43 | 159 | 2.2 | 6.3 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 2 | 1 | 707 | 40 | 173 | 38 | 185 | 2.1 | 6.1 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 4 | 1 | 108 | 49 | 231 | 32 | 209 | 2.1 | 6.8 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 4 | 1 | 408 | 47 | 211 | 29 | 133 | 1.7 | 6.5 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 4 | 1 | 708 | 47 | 205 | 33 | 141 | 2.2 | 6.1 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | .5 | 1 | 109 | 39 | 151 | 54 | 169 | 1.9 | 5.9 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | .5 | 1 | 409 | 40 | 153 | 57 | 167 | 2.0 | 5.9 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | .5 | 1 | 709 | 37 | 149 | 59 | 142 | 1.9 | 6.0 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 2 | 110 | 42 | 184 | 46 | 137 | 2.1 | 6.2 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 2 | 410 | 41 | 169 | 47 | 169 | 2.0 | 6.0 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 2 | 710 | 38 | 162 | 50 | 148 | 1.8 | 5.8 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 4 | 111 | 54 | 245 | 27 | 124 | 2.7 | 6.3 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 4 | 411 | 53 | 240 | 34 | 103 | 1.9 | 6.2 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | 4 | 711 | 51 | 229 | 35 | 139 | 2.1 | 5.8 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | .5 | 112 | 41 | 157 | 61 | 123 | 1.8 | 5.8 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | .5 | 412 | 37 | 153 | 47 | 135 | 1.9 | 6.1 |
| $1.D-6$ | $.4D-6$ | 60 | 1 | 1 | .5 | 712 | 35 | 142 | 52 | 167 | 2.2 | 6.0 |
| $1.D-6$ | $.4D-6$ | 60 | 2 | 2 | 4 | 713 | 54 | 251 | 24 | 112 | 2.4 | 7.0 |

Furthermore keping the standard values fixed (that is, $W^u_{max} = 1.\text{D-}6$, $W^u_{min} = 0.4\text{D-}6$, $\Delta t = 60$, $a_1 = a_2 = a_3 = 1$) we have done 999 simulations. The Table 3.4 shows the relevant statistics.

| | *man* | $\Delta v$ | $t_{min}$ | $t_{max}$ | $v_{min}$ | $v_{max}$ |
|---|---|---|---|---|---|---|
| mean | 39.41 | 159.13 | 51.98 | 164.96 | 2.04 | 5.93 |
| st. dev. | 2.09 | 11.82 | 5.43 | 30.00 | 0.20 | 0.21 |
| minimum | 32.00 | 115.34 | 25.46 | 110.71 | 1.51 | 5.22 |
| maximum | 46.00 | 196.91 | 62.34 | 282.08 | 3.02 | 6.57 |

Table 3.4

### 3.7.2 Discussion on the Effect of the Different Errors and Magnitudes

The standard values of $W^u_{max}$ and $W^u_{min}$ in the runs have been $10^{-6}$ and $0.4 \times 10^{-6}$. An increase in these values is not suitable. A decrease is possible without producing too near manoeuvres. This produces a reduction on $\Delta v$ of roughly 35% if $W^u_{max,min}$ are halved.

The comparison of results for different values of $a_1$ shows that the tracking errors are responsible of the most important part of the total $\Delta v$ required. Hence, it is important to reduce as much as possible the tracking errors.

The errors in the execution of the manoeuvres have a smaller contribution that the ones due to tracking and the ones which have the smallest contribution are those coming from the random errors in the radiation pressure.

If the contribution of all these three errors is set equal to zero, there is still some small amout of $\Delta v$ required (45 cm/s roughly). This is due to the error propagation during the integration.

### 3.7.3 Conclusions

The station keeping of quasiperiodic halo orbits, for the Earth+Moon-Sun system, is not a serious problem in what concerns the $\Delta v$ requirement. A dramatic reduction of the fuel consumption is achieved if the nominal orbit is a true orbit of the motion of a spacecraft in the Solar System. The tracking errors give the most important contribution to $\Delta v$ .

The problem becomes much more critical when the time scales change. This happens, for instance, for the Earth-Moon system. The methods applied in this Chapter can be used for this problem, of course, but the time scale is much shorter. This means that, using the same figures as above for the "standard" errors in tracking, execution of manoeuvres and model, the total $\Delta v$ per year will be increased by a factor around 13 and this is still feasible. However the reduction of the time interval between manoeuvres requires a fast and accurate tracking.

# Chapter 4

# Transfer from the Earth to a Halo orbit

## 4.1 Introduction

The usual way to obtain the transfer trajectories from the Earth to a halo orbit is by means of an optimization procedure. The procedure looks for an orbit between the Earth and the halo maintaining some boundary conditions and minimizing the total fuel to be spended in manoeuvres during the transfer (see [5]). The approach we propose is a geometrical one, avoiding the main procedure of optimization which gives no information about the characteristics of the problem.

Due to the strong hyperbolic character of the halo orbits, the stable manifold approaches the halo orbit in a very fast way. This fact means that if we are able to put the satellite in the stable manifold of a halo orbit, it will be close to the halo orbit (say at few kilometers) in a reasonable period of time.

Proceeding in this way we shall avoid the insertion manoeuvre into the halo orbit. So we must take into account only the insertion manoeuvre into the stable manifold and the tracking manoeuvres, in order to follow the transfer path, which of course will be needed, in any case, because we are approaching to an hyperbolic orbit.

Following this procedure, the main thing we have to look for are the approaches of the stable manifolds of the halo orbits to the Earth, since the insertion manoeuvres into the stable manifold should be done, in principle, at the moment of the departure of the satellite from the vicinity of the Earth. If this were not the case, then we had to do two manoeuvres: at the departure from the Earth and for the insertion in the stable manifold. If the last one is done far away from the Earth it should be of a small magnitude. It is difficult to predict beforehand the regions where this manoeuvre could be accomplished.

For convenience, the proposed method has been splitted and implemented in several parts. In this way, when necessary, the user can go backwards without doing all the computations from the starting point.

The steps of the method are:

- To obtain the quasiperiodic halo orbits in the solar system to be explored (program `GQPTPS`). This point is described in Chapter 3.
- To obtain the local approximation of the stable manifold near the quasiperiodic halo orbits selected (program `VECSWS`).
- To globalize, by numerical integration, the stable manifold from the local approximation obtained (program `ESTWS1`).
- To explore the manifold obtained, searching the passages near the Earth (program `SELWS`).
- To obtain, from the previous search, the ranges of the manifold well suited for the transfer (programs `AFMIN` and `INTER`).
- To study the characteristics of the orbits contained in the ranges of the previous point (program `FINES`).

## 4.2 Local Approximation of the Stable Manifold

What we are going to do is to compute the linear approximation of the stable manifold for the quasiperiodic halo orbit.

