Universitat de Girona
Escola Politècnica Superior

Treball final de grau

Estudi: Grau en Enginyeria en Tecnologies Industrials

Títol: Dinàmica dels asteroides troians

Document: Memòria

Alumne: Felix Lauwaert Luyten

Tutor: Esther Barrabes Vera
Departament: Informàtica, Matemàtica Aplicada i Estadística
Àrea: Matemàtica aplicada

Convocatòria (mes/any): 09/2015

ÍNDEX

1- INTRODUCCIÓ 4

1.1- ANTECEDENTS 4

1.2- OBJECTE 4

1.3- ESPECIFICACIONS I ABAST 5

2- MODEL MATEMÀTIC 7

2.1- PROBLEMA RESTRINGIT DE TRES COSSOS. 7

2.2- COORDENADES POLARS 10

2.3- PUNTS DE LAGRANGE 12

2.3.1- Representació dels punts d'equilibri en els plans X-Y i θ-R 14

2.4- NIVELLS D'ENERGIA 15

2.5- ÒRBITES TADPOLE I HORSESHOE 17

2.5.1- Òrbites tadpole 17

2.5.2- Òrbites horseshoe 19

2.5.3- Condicions inicials 19

2.6- SECCIÓ DE POINCARÉ 21

3- METODOLOGIA 25

3.1- RESOLUCIÓ D'EQUACIONS DIFERENCIALS 25

3.2- MÈTODE D'INTEGRACIÓ NUMÈRICA: RUNGE-KUTTA-FEHLBERG. 25

3.3- CÀLCUL DE LA SECCIÓ DE POINCARÉ. 26

3.4- REPRESENTACIÓ DE LES VARIABLES POLARS. 26

3.5- PROCESSAMENT I VISUALITZACIÓ DE DADES 27

4- JUMPING TROJANS 29

5- RESULTATS 31

5.1- SOL-JÚPITER 31

5.2- SOL-TERRA 37

5.3- SOL-NEPTÚ 39

5.4- COMPARACIÓ ENTRE ELS SISTEMES SOL-JÚPITER, SOL-TERRA I SOL-NEPTÚ 42

6- RESUM DEL PRESSUPOST 44

7- CONCLUSIONS 45

7.1- PROPOSTES DE MILLORA 46

8- RELACIÓ DE DOCUMENTS 47

9- BIBLIOGRAFIA 48

ANNEXES 49

A- CODI INFORMÀTIC 49

A.1- BiseccioDef 49

A.2- BiseccioDef_JT 50

A.3- Builder 51

A.4- condIni_PtoC 53

A.5- convPol 53

A.6- formulari 54

A.7- ham 55

A.8- integrador_cart_pol_E 55

A.9- jumpingTrojans 56

A.10- moulder 58

A.11- nInt 59

A.12- ode78 59

A.13- plotter 63

A.14- plotter_moulder 66

A.15- PS_solver 67

A.16- PSexe 68

A.17- puntsLagrange 69

A.18- Tim_PS_solver 71

A.19- Tim_PS_solver_JT 71

B- FIGURES 72

B.1- Figura 20 73

B.2- Figura 23 74

B.3- Figura 24 75

B.4- Figura 25 76

B.5- Figura 27 77

B.6- Figura 28 78

B.7- Figura 29 79

B.8- Figura 30 80

B.9- Figura 31 81

B.10- Figura 32 ..... *82*

C- PRESSUPOST 83

1- Introducció

Aquest capítol es compon de dues parts: Memòria i Annexos. En la memòria es descriuen les bases teòriques i metodològiques emprades per a la realització d'aquest treball, els resultats obtinguts i l'anàlisi d'aquests. En els annexos s'hi inclou el codi de programació emprat per la obtenció dels resultats i les imatges més rellevants de les que en deriven les conclusions del treball.

1.1- Antecedents

Recentment s'ha publicat un article científic titulat *Jumping mechanisms of Trojan asteroids in the planar restricted three- and four-body problems*, realitzat per Oshima K. i Yanao T. del *Department of Applied Mechanics and Aerospace Engineering,* de la *Wasada University*, Tokio [1].

Aquest treball pren l'article anterior, que estudia els asteroides presents al sistema Sol-Júpiter, com a referència metodològica. L'objectiu al que es pretén arribar és comprovar els mecanismes que permeten l'existència d'asteroides troians que salten d'un camp a un altre que s'estudien en l'esmentat article i estendre'n l'estudi a altres casos (Sol-Terra i Sol-Neptú).

L'autor d'aquest treball ha triat aquest tema per permetre-li obtenir una base de coneixement de sistemes dinàmics i tècniques de programació per realitzar un màster de matemàtiques minimitzant les carències pròpies d'haver realitzat un grau en enginyeria.

1.2- Objecte

Els asteroides troians són asteroides que comparteixen òrbita amb un planeta del sistema solar. No orbiten al seu voltant com fan les llunes, sinó que ho fan al voltant d'uns punts d'equilibri anomenats punts de Lagrange triangulars, que es troben en dues posicions per davant i per darrere del planeta en qüestió, i són una característica del model restringit de tres cossos, que és el model matemàtic en el que es basa el treball (d'ara endavant, R3BP) (veure figura 1).

Els asteroides troians segueixen trajectòries capgròs (tadpole en anglès). Aquests es troben repartits en dues zones diferents dins el sistema Sol-planeta i s'anomenen camp grec i camp troià. Per altra banda, existeixen trajectòries de ferradura (horseshoe en anglès) que es mouen d'un camp a l'altre de manera cíclica. D'ara endavant s'utilitzarà la terminologia anglesa per classificar les trajectòries dels

asteroides. A la figura 1 es poden observar diversos punts en que els asteroides segueixen òrbites de tipus tadpole.

Per altra banda, existeixen alguns asteroides que semblen seguir una trajectòria tipus tadpole però en un moment determinat canvien la seva òrbita i salten d'una zona de tadpoles fins a l'altra. Aquests asteroides tan particulars s'anomenen *Jumping Trojans* i són l'objecte d'aquest treball, que pretén entendre què provoca el canvi de camp en el cas Sol-Júpiter i investigar la possibilitat de trobar casos similars als sistemes Sol-Terra i Sol-Neptú.