We know that if an orbit is periodic, the eigenvectors of the variational matrix computed over one period of the orbit, that is the monodromy matrix, give the directions of the tangent spaces to the invariant manifolds at the starting point. Seen in the normalized adimensional reference frame our orbit is not periodic but quasiperiodic with small deviations from the halo periodic orbit that we would see in the restricted three body problem. So, in this reference system, the eigenvector associated to the eigenvalue with smallest absolute value of the variational matrix computed over a revolution (two succesive passages through the plane $y = 0$ in adimensional coordinates), approximates quite well the stable direction when the orbit crosses $y = 0$.

We want to use the orbits computed in Chapter 3 as nominal orbits for a mission lasting a certain time span. In what concerns the transfer, we are interested in the stable manifold associated to the initial revolutions. Due to possible small bends of the manifold because of the boundary conditions in the computations of the qpo by the parallel shooting procedure, the first revolution is skipped and we shall compute the stable manifold from the begining of the second one[1]. From this point we shall compute the stable direcction in a set of points of the halo orbit usually equally spaced in the whole revolution[2] in the following way.

As in the computations of the magnitudes related to the control (Chapter 3 Section 7), instead of computing the stable direction associated to the selected revolution, by means of its variational matrix, in order to avoid the small discontinuity obtained in the stable direction when we go from one revolution to the next one, the variational matrix $\bar{A}$ associated to the whole quasiperiodic orbit has been taken as a monodromy matrix. We recall that $\bar{A}$ is a $7 \times 7$ matrix because the time was added in the equations of motion in order to get an autonomous system and its pattern is:

$$\bar{A} = \left( \begin{array}{c|cccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline g_1 & & & & & & \\ g_2 & & & & & & \\ g_3 & & & A & & & \\ g_4 & & & & & & \\ g_5 & & & & & & \\ g_6 & & & & & & \end{array} \right),$$

where $A$ is the part of $\bar{A}$ associated to positions and velocities. We note that if $v$ is an eigenvector of $A$, then $(0, v)$ is an eigenvector of $\bar{A}$. We shall compute the stable direction for the $6 \times 6$ matrix $A$.

If $A_i$ is the $6 \times 6$ matrix related to positions and velocities of a certain variational matrix $\bar{A}_i$ associated to the $i$-th revolution of a numerical qpo made of $N$ revolutions, we have:

$$A = A_N \times A_{N-1} \times \ldots \times A_1.$$

Because of possible overflows and rounding errors, due to the fact that the eigenvalue of each $A_i$ associated to the unstable manifold is of the order of 1700, the above product must not be computed. With the power method applied to

$$A^{-1} = A_1^{-1} \times A_2^{-1} \times \ldots \times A_N^{-1},$$

we can compute the eigenvector $v_1$ of the matrix $A$ associated to the stable manifold. Then the eigendirections $v_j$ related to the begining of each $j$-th revolution are computed by means of:

$$v_j = \frac{A_j^{-1} v_{j+1}}{\mid A_j^{-1} v_{j+1} \mid}, \quad j = N, \ldots, 2,$$

[1] This fact corresponds in the programs to take NVO= 3, which means to start the computation of the stable manifold in the third cut of the orbit with $y = 0$.

[2] In our computations the usual number which was taken was 250.

where $v_{N+1} = v_1$. The "contraction"[3] corresponding to the stable direction asociated to the $j$-th revolution is:

$$\delta_j = \mid A_j^{-1} v_{j+1} \mid^{-1}.$$

When we have the direction of the stable manifold at the beginning of a revolution, we can obtain the stable direction at any intermediate point in the revolution transporting this vector by means of the $6 \times 6$ differential matrix of the flow corresponding to the coordinates of position and velocity. That is, if $v_j(t_0) = v_j$ is the initial vector at time $t_0$ we have

$$v_j(t) = A_j(t) v_j(t_0),$$

where $A_j(t) = Id$.

Essentially this is the way by which the stable directions are computed. Given the vector $v_j(t_0)$ in normalized coordinates, corresponding to the stable direction at the initial point $p_j(t_0)$ of a certain revolution in the phase space in normalized coordinates, and using the routine DTRAN, $p_j(t_0)$ is converted into ecliptical coordinates, and the differential matrix, $D_{ne}$, of the transformation from normalized to ecliptical coordinates at time $t_0$ and point $p_j(t_0)$ is obtained. Then the stable direction in ecliptical coordinates, $v_j^e(t_0)$, at the image point, $p_j^e(t)$, is computed by means of

$$v_j^e(t_0) = D_{ne} v_j(t_0).$$

From now on the computations are performed in ecliptical coordinates. The differential $D_j(t)$ of the flow is given by the integration of the vectorfield given by routine DERAES, which contains also the variational equations. So we have that the stable direction in ecliptical coordinates at any point of the selected revolution can be obtained by

$$v_j^e(t) = D_j(t) v_j^e(t_0).$$

The main output of this study is $v_j^e(t)$ at some points of the qpo. It is scaled in such a way that the three components of these vectors relative to the positions have euclidean norm equal to one. So what we write is:

$$\bar{v}_j^e(t) = \frac{v_j^e(t)}{\sqrt{(v_j^e)_x^2 + (v_j^e)_y^2 + (v_j^e)_z^2}}.$$

In this way if we want to take initial conditions at a selected physical distance from the point $p_j^e(t)$ which is in the qpo, we have only to multiply $\bar{v}_j^e(t)$ by the selected distance in kilometers with the desired sign.

[3] Eigenvalue in the case of a periodic orbit

### 4.2.1 Main Program and Basic Routines

Letting aside the routines associated to the numerical integration and to the changes of coordinates, the main program and routines are the following:

**Program VECSWS** It is the main program of the local computation of the stable manifold. It selects the orbit or orbits computed by program GQPTPS, and produces a file with the stable directions at the desired points of the orbit. Through the standard output writes the main incidences of the execution of the program.

**Routine RGQTP** This routine reads the output of program GQPTPS.

**Routine SDIREC** It is used to compute the stable direction $v_j$, from the variational matrices $A_1, \ldots, A_N$, in normalized coordinates and corresponding to each revolution of the qpo.

**Routine DIRS** It computes the stable directions in ecliptical as it has been explained, at selected different epochs from the beginning of the chosen revolution.

## 4.3 Globalization of the Manifold

When the output of program VECSWS is available, the next thing we have to do is the globalization of the stable manifold. This manifold will be computed from the points and directions of the output of VECSWS, selecting all of them or only a subset.

Given a displacement, $D$, in kilometers from the selected point in the qpo, in the sense of the stable manifold which gives approaches to the Earth, and taking into account that the stable directions in the output of VECSWS are in ecliptical coordinates scaled as explained before, initial conditions $X_{ws}^0$ in ecliptical coordinates, in the linear aproximation of the manifold are given by means of:

$$X_{ws}^0 = X_{qpo} + D \cdot V_{ws},$$

where $X_{qpo}$ is the selected point of the qpo and $V_{ws}$ is the scaled stable direction in the point $X_{qpo}$. All of them in ecliptical coordinates.

It must be noted that the magnitude $D$ can not be too small, in absolute value, in order to prevent rounding errors and large integration periods of time when globalizing the manifold. However, it can not be too large because the linear approximation is good near the point $X_{qpo}$. Several tests about this value have been done. In spite of the fact that the attractive character of the manifold towards the qpo was significative even for large values of $D$, taking also into account the integration time during the globalization of the manifold, we suggest values of about 200 or 250 km (in the right sense). In our computations we took $D = 200$. This implies that when we say that we reach the qpo we mean that we are at 200 km of the nominal point $X_{qpo}$ in the qpo. We recall that to reach the qpo has no meaning because of the asymptotic behaviour of the manifold towards the orbit.

When $X_{ws}^0$ is computed, what we must do only is to integrate this initial condition backwards in time till a local minimum near the Earth is obtained. However, mainly due to collisions with the Moon and to the big size of the manifold, the algorithm must take into account several problems which can appear during the integration.

Concerning the orbits which pass near the Earth, in which we are interested on, the integration is stopped when we reach a distance of less than 6000 km from the centre of the Earth. We could use the radius of the Earth as stopping criteria, but a little bit less is better for the subsequent programs for the transfer.

If the orbit does not collide with the Earth, a Newton procedure is used in order to get the minimun distance to the Earth looking for a zero of the function

$$F(t) = \vec{r}_E(t) \cdot \dot{\vec{r}}_E(t),$$

where $\vec{r}_E(t)$ is the vector position of the satellite with respect to the Earth at the epoch $t$. If the minimum is close to the Earth (we took less than 100000 km in our computations), and during the numerical integration the distance from the orbit to the Moon is less than some amount (100000 km in our computations), another Newton procedure is started in order to get the minimum distance to the Moon. Its starting point is the value obtained during the integration.

### 4.3.1 Main Program and Basic Routines

The main program and basic routines involved in the globalization of the stable manifold are the following:

**Program ESTWS1** It is the main program of the globalization. It selects the orbits and the points in each orbit to be studied from the output of program `VECSWS`. When computing the globalization writes the results in a file, and through the standard output writes the main incidences of the execution of the program.

**Routine CIWSDS** This routine perfoms the computation of the initial conditions in the stable manifold using the formula of the previous section.

**Routine MDTWS** Is the main routine of the program. From the initial conditions obtained by routine `CIWSDS` it integrates backwards in time until it reaches the minimum distance from the orbit to the Earth. It computes also the distance from the orbit to the Moon and epochs and coordinates of both events.

**Routine DTTLLS** This routine computes the necessary things to perform the Newton algorithm for computing the minimum distance from the orbit to the Earth.

## 4.4 Selecting Passages Near the Earth

This step is a rough selection of the local minima which have the graphics of the minimum distances from the orbits of the stable manifold to the Earth, as computed by program ESTWS1. The objective is to give to the user a simple, but good approximation, of the behaviour of those minima and at the same time, to prepare the data for new computations.

All the local minima of the stable manifold below a certain amount, given in kilometers, from the centre of the Earth[4] are selected.

The stable manifold is a two dimensional manifold which can be parametrized in the following way. Once the desplacement $D$ of the previous section is selected, a point $X_{qpo}$ in the halo orbit gives an initial condition by the relation given in 4.2. Following the flow forwards or backwards we get all the points in the manifold associated to $X_{qpo}$. In this way $X_{qpo}$ can be thought as one of the parameters which generate the manifold. We shall call it the *parameter along the orbit*. The other one is the elapsed time $t - t_0$ from the initial condition $X^0_{ws}$ at time $t_0$ to the point $X_{ws}$ at time $t$ following the flow. We call it *the parameter along the flow*.

We remark that this parametrization depends on the choice of $D$ and the way in which the stable directions are scaled. If the stable directions are scaled in the usual way, a small change in $D$ produces an effect equivalent to a small change in the parameter along the orbit. That is, with a small change in $D$ we can get the same orbits of the manifold as with a small change of $X_{qpo}$. Only a small shift in the parameter along the flow will be observed. This is because the stable direction is transversal to the flow.

Due to the fact that the program ESTWS1 stops the numerical integration when a specific local minimum in the distance from the orbit to the Earth is found, our globalization of the stable manifold is, in fact, uniparametric. The graphic of minimum distances is a curve which depends only of the parameter along the flow.

Suppose that $u_n$ is a certain point $X^0_{qpo}$ which is a minimum in the plot of minimum physical distances from the stable manifold to the Earth. Because of the discretization, $u_n$ does not correspond to the value of the parameter along the orbit giving the exact minimum distance (it is between $u_{n-1}$ and $u_{n+1}$), and later on, we will be interested on a refined value. Of course if we want to compute it, we could start again the computations of the aproximation of the stable manifold near $u_n$ like in program VECSWS, and with an iterative procedure we can obtain an improved

[4] In our computations we have chosen 100000 km (see the DATA RTOL in program SELWS).

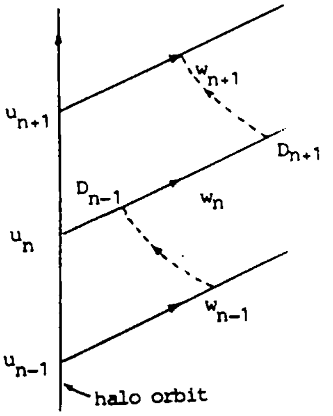
value of the parameter along the orbit. Instead of this, it is better and easier to change the parameter along the orbit, keeping $u_n$ and varying $D$. The minimum near $u_n$ will be obtained by an iterative procedure over $D$. In Section 4.2, we said that to reach the qpo means to arrive at $\mid D \mid$ kilometers of the point in which we computed the stable direction. This is not exact. However the variations in $D$ are very small if the discretization has a reasonable number of points and, for practical purposes, it does not matter.

The graph of the manifold can be strongly bent and with some discontinuities because of near passages and collisions with the Moon. It is necessary to obtain the interval in the parameter along the orbit, $D$, at which the local minimum is located, in order to prevent from possible problems in the next programs. This is done as follows.

Let $D_n$ be the displacement for which the manifold is globalized. Let $v_{n-1}$, $v_n$ and $v_{n+1}$ be the stable directions computed at $u_{n-1}$, $u_n$ and $u_{n+1}$ respectively. According to 4.3

$$w_{n-1} = u_{n-1} + D_n v_{n-1} \text{ and } w_{n+1} = u_{n+1} + D_n v_{n+1}.$$

We know that $u_{n-1}$, $u_n$ and $u_{n+1}$ are points in the halo orbit, and the flow goes from $u_{n-1}$ to $u_{n+1}$ when the time increases. In order to carry them towards the $D$ parameter we take $w_{n-1}$ and we integrate this initial condition forward till it crosses the direction $u_n + \lambda v_n$ for a certain value of $\lambda$. The $D_{n-1}$ is equal to the value of $\lambda$, and $D_{n+1}$ is computed in a similar way but following the time backwards.