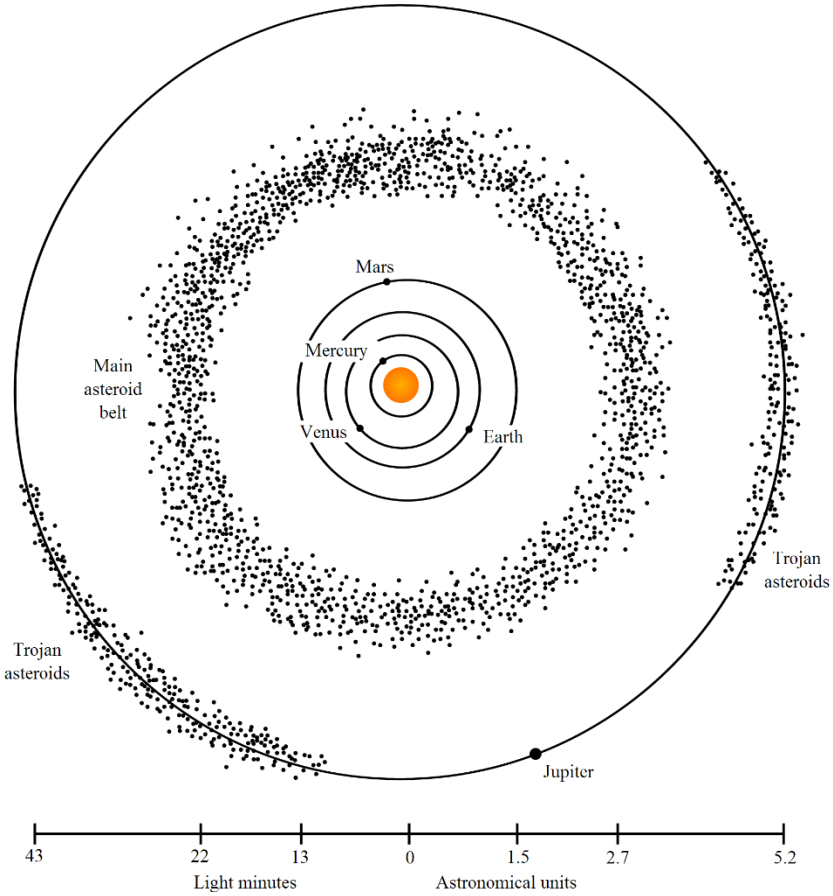


Figura 1: Representació d'asteroides troians de tipus tadpole. Els asteroides amb trajectòries tipus horseshoe envolten tots dos grups d'asteroides amb òrbites tadpole i passen pel costat oposat a Júpiter del Sol, creant així una forma de ferradura. FONT: [1]

1.3- Especificacions i Abast

Per a la realització de l'estudi es simularan òrbites anomenades horseshoe i tadpole i es compararan. Basant-se en l'estudi realitzat en l'article *Jumping mechanisms of Trojan asteroids in the planar restricted three- and four-body problems*, es pretén obtenir un mètode que permeti identificar, en primera instància, les probabilitats de l'existència de *jumping trojans* per diferents sistemes Sol-

planeta. No es pretén determinar amb exactitud possibles òrbites de *jumping trojans*, sinó comprovar la seva existència i analitzar qualitativament les probabilitats de trobar-ne per diferents planetes.

2- Model matemàtic

En aquest capítol s'exposarà el model matemàtic amb el que s'estudiarà el comportament dels asteroides troians i en el que se'n calcularan les òrbites que recòrren. Aquest model s'anomena "Problema restringit de tres cossos". També s'expliquen algunes particularitats d'aquest que permeten estudiar els asteroides troians.

2.1- Problema restringit de tres cossos

El problema restringit de tres cossos, anomenat en anglès *restricted three body problem* (d'ara endavant, R3BP) és un model definit com un sistema d'equacions diferencials autònom (no depèn explícitament del temps).

És d'una importància clau pel modelat del moviment d'un cos petit (una sonda, una nau espacial o un asteroide, per exemple) sota l'acció gravitatòria del Sol i un planeta. Aquest model dóna una molt bona aproximació del moviment real d'aquests cossos, que es consideren una pertorbació del R3BP. Per tant, entenent el comportament d'un cos dins el model es pot entendre el sistema real.

El R3BP considera dues masses principals interactives m_1 i m_2. També hi ha una tercera massa m_3 que es considera prou petita com per negligir la seva influència gravitacional en les dues principals i que el seu moviment ve definit únicament per les forces gravitacionals exercides per les dues masses principals (el cos el moviment del qual es vol simular).

Entre les masses principals, cal distingir la massa més gran, m_1, de la massa secundària m_2. Així, m_1 sol fer referència al Sol i m_2 al planeta que es consideri oportú.

En el R3BP es considera que totes tres masses orbiten en el mateix pla, i que les masses principals m_1 i m_2 tenen òrbites circulars respecte el seu centre de masses.

Les equacions diferencials que defineixen el R3BP són [7]:

$$\begin{aligned} \ddot{x} - 2\dot{y} &= -\bar{U}_x \\ \ddot{y} + 2\dot{x} &= -\bar{U}_y \end{aligned} \qquad \textbf{(2.1.1)}$$

On x i y respresenten la posició de m_3 dins el sistema en rotació (es considera que els eixos de coordenades roten amb m_2 mantenint així aquesta massa sobre el mateix punt a $x = 1 - \mu$) tal i com es mostra a la figura 2.

En aquest sentit, cal observar que les coordenades x i y es troben escalades per tal de fer coincidir la distància entre m_1 i m_2 amb la unitat, sigui quina sigui la distància real entre aquests dos cossos.

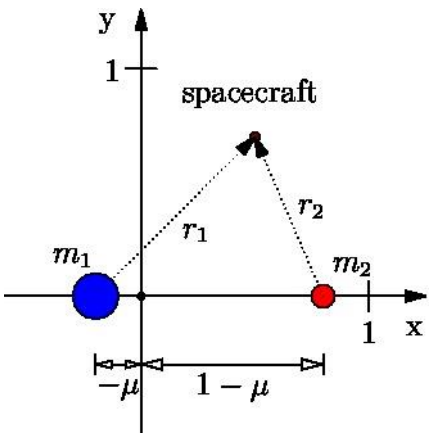


Figura 2: representació dels eixos de coordenades dins el R3BP.
FONT: http://jat.sourceforge.net

Per altra banda, es defineix el potencial efectiu com:

$$\bar{U} = -\frac{1}{2}(x^2 + y^2) - \frac{1-\mu}{\sqrt{((x+\mu)^2) + y^2}} - \frac{\mu}{\sqrt{(x-1+\mu)^2 + y^2}} \qquad \textbf{(2.1.2)}$$

On $\bar{U}_i$ es considera la derivada parcial $\partial\bar{U}/_{\partial i}$.