### 4.4.1 Main Program and Basic Routines

The main program and routines of this section are the following:

**Program SELWS** It is the main program of the procedure. It reads the output from programs ESTWS1 and VECSWS and produces a file with the selected local minima and the main information concerning those minima.

**Routine REST** This routine is used to read the output of program `ESTWS1`.

**Routine RVECS** This routine is used to read the output of program `VECSWS`.

**Routine MARGES** Is the main routine of the program. It computes the interval in $D$ where the local minimum is contained.

## 4.5 Ranges in the Manifold Suited for the Transfer

The purpose of this section is to obtain the ranges well suited for the transfer of the parameter along the orbit on the stable manifold.

First of all the local minima selected in the previous section are improved by quadratic interpolation using the values $D_{n-1}$, $D_n$ and $D_{n+1}$, obtained by program `SELWS`, as a starting point. The iterative procedure give us the desplacement $D_m$, in the sense of the stable direction, from the point $u_n$ in the halo orbit, which gives the local minimum distance.

Then we choose a closer distance to the centre of the Earth, $H_o$, (in our computations we took $H_o = 7000$ km) and we look for the local minima which are below $H_o$. For them, the interval in the $D$ parameter around $D_m$, $(D_l^o, D_r^o)$ is computed in such a way that its boundary values $D_l^o$ and $D_r^o$ are the desplacements from $u_n$ which give the orbits in the stable manifold with the local minimum to the Earth at $H_o$ kilometers. In principle all the orbits between $D_l^o$ and $D_r^o$ could be well suited for the transfer. But due to the fact that the departure from a Geostationary Transfer Orbit is better done orthogonal to a line containing a radius of the Earth, the insertion in the orbit of the stable manifold must be done when it is just at the minimum distance. If this minimum is under a certain height, it must be discarded. Then we fix a minimum distance $H_i$ from the centre of the Earth (in our computations we took $H_i = 6400$ km) and we compute $D_l^i$ and $D_r^i$ in the same way we computed $D_l^o$ and $D_r^o$. We have another interval $(D_l^i, D_r^i)$[5] which must be removed from $(D_l^o, D_r^o)$. The obtained intervals $(D_l^o, D_l^i)$ and $(D_r^i, D_r^o)$, when $(D_l^i, D_r^i)$ is not empty, or $(D_l^o, D_r^o)$, when $(D_l^i, D_r^i)$ is empty give the ranges of the manifold well suited for the transfer.

### 4.5.1 Main Programs and Basic Routines

This section contains two main programs:

[5] This is an empty interval if $H_i$ is less than the minimum distance given by $D_m$.

**Program AFMIN** This program reads the output of the preceeding two programs: SELWS and VECSWS. Then it computes the local minima of the stable manifold. The output is written in a file.

**Routine RSEL** This routine reads the output of program SELWS.

**Routine RVECS** This routine reads the output of program VECSWS.

**Routine MDTWSQ** It is the main routine for the computation of the local minima of the stable manifold. It uses quadratic interpolation in order to find them.

**Program INTER** It is the main program for the computation of the intervals well suited for the transfer orbits. It reads the output of the programs AFMIN and VECSWS and writes its output in a file.

**Routine RAFIN** This routine reads the output of program AFMIN.

**Routine IWSMD** It is the main routine for the computations of the intervals well suited for the transfer orbits.

## 4.6 Characteristics of the Orbits Near the Earth

Finally, the last step is to study the characteristics of the orbits contained in the intervals computed in the previous section and to plot the results.

The orbital elements of the orbit, with respect to the Earth, are computed because they give the necessary and sufficient information about the orbit in the neighbourhood of the Earth. Moreover they hardly change near the Earth. Other things such are periapsis height, the epochs of insertion, and distance from the orbit to the Moon are also computed.

A thing to take into account is that in our computations the Earth has been considered as if its mass were concentrated at a single point. A good thing would be to use a more realistic Earth's potential when the orbit comes near to the Earth.

### 4.6.1 Main Program and Basic Routines

The program and routines involved in this step are the following:

**Program FINES** It is the main program for the computations of the characteristics of the orbits. It reads the output of program INTER and writes its output in a file.

**Routines RINTPA and RINTER** They are used for reading the output of program INTER.

**Routine HCTWS** This routine takes an orbit in the interval obtained by program INTER, and follows it backwards in time till it is at a certain fixed height from the centre of the Earth.

**Routine PVELEM** This routine is used to compute the orbital elements from the position and velocity given by routine HCTWS.

### 4.6.2 Some Numerical Explorations

Several runs have been done with the above mentioned software. We present here the results obtained, associated to the quasiperiodic halo orbits of z-amplitude 0.08, for values of $t_0$, the initial epoch used for the refinement of the qpo, between the epochs 16470 and 16705 (mjd).

The following table contains some insertion (on the stable manifold) windows found for this time interval:

| $T_i$ | $T_f$ | $\Delta T$ |
|---|---|---|
| 16462.45474 | 16464.33193 | 1.87719 |
| 16491.95449 | 16493.79283 | 1.83834 |
| 16522.09843 | 16523.24531 | 1.14688 |
| 16608.41647 | 16610.88113 | 2.46466 |
| 16669.11615 | 16669.81857 | 0.70242 |
| 16698.87606 | 16700.47233 | 1.59627 |

Table 4.7 Some insertion windows between 16462 and 16700 (mjd). $T_i$ is the opening date of the window, $T_f$ is the closing date and $\Delta T$ is the width of the window.

The Figures 4.1 to 4.7 show several windows with data related to the transfer orbit. They give information about several magnitudes such as epoch, inclination, velocity with respect to the Earth, $\Delta v$ for the injection from a circular orbit, minimum distance to the Moon along the transfer orbit and Sun-Earth-Moon angle at the insertion epoch for a range of perigee between 6400 and 7000 Km. We remark that the insertion windows have been selected looking only at perigee distances. In fact one should also add the requirements concerning the admissible inclinations. The distances to the Moon should be as large as possible within the window to prevent from too strong manoeuvres to correct the errors of the injection manoeuvre.

The Figures 4.8 to 4.12 display those magnitudes against the mdj-16462 for a value of the perigee equal to 6564.1 Km.

The Figure 4.13 represents a transfer orbit to the halo orbit with $t_0 = 16470$ and $\beta = 0.08$. The perigee distance for that transfer orbit is roughly 6870 Km and the ecliptic inclination is roughly 24°. The transfer takes 208.1 days from the injection till a distance of 197.5 Km to the halo orbit, but it takes less than 170 days to arrive at 1000 Km from the halo.

Finally Figure 4.14 shows the continuation of the tranfer orbit given in the previous Figure for 7 additional months.

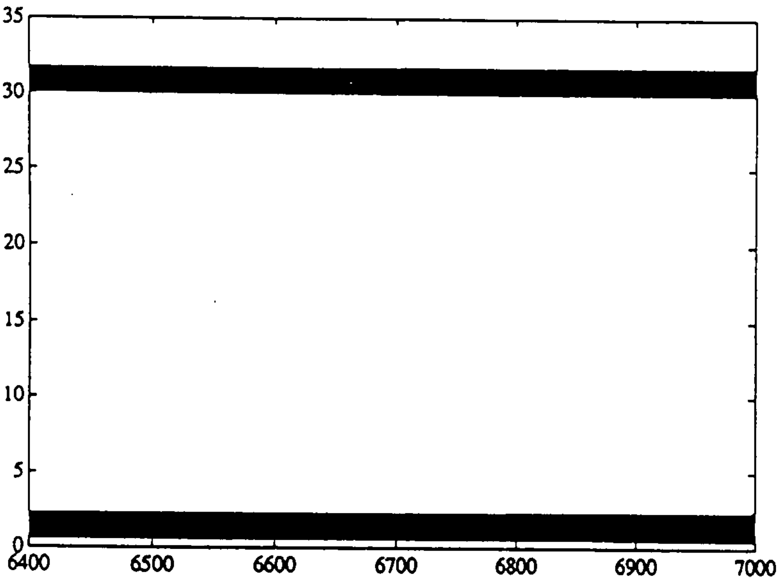


Figure 4.1 Insertion windows. The abcisas are the perigee distances in Km. The ordinates are the mjd minus 16462. For each perigee distance the admissible insertion window consists of several intervals. Here two of them are shown. Most of the other intervals are spaced by roughly 30 days.

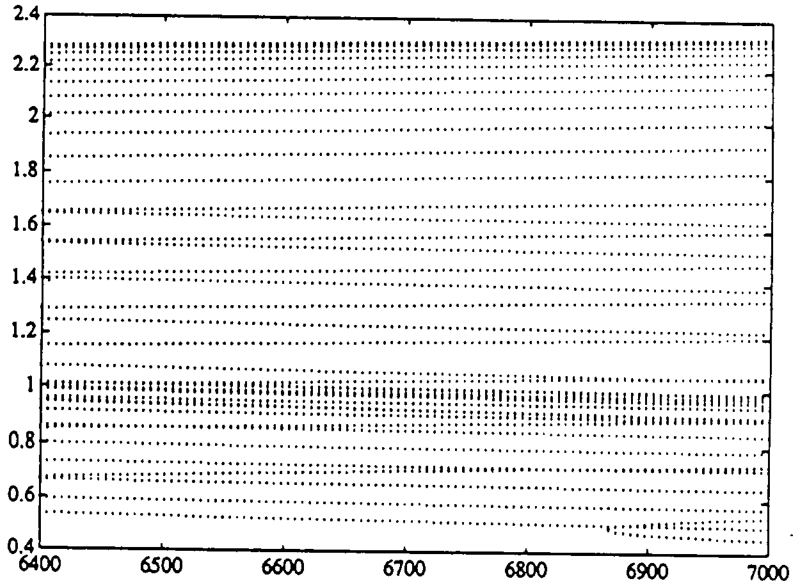


Figure 4.2 Same as Figure 4.1 but restricted to the window displayed at the bottom of Figure 4.1. In fact the strip between the lower and the upper (dotted) curves displayed here is full of curves. For convenience we only display some of them. Each, roughly horizontal, dotted curve corresponds to a given value of the initial value, $t_0$, of the refined halo orbit. Along each one of those curves the points are obtained by changing the parameter along the orbit. We remark that given $t_0$ the corresponding curve cuts twice the perigee range 6400 to 7000 Km. For instance, on the right lower part of the Figure one can see the minimum (of the perigee distance) of one of those curves. The minima of the other ones are located to the left of the Figure, outside the range of interest.

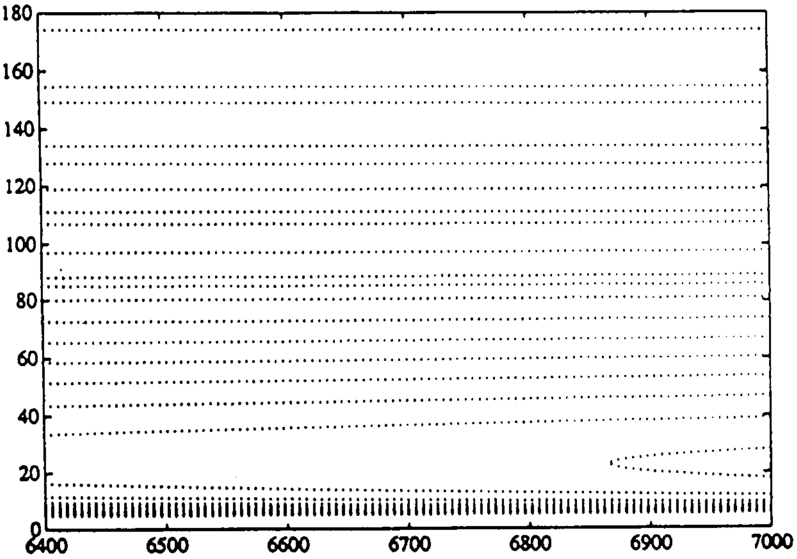


Figure 4.3 Same as Figure 4.2 displaying the inclination in degrees with respect to the ecliptic plane instead of the mjd.

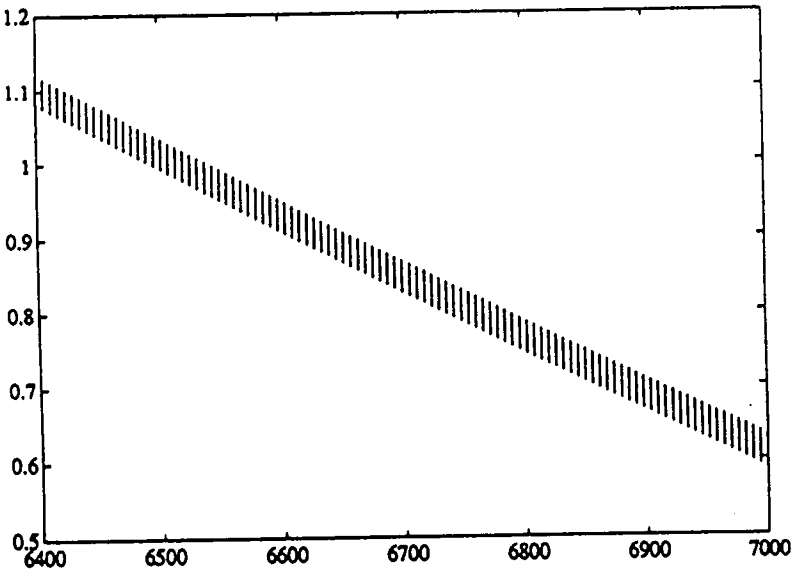


Figure 4.4 Same as Figure 4.2 displaying in ordinates the velocity with respect to the Earth at the perigee in Km/s after substracting 10 Km/s.