Per altra banda, el paràmetre μ representa el paràmetre de massa i es defineix com:

$$\mu = \frac{m_2}{m_1 + m_2} \qquad \textbf{(2.1.3)}$$

En aquest treball, es consideraran 3 valors de μ diferents, en funció del sistema Sol-Planeta que s'estigui estudiant. Així, els valors que prendrà μ seran:

Pel sistema Sol-Júpiter (S-J), $\mu = 9.537 \cdot 10^{-4}$

Pel sistema Sol-Terra (S-T), $\mu = 3.036 \cdot 10^{-6}$

Pel sistema Sol-Neptú (S-N), $\mu = 5.151 \cdot 10^{-5}$

El R3BP és un sistema conservatiu, en que l'energia constant del cos amb massa negligible m_3 és:

$$E = \frac{1}{2}(\dot{x}^2 + \dot{y}^2) - \frac{1}{2}(x^2 + y^2) - \frac{1-\mu}{\sqrt{(x+\mu)^2 + y^2}} - \frac{\mu}{\sqrt{(x-1+\mu)^2 + y^2}} \qquad \textbf{(2.1.4)}$$

Suposant ara:

$$x = x_1$$
$$y = x_2$$

(2.1.5)

Llavors:

$$\dot{x} = \dot{x}_1 = x_3$$
$$\dot{y} = \dot{x}_2 = x_4$$

Derivant, aplicant les equacions de 2.1.5 i les equacions de 2.1.1 s'obté:

$$\begin{aligned} \dot{x}_1 &= x_3 \\ \dot{x}_2 &= x_4 \\ \dot{x}_3 &= -\overline{U}_x + 2x_4 \\ x_4 &= -\overline{U}_y - 2x_3 \end{aligned} \qquad \textbf{(2.1.6)}$$

Introduint-hi les derivades parcials de equació 2.1.2, s'obté:

$$\begin{aligned} \dot{x}_1 &= x_3 \\ \dot{x}_2 &= x_4 \\ \dot{x}_3 &= 2x_4 + x_1 + \frac{(\mu-1)(\mu+x_1)}{\sqrt{({x_2}^2 + (\mu+x_1)^2)^3}} - \frac{\mu(\mu+x_1-1)}{\sqrt{({x_2}^2 + (\mu+x_1-1)^2)^3}} \\ \dot{x}_4 &= -2x_3 + x_2 + \frac{(\mu-1)x_2}{\sqrt{({x_2}^2 + (\mu+x_1)^2)^3}} - \frac{\mu x_2}{\sqrt{({x_2}^2 + (\mu+x_1-1)^2)^3}} \end{aligned} \qquad \textbf{(2.1.7)}$$

Aquest sistema de quatre equacions diferencials es compon de les equacions a integrar per resoldre el R3BP. Així, com a solució s'obtenen els valors de 2.1.5 corresponents a *x, y*, $\dot{x}$ i $\dot{y}$ per diferents instants de temps. El significat d'aquests valors és el següent:

- $[\,x\,, y\,]$ representen la posició d'un cos respecte el centre de masses del R3BP $[m_1\,,\ m_2]$ (veure figura 2).
- $[\,\dot{x}\,, \dot{y}\,]$ representen les taxes de variació de x i y respectivament.

Les unitats temporals estan establertes de tal manera que 2π unitats es corresponen a un període orbital de m_2 respecte el centre de masses. Aquest fet, juntament amb l'escala a la que es troben x i y, són els motius pels quals no apareix la constant de gravitació universal G dins el R3BP.

2.2- Coordenades polars

El problema restringit de 3 cossos està expressat, com es pot veure, en coordenades cartesianes.

Per a la realització d'aquest treball ha estat necessària la conversió dels resultats obtinguts pel R3BP a coordenades polars. Aquestes coordenades s'adapten molt bé a aquest model ja que considera moviments circulars pels dos cossos principals. La relació entre coordenades cartesianes i polars s'observen a la figura 3, on es representen r i θ dins els eixos cartesians. Els punts L_i s'expliquen a l'apartat 2.3.

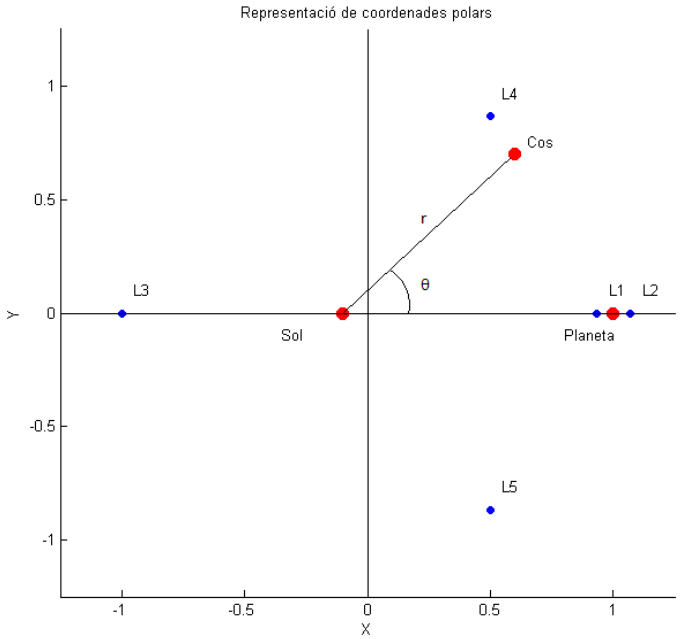


Figura 3: Representació de les coordenades polars dins els eixos cartesians.

Les equacions corresponents a la conversió entre coordenades polars i cartesianes [7] són:

$$\begin{aligned} x &= r\cos\theta - \mu \\ y &= r\sin\theta \\ \dot{x} &= \dot{r}\cos\theta - r\dot{\theta}\sin\theta \\ \dot{y} &= \dot{r}\sin\theta + r\dot{\theta}\cos\theta \end{aligned} \qquad \textbf{(2.2.1)}$$

Per altra banda, la conversió inversa (de coordenades polars a cartesianes), també ha estat important per la elaboració d'aquest treball. Les equacions que permeten aquesta conversió són:

$$r = \sqrt{(x+\mu)^2 + y^2}$$

$$\theta = \tan^{-1}\left(\frac{y}{x+\mu}\right) \qquad \textbf{(2.2.2)}$$

$$\dot{r} = \frac{(x+\mu)\dot{x} + y\dot{y}}{\sqrt{(x+\mu)^2 + y^2}}$$

$$\dot{\theta} = \left[\frac{\dot{y}}{x+\mu} - \frac{y\dot{x}}{(x+\mu)^2}\right] cos^2\theta$$

El significat de les variables polars és el següent:

- r representa la distància entre m_1 i m_3, sent m_1 la massa principal i m_3 el cos de massa negligible. Una unitat de r representa la distància entre m_1 i m_2. Pel cas de la Terra, r coincideix amb 1 U.A..
- θ representa l'angle que formen l'eix que uneix m_1 amb m_2 i el que uneix m_1 amb m_3, en sentit dextrogir.
- $\dot{r}$ i $\dot{\theta}$ representen les taxes de variació de les variables r i θ respectivament.