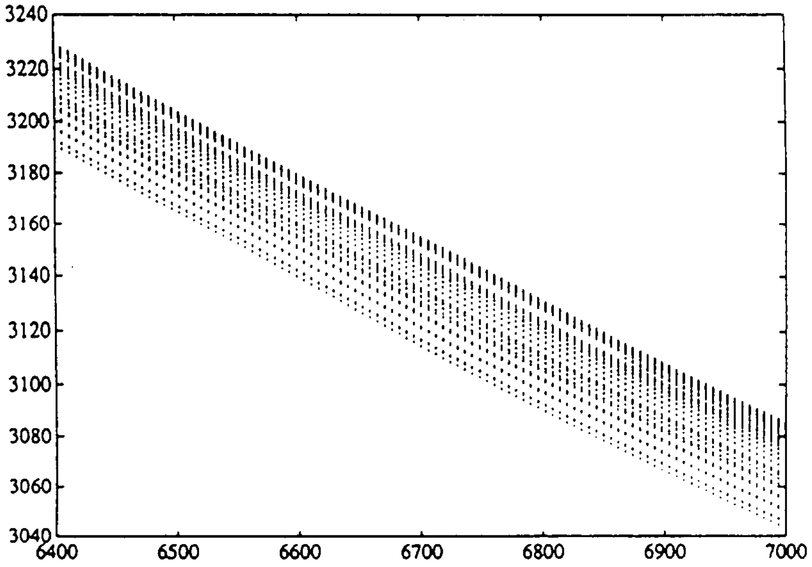


Figure 4.5 Same as Figure 4.2 displaying in ordinates the $\Delta v$ required at the perigee to insert into the transfer orbit from a circular orbit. It is given in m/s.

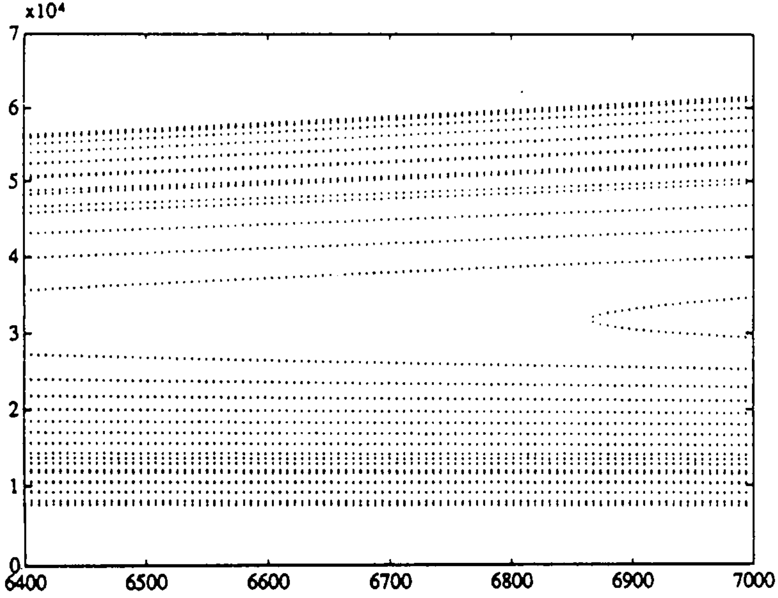


Figure 4.6 Same as Figure 4.2 displaying in ordinates the minimum distance to the Moon, in Km, along the transfer orbit.

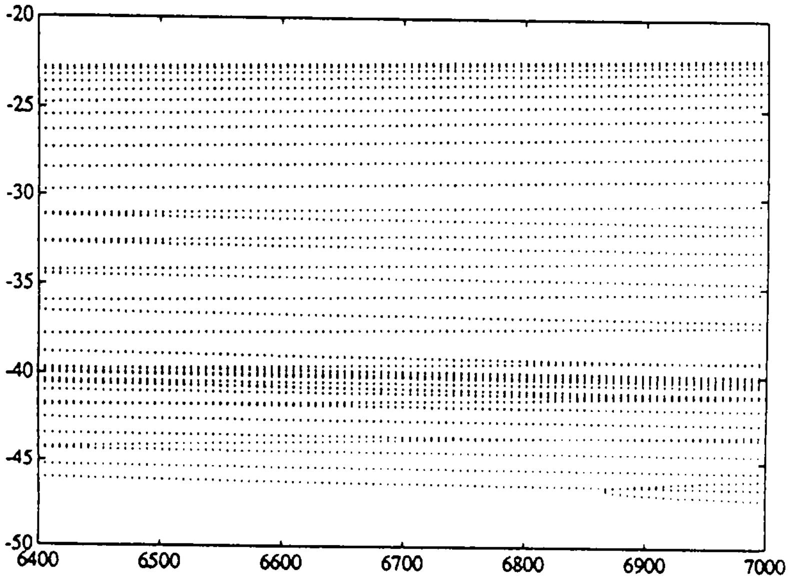


Figure 4.7 Same as Figure 4.2 displaying in ordinates the Sun-Earth-Moon angle, in degrees, at the moment of the insertion into the transfer orbit. The negative angle means that the Moon is waning.

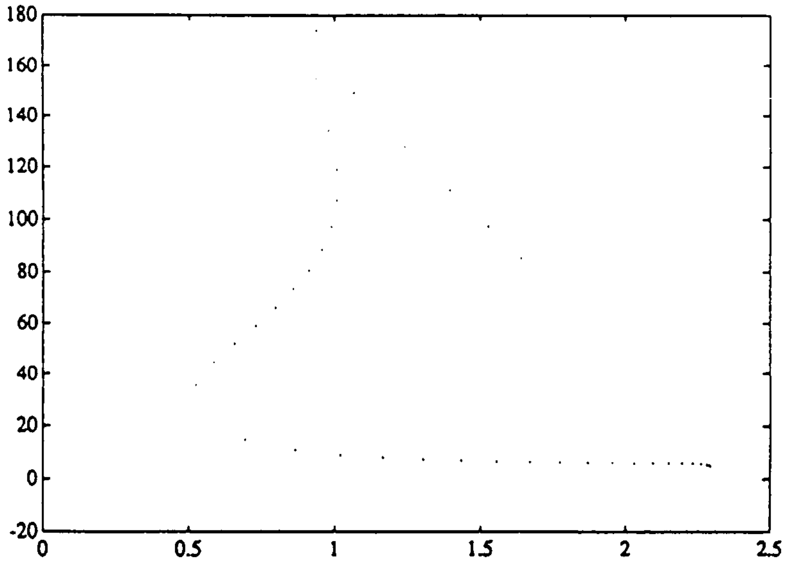


Figure 4.8 For the selected perigee distance 6564.1 Km (which was the value taken for the ISEE-3 mission) we display in abcisas the mjd 16462, and in ordinates the ecliptic inclination.

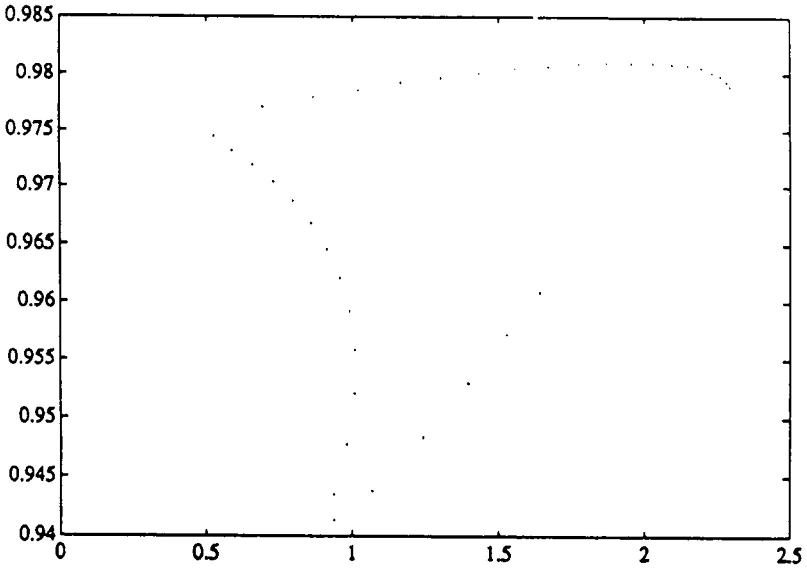


Figure 4.9 Same as Figure 4.8 but in ordinates the relative velocity to the Earth is shown (in Km/s minus 10 Km/s).

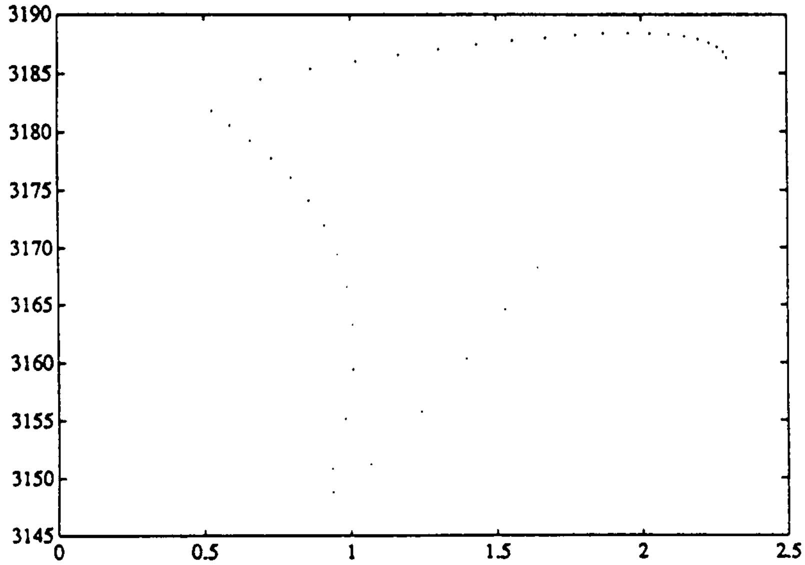


Figure 4.10 Same as Figure 4.8 but in ordinates the required $\Delta v$ to inject into the transfer orbit from a circular one is given (in m/s).

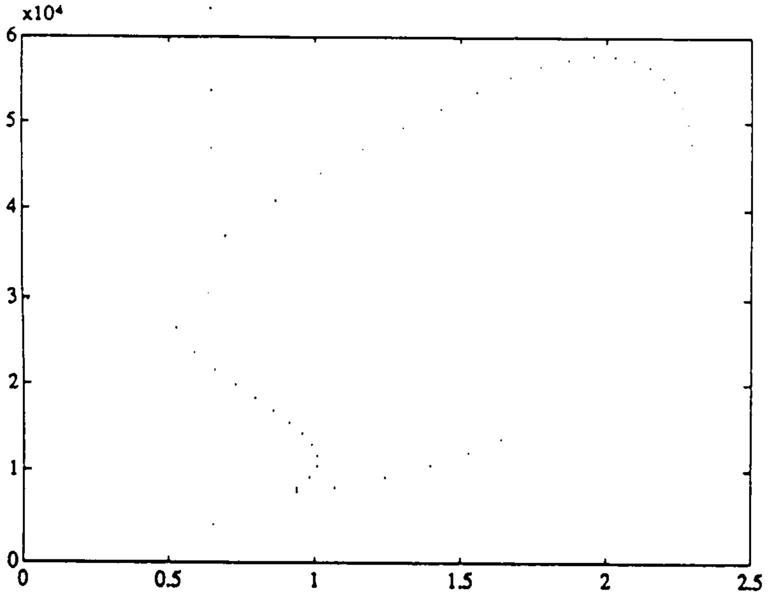


Figure 4.11 Same as Figure 4.8 but in ordinates the minimum distance to the Moon (in Km) along the transfer orbit is shown.

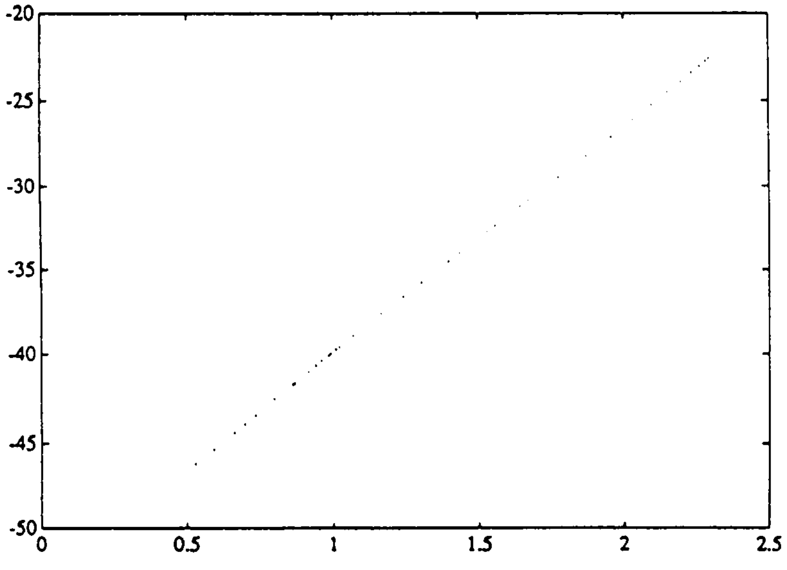


Figure 4.12 Same as Figure 4.8 but in ordinates the Sun-Earth-Moon angle at the insertion epoch is displayed.