Les unitats de θ són els radians. Les taxes de variació es troben expressades en $unitats/unitat\ de\ temps$.

Durant la obtenció del valor de θ cal tenir cura de considerar el quadrant en que es troba m_3, ja que les coordenades cartesianes del primer i tercer quadrant es confonen. De la mateixa manera, també es confonen el segon i quart quadrants. Això és degut a que la divisió $\frac{y}{x+\mu}$ tindrà el mateix resultat en cas que el numerador o el denominador tinguin signe positiu (quart i segon quadrants, respectivament). La divisió també tindrà el mateix resultat si cap valor o els dos són negatius (primer i tercer quadrant respectivament). Per tant, durant la conversió de coordenades cartesianes a polars cal tenir en compte el signe de cada component i corregir el valor de θ resultant en cas que sigui necessari.

Una relació important en la elaboració d'aquest treball es pot obtenir de les relacions presentades a les equacions de 2.2.1. Operant, s'aconsegueix:

$$\dot{x}^2 + \dot{y}^2 = \dot{r}^2 + r^2\dot{\theta}^2 \qquad \textbf{(2.2.3)}$$

Aquesta relació és important en cas de voler obtenir les coordenades cartesianes a partir de les polars sense especificar el valor de $\dot{r}$ i en el seu lloc donar un valor per l'energia. En tal cas, es pot introduir aquesta relació a la equació 2.1.4 i obtenir així el valor de $\dot{r}$ que permetrà trobar $\dot{x}$ i $\dot{y}$.

2.3- Punts de Lagrange

En aquest apartat s'introdueix el concepte de punts de Lagrange. A més, s'exposen teòricament els principis físics que permeten la seva existència i la necessitat d'estudiar-los per entendre la dinàmica dels asteroides Troians.

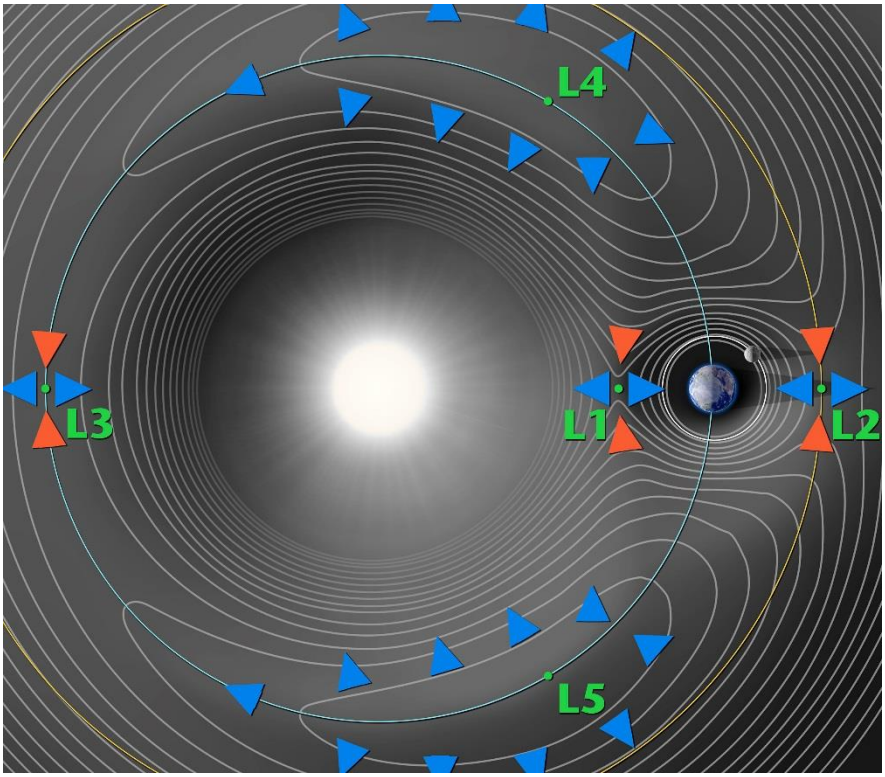


Figura 4: Representació de la situació dels 5 punts de Lagrange en el sistema Sol-Terra
FONT: [4]

L'any 1772, el matemàtic Joseph-Louis de Lagrange, mentre treballava per trobar una manera de calcular la interacció gravitatòria d'un nombre arbitrari de cossos en un sistema, va descobrir 5 punts dins el R3BP on un cos de massa negligible es troba sotmès a força zero. Aquests 5 punts s'anomenen L_1, L_2, L_3, L_4 i L_5. La situació en l'espai d'aquests punts s'observa a la figura 4. Tot i que la figura representa el sistema Sol-Terra, la distribució dels punts de Lagrange és aplicable a qualsevol combinació de masses principals dins el R3BP. Cal dir que a la figura 4 s'han modificat les distàncies entre els punts i les masses principals per permetre un millor enteniment de la situació aproximada d'aquests.

Matemàticament, el comportament de les òrbites al voltant dels punts d'equilibri es pot estudiar calculant els valors propis (VAPs) de cada punt. Així, un punt de Lagrange amb coordenades $X' = F(X_0)$ és un punt d'equilibri i els VAPs de la matriu $\nabla F(X_0)$ indiquen si aquest punt és linealment estable o no.

D'aquesta manera es pot determinar si el punt d'equilibri és una font (inestable o repulsiu en totes direccions), un pou (estable o atractiu en totes les direccions), un punt de sella (atractiu i repulsiu alhora, depenent de la direcció) o centre (un cos tendeix a traçar una trajectòria corba al seu voltant).

A més, cada punt es deu a uns principis físics propis:

- L_1 es troba entre m_1 i m_2, exactament al punt on la força gravitatòria de cada massa es compensa amb l'altra. Tal i com es veu a la figura 4, L_1 és un punt de sella. Es tracta, per tant, d'un punt inestable, ja que un cos situat a les proximitats de L_1 acaba allunyant-se d'ell.

- L_2 es troba sobre la línia que uneix m_1 i m_2, situat al costat oposat de m_1 respecte m_2. En aquest punt, el període orbital d'un cos respecte el centre de masses seria major al de la massa secundària, però gràcies a la atracció gravitacional addicional de m_2, el període de m_3 es veu accelerat fins a ser idèntic a la de m_2. Tal i com es veu a la figura 4, L_2 és un punt de sella. Es tracta, per tant, d'un punt inestable, ja que un cos situat a una distància major (o menor) a m_2 tendiria a tenir un període orbital superior (o inferior) al de m_2, allunyant el cos de L_2.

- L_3 es troba al costat oposat de de m_2 respecte m_1. En aquest punt, la atracció gravitacional combinada de les dues masses principals provoca que el període orbital sigui igual al de m_2. La distància de L_3 al centre de masses del R3BP és igual a la distància entre les dues masses principals. Tal i com es veu a la figura 4, L_3 és un punt de sella. Es tracta, per tant, d'un punt inestable, ja que un cos situat a les proximitats de L_3 però a una òrbita superior (inferior) respecte el centre de masses, augmenta (disminueix) el període orbital, allunyant així el cos del punt d'equilibri.

- L_4 i L_5 es situen sobre la mateixa òrbita que L_3, però a les posicions simètriques en què la

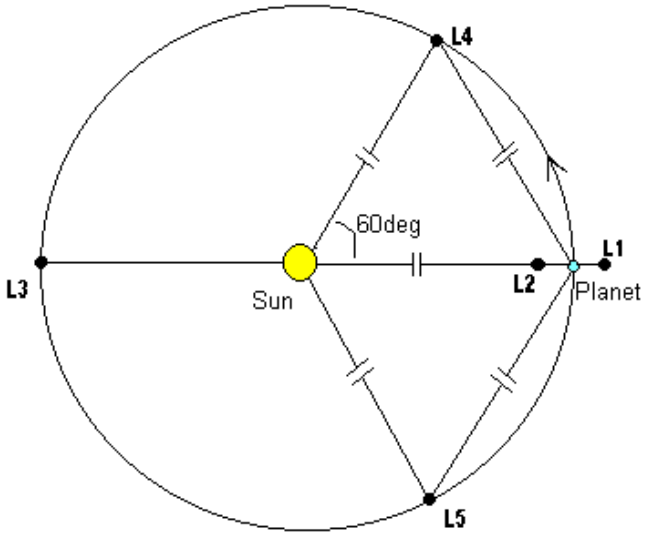


Figura 5: Posició dels punts L_4 i L_5 dins el R3BP. FONT: [1]

distància a m_1 és igual a la distància a m_2. La figura 5 representa bé la posició de L_4 i L_5. Aquests dos punts són estables. Quan un cos tendeix a allunyar-se de L_4 o L_5, l'acceleració de Coriolis creada pel sistema en rotació corba la trajectòria d'aquest, creant així una òrbita amb forma de fava que rodeja el punt d'equilibri.

Les coordenades de cada punt de Lagrange es poden trobar aplicant les equacions següents:

$$x_{L1} = 1 - \left(\frac{\mu}{3}\right)^{1/3} + \frac{1}{3}\left(\frac{\mu}{3}\right)^{2/3} - \frac{26}{9}\frac{\mu}{3}$$

$$y_{L1} = 0$$

$$x_{L2} = 1 + \left(\frac{\mu}{3}\right)^{1/3} + \frac{1}{3}\left(\frac{\mu}{3}\right)^{2/3} - \frac{28}{9}\frac{\mu}{3}$$

$$y_{L2} = 0$$

$$x_{L3} = -1 - 5\frac{\mu}{12} \qquad (3.1.1)$$

$$y_{L3} = 0$$

$$x_{L4} = x_{L5} = \frac{1}{2} - \mu$$

$$y_{L4} = -y_{L5} = \frac{\sqrt{3}}{2}$$

Els asteroides troians són aquells que orbiten al voltant dels punts d'equilibri L_4 i L_5, realitzant òrbites tadpole (veure figura 4). Aquests seran, per tant, els punts més interessants al llarg d'aquest treball.

2.3.1- Representació dels punts d'equilibri en els plans X-Y i θ-R

A l'apartat 2.2 s'ha introduït la representació d'òrbites en coordenades polars. En aquest apartat es mostrarà un altre avantatge de l'ús d'aquest tipus de coordenades.

A la figura 6 es pot apreciar com queden representats els punts de Lagrange en els sistemes de coordenades cartesià i polar. Com que els punts de Lagrange L_1 i L_2 són de poc interès, en coordenades polars es pot ajustar la mida dels gràfics per estudiar únicament el que passa a les proximitats de L_3, L_4 i L_5, mentre que per aconseguir el mateix en coordenades cartesianes seria més complicat.

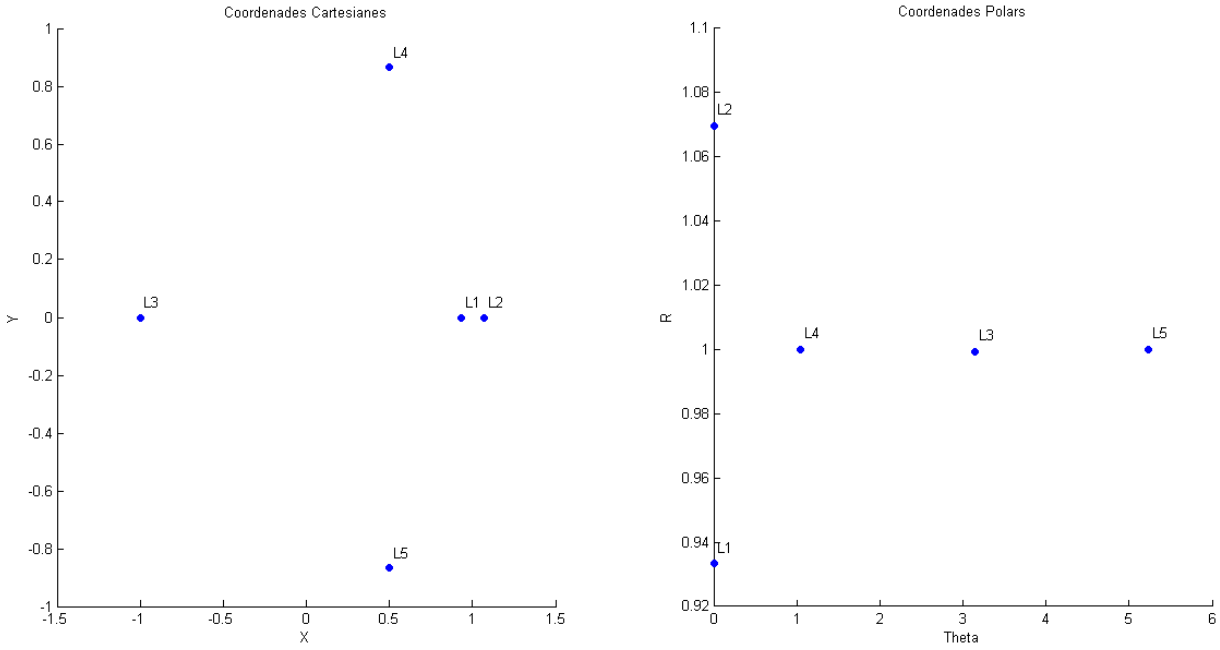


Figura 6: Comparació entre la representació cartesiana (esquerra) i polar (dreta) dels punts de Lagrange

2.4- Nivells d'energia

Com ja s'ha explicat, un cos situat sobre un punt de Lagrange experimenta una força igual a zero. En aquest sentit cal introduir un nou concepte: l'energia d'una trajectòria.