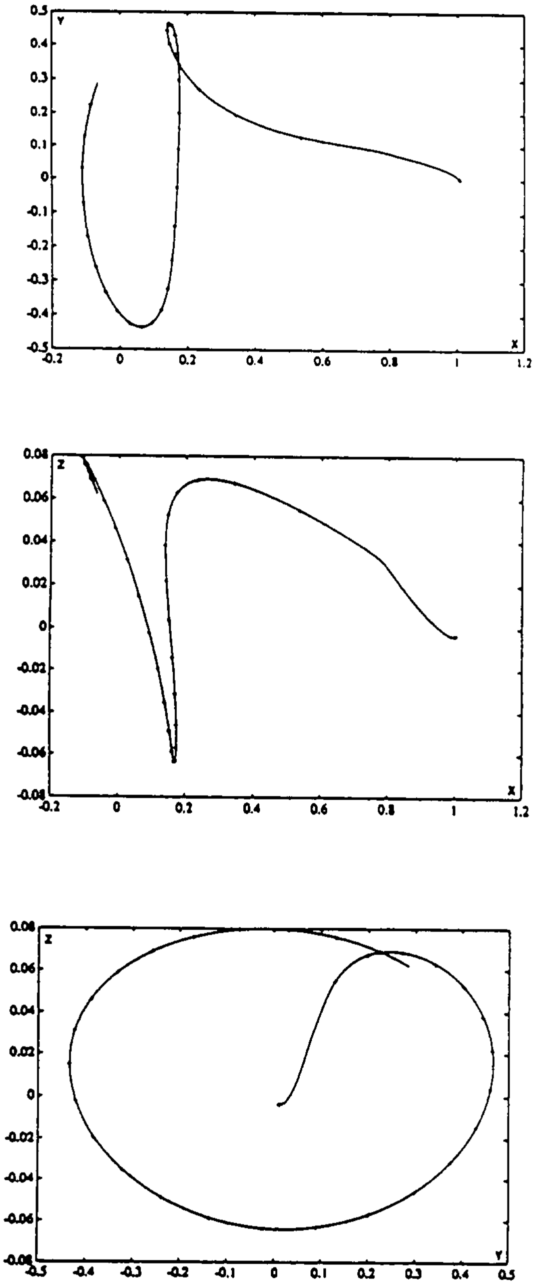


Figure 4.13 Transfer orbit for a halo orbit with $t_0 = 16470$ mjd and $\beta = 0.08$. Epoch of insertion into the transfer orbit: 16462.4603 mjd. Arrival to 197.5 Km of the halo orbit along the stable direction at the epoch 16670.5603 mjd. The crosses along the orbit denote intervals of 7 days starting at the insertion epoch.

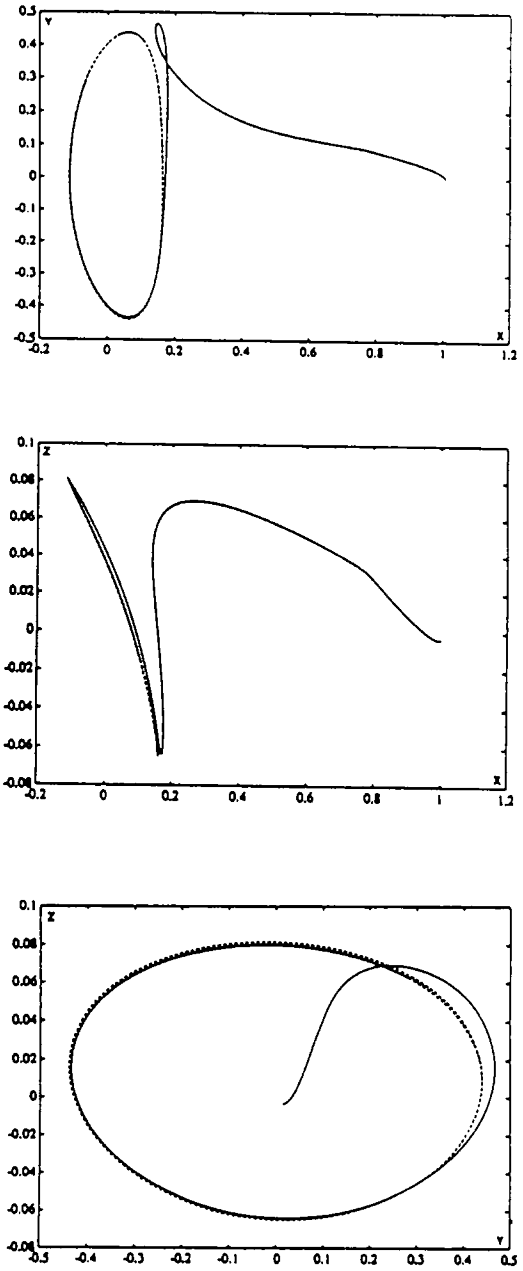


Figure 4.14 Same as Figure 4.14 but continuing the integration for more than 200 days.

## 4.7 Conclusions

The transfer from the vicinity of the Earth to a halo orbit near the $L_1$ point of the Earth+Moon-Sun system is a well known topic. However the approach using routinely a general optimization procedure gives no insight on the reasons explaining the nature of the orbit.

The Dynamical Systems approach, using the stable manifold of the orbit, gives a way to produce possible transfer orbits when the target orbit is unstable. In other words, we are asking the proper dynamics of the motion to help us in the transfer process.

# Bibliography

[1] Díez C., Jorba A. and Simó C.: A Dynamical Equivalent to the Equilateral Libration Points of the Earth-Moon Real System, to appear in *Celestial Mechanics.*

[2] Gómez G., Llibre J., Martínez R. and Simó C.: Station Keeping of Libration Point Orbits, ESOC Contract 5648/83/D/JS(SC), Final Report, xii+689 p., 1985.

[3] Gómez G., Llibre J., Martínez R. and Simó C.: Study on Orbits near the Triangular Libration Points in the perturbed Restricted Three-Body Problem, ESOC Contract 6139/84/D/JS(SC), Final Report, iv+238 p., 1987.

[4] Gómez G., Jorba A., Masdemont J. and Simó C.: Study Refinement of Semi-Analytical Halo Orbit Theory. Final Report, 213 p, April 1991. Users and Programmers Guide, February 1991. ESOC Contract 8625/89/D/MD(SC), 1991.

[5] Hechler M.: SOHO Mission Analysis $L_1$ Transfer Trajectory, MAO Working Paper No. 202, 1984.

[6] Jorba A. and Simó C.: Quasiperiodic Perturbations of Elliptic Equilibrium Points. In progress.

[7] Richardson D.L.: A Note on the Lagrangian Formulation for Motion About the Collinear Points, Celestial Mechanics 22 (1980), 231-236.

[8] Shampine L. F. and Gordon M. K.: Computer Solution of Ordinary Differential Equations. The Initial Value Problem. Freeman, 1975.

[9] Stoer J. and Bulirsch R.: Introduction to Numerical Analysis, Springer-Verlag, 1983 (second printing).