Com que el R3BP és un sistema conservatiu, l'energia d'un cos es conserva al llarg de tot el seu recorregut. Per tant, es pot estudiar el R3BP per capes: definint diferents nivells d'energia es pot estudiar com són les òrbites que viuen en cada nivell.

Planeta	Terra	Júpiter	Neptú
L1	-1,500449	-1,519378	-1,502909
L2	-1,500447	-1,518742	-1,502875
L3	-1,500002	-1,500477	-1,500026
L4	-1,499998	-1,499524	-1,499974
L5	-1,499998	-1,499524	-1,499974

Taula 1: Energia dels punts de Lagrange per cada sitema Sol-planeta.

L'equació 2.1.4 permet calcular l'energia d'una trajectòria dins el R3BP. D'aquesta manera, es pot avaluar també l'energia que tindrian els punts d'equilibri. Així, l'energia dels punts de Lagrange per cada planeta que s'estudiarà en aquest treball queda reflectida a la taula 1. Com es pot veure, l'energia de cada punt és creixent des de L_1 fins L_4, sent aquesta igual a la de L_5.

El fet que els punts de Lagrange tinguin una energia característica té una importància clau en l'estudi dels asteroides troians i, en general, de qualsevol estudi de transport de matèria dins el context del R3BP.

El fet que el R3BP sigui un sistema conservatiu implica que no tots els asteroides seran capaços de recórrer qualsevol tipus d'òrbita, sinó que depenent del seu nivell d'energia, tindran restringides certes regions a l'espai, tal i com es pot observar a la figura 7. Aquesta figura s'ha obtingut a partir de la equació 2.1.4. Arreglant la equació es troba:

$$\frac{1}{2}(\dot{x}^2+\dot{y}^2)-E=\frac{1}{2}(x^2+y^2)+\frac{1-\mu}{\sqrt{(x+\mu)^2+y^2}}+\frac{\mu}{\sqrt{(x-1+\mu)^2+y^2}}$$

On la part dreta de la igualtat és sempre positiva. Així, mentre la part esquerra sigui també positiva sempre hi haurà un conjunt de valors $[\,x\,,y\,,\dot{x}\,,\dot{y}\,]$ que compleixin la igualtat. Ara bé, si $E>\frac{1}{2}(\dot{x}^2+\dot{y}^2)$ el sistema deixa de tenir solució, fet que es reflecteix físicament amb la aparició de zones on no poden existir trajectòries amb certs valors de E: les regions de Hill.

En aquesta figura, l'àrea ombrejada indica l'espai que no podrà recórrer un cos en cas de no tenir energia suficient. El cas 1 representa el cas en què el cos té una energia inferior a la del punt de Lagrange L_1, en què només existeixen òrbites al voltant de m_1, m_2 o el conjunt m_1-m_2 per la part exterior. El cas 2 representa el cas d'un cos ab¡mb una energia inferior a L_2, on podran existir òrbites, a més, al voltant de m_1 i m_2 a la zona interior de l'àrea ombrejada. El cas 3 representa una energia inferior a L_3, el cas 4, una energia inferior a L_4 i L_5 i el cas 5 una energia superior a la de tots els punts de Lagrange, on poden existir òrbites que poden estar afectades per qualsevol punt de Lagrange i els cossos m_1 i m_2.

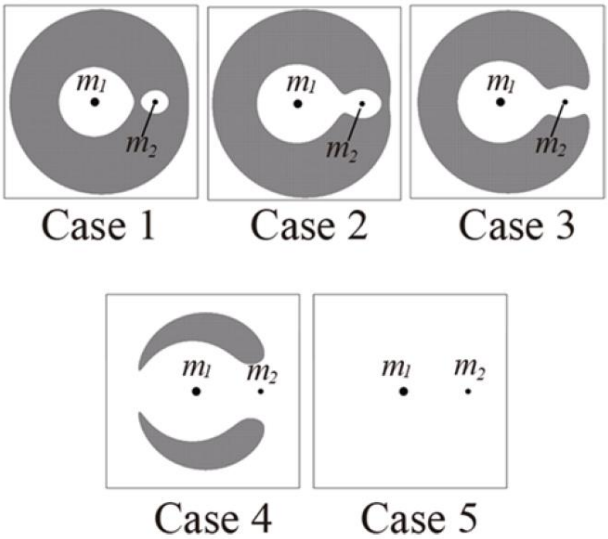


Figura 7: Restriccions de trajectòries segons els nivells d'energia. A major energia, major llibertat d'òrbites podrà tenir un cos.
FONT: [7]

Cal observar que un cos únicament podrà tenir una òrbita propera i al voltant d'un punt de Lagrange si l'energia d'aquest és superior a la del propi punt.

Com que els asteroides troians orbiten al voltant dels punts L_4 i L_5, l'energia de tots ells haurà de ser superior a l'energia d'aquests. En cas contrari, no tindria sentit parlar d'asteroides troians, sinó que es tractarien d'asteroides amb altres tipus d'òrbites més restringides. Cal dir, però, que no tots els asteroides que compleixin el requisit d'energia seran forçosament troians. Per tant, té sentit parlar d'aquests únicament dins un rang finit de valors d'energia.

Caldrà, a més, que els asteroides candidats a seguir òrbites troianes compleixin altres condicions de caràcter dinàmic. Aquestes es poden establir suposant condicions inicials dels cossos a estudiar pròximes als punts L_4 o L_5 (punts amb valor d'energia més alt).

Segons les simulacions realitzades, els rangs d'energia aproximats que poden donar lloc a asteroides troians es poden observar a la taula 2.

		Planeta		
		Terra	**Júpiter**	**Neptú**
Energia	**Valor mínim**	-1,499	-1,499	-1,499
	Valor màxim	-1,487	-1,478	-1,481

Taula 2: Valors extrems entre els quals té sentit parlar d'òrbites troianes per l'estudi de diferents planetes.

Una dada interessant és que el valor mínim de l'energia és lleugerament superior al dels punts L_4 i L_5. Això es deu a que per energies molt semblants a les d'aquests punts, les òrbites són difícils d'apreciar i es troben molt a prop dels punts d'equilibri.

2.5- Òrbites tadpole i horseshoe

En aquest apartat es descriuen les òrbites tadpole i horseshoe, a més d'explicar com aconseguir simular aquestes òrbites seleccionant uns valors inicials adequats.

2.5.1- Òrbites tadpole

Els asteroides troians són tots aquells cossos celestes que orbiten el Sol amb el mateix període orbital d'un planeta i es situen al voltant dels punts estables de Lagrange L_4 i L_5 del mateix sistema Sol-planeta seguint òrbites de tipus tadpole. També existeixen asteroides troians en sistemes planeta-lluna, tot i

que és més difícil. Al situar-se als punts de Lagrange, la dinàmica d'aquests asteroides pot ser estudiada amb el R3BP.

Fins ara, s'han descobert varis planetes que tenen asteroides troians orbitant amb ells. Aquests planetes són Júpiter, Neptú, Urani, Venus, la Terra i Mart. Entre ells, Júpiter és el que té més quantitat d'asteroides troians detectats, sent el seu nombre similar al nombre de cossos presents al cinturó d'asteroides del sistema solar. També cal destacar el cas de Neptú, pel qual s'han detectat 9 asteroides troians, tot i que es creu que el nombre d'asteroides amb un radi superior als 100 km podria superar en un ordre de magnitud a aquells pertanyents a Júpiter [5].

En el cas Sol-Júpiter, els asteroides que orbiten L_4 es troben dins l'anomenat camp grec i els que orbiten L_5 dins el camp troià.

Per simular aquest tipus d'òrbites, cal establir unes condicions inicials que situïn el cos molt proper als punts d'equilibri L_4 o L_5. En aquest sentit, una observació interessant de la figura 4 són les corbes de nivell que hi ha representades. Les més properes als punts d'equilibri estables segueixen una trajectòria que envolta únicament els mateixos punts, amb una forma de fava (en coordenades cartesianes). Aquest tipus de trajectòria és la que descriuen les òrbites tadpole.

A la figura 8 es pot veure la representació en coordenades cartesianes i polars de les dues òrbites de coordenades inicials $[r\,,\theta\,,\dot{\theta}\,,E\,] = [0.99\,,\ \pm 1.047\,,0\,,-1.494]$. Es pot apreciar fàcilment que les dues òrbites es troben al voltant dels punts de Lagrange L_4 i L_5.

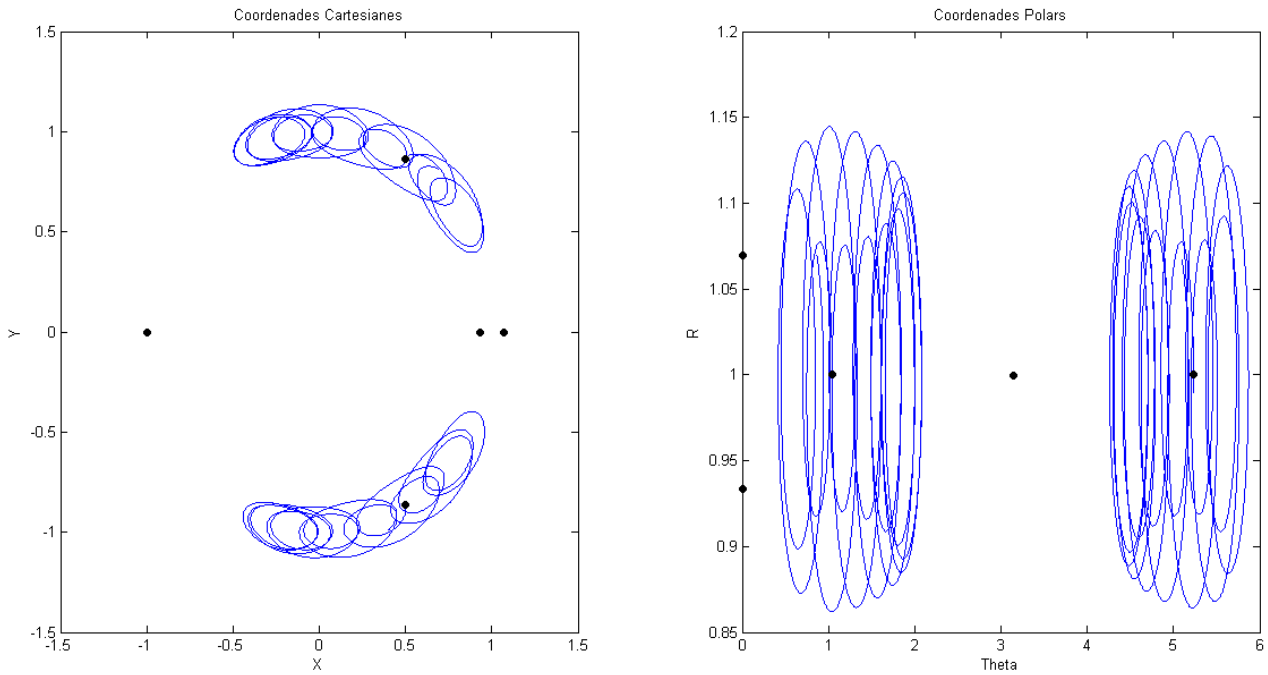


Figura 8: Representació de dues òrbites tadpole pel sistema Sol-Júpiter en coordenades cartesianes (esquerra) i polars (dreta) en color blau juntament amb els cinc punts de Lagrange representats de color negre. Les condicions inicials són $[r\,,\theta\,,\dot{\theta}\,,E\,] = [0.99\,,\ \pm 1.047\,,0\,,-1.494]$ *i el temps que s'ha calculat és de 83 unitats.*

2.5.2- Òrbites horseshoe

La segona classe d'òrbites que es tractaran en aquest treball són les òrbites horseshoe. Com el seu nom indica, tenen una forma de ferradura (en coordenades cartesianes). La trajectòria que segueix una horseshoe envolta els punts L_3, L_4 i L_5 [2]. A la figura 9 es pot apreciar la representació d'una òrbita horseshoe en coordenades cartesianes i polars amb coordenades inicials $[r, \theta, \dot{\theta}, E] = [0.983, \pi/2, 0, -1.494]$. Una distància inicial al punt L_4 major que la de la figura 8 provoca que el cos escapi de les proximitats del punt i es dirigeixi a L_3. Al sobrepassar aquest punt inestable, el cos continua allunyant-se de L_4 fins a trobar el punt estable L_5. Aquest és capaç d'invertir la tendència del cos i dirigir-lo novament cap L_3, el qual el dirigirà cap L_4 per acabar realitzant una òrbita periòdica.

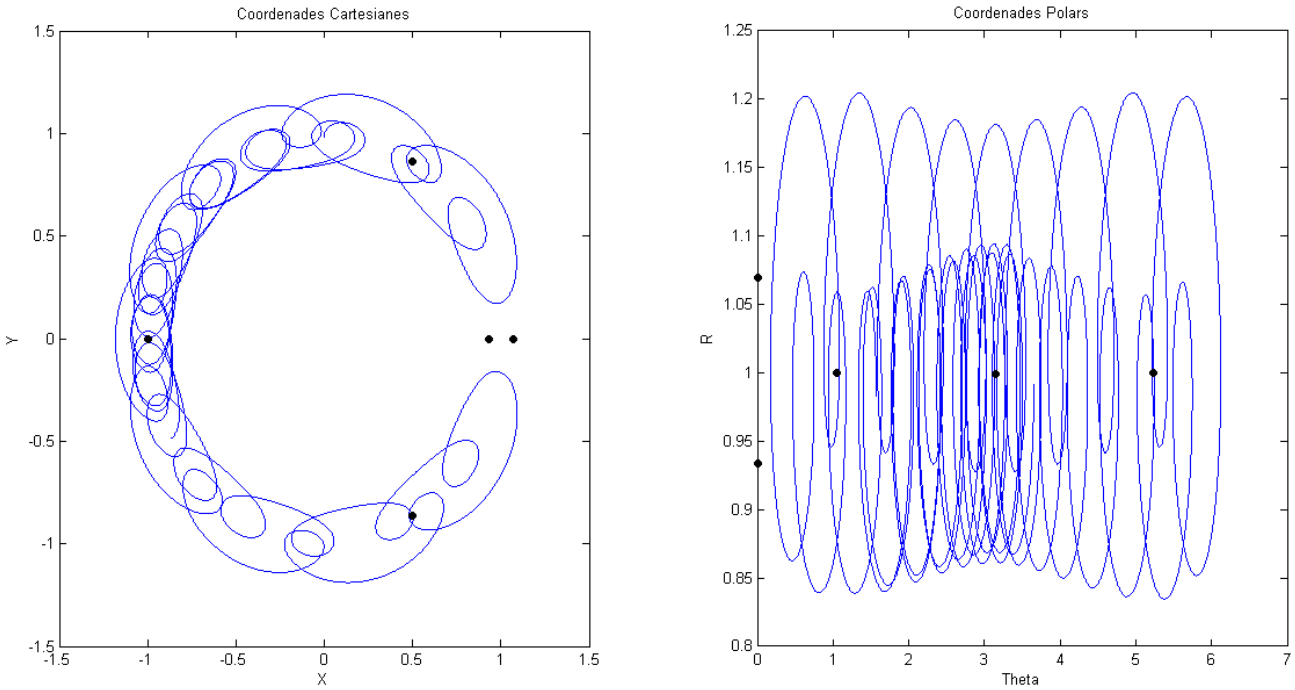


Figura 9: representació en coordenades cartesianes (esquerra) i polars (dreta) d'una òrbita horseshoe de condicions inicials $[r, \theta, \dot{\theta}, E] = [0.983, \pi/2, 0, -1.494]$ *i el temps calculat és de 200 unitats.*

2.5.3- Condicions inicials

La sel·lecció dels punts inicials a l'hora de calcular una òrbita és molt important. Així, a l'hora de calcular trajectòries tadpole cal prèndre condicions inicials x i y properes als punts d'equilibri estables, al voltant dels quals orbiten. Per altra banda, per obtenir òrbites horseshoe la sel·lecció de condicions x i y no és tan trivial. A més, es desconeixen els valors inicials adequats de $\dot{x}$ i $\dot{y}$ per cada x i y triades per tal d'aconseguir les òrbites desitjades.

En coordenades polars, els valors que pren la coordenada θ per les òrbites tadpole al voltant de L_4 es situa entre $\pi/6$ i $\pi/2$ aproximadament. Per tadpoles de L_5, aquests valors prenen signe negatiu, pel que varien entre $-\pi/6$ i $-\pi/2$. Per les òrbites horseshoe, θ pot variar entre $-\pi/6$ i $\pi/6$. Aquests valors són aproximats i poden variar entre diferents casos de cada òrbita. Per altra banda, el valor de r sempre es manté proper a la unitat, sent el cas de les òrbites tadpole més estricte que el de les horseshoe.

Així, per escollir unes condicions inicials per simular una òrbita tadpole en coordenades polars és més senzill. Les òrbites horseshoe i tadpole passen al voltant dels punts L_4 i/o L_5, pel que escollir un angle θ igual al d'un daquests punts i triar diferents valors de r propers a la unitat permet trobar-les. El mateix passa amb les horseshoe, tot i que per això caldrà prendre valors de r més allunyats de 1. En quant a la variable $\dot{\theta}$, és suficient amb suposar-la igual a zero. No és necessari obtenir un valor inicial per $\dot{r}$ ja que, gràcies a les relacions entre coordenades cartesianes i polars exposades a l'apartat 2.2, les condicions inicials del R3BP poden estar compostes per qualsevol grup de variables que permeti la obtenció de les variables cartesianes. Així, és possible especificar com a coordenades inicials $[r\ \theta\ \dot{\theta}\ E]$.

En aquest cas, un cop trobades les coordenades x i y, introduint l'equació 2.2.3 dins l'equació 2.1.4 s'aconsegueix el valor de $\dot{r}$ necessari per la obtenció de totes coordenades cartesianes. D'aquesta manera, a més, es pot especificar directament la energia que es vol estudiar de manera directe.

Cal recordar que la equació 2.1.4 és:

$$E = \frac{1}{2}(\dot{x}^2 + \dot{y}^2) - \frac{1}{2}(x^2 + y^2) - \frac{1-\mu}{\sqrt{(x+\mu)^2 + y^2}} - \frac{\mu}{\sqrt{(x-1+\mu)^2 + y^2}}$$

Introduïnt la relació 2.2.3 i prenent $\dot{\theta} = 0$ s'obté:

$$\frac{1}{2}(\dot{r}^2) = E + \frac{1}{2}(x^2 + y^2) + \frac{1-\mu}{\sqrt{(x+\mu)^2 + y^2}} + \frac{\mu}{\sqrt{(x-1+\mu)^2 + y^2}}$$

A model d'exemple, la solució obtinguda per les condicions polars inicials $[r\ \theta\ \dot{\theta}\ E] = [0.99\ ,1.047\ ,0\ ,-1.494]$ prenent com a valors inicials i finals de temps $[t_0\ t_{final}] = [0\ ,83]$ es troba representada en coordenades cartesianes a la figura 10, sent una òrbita tadpole.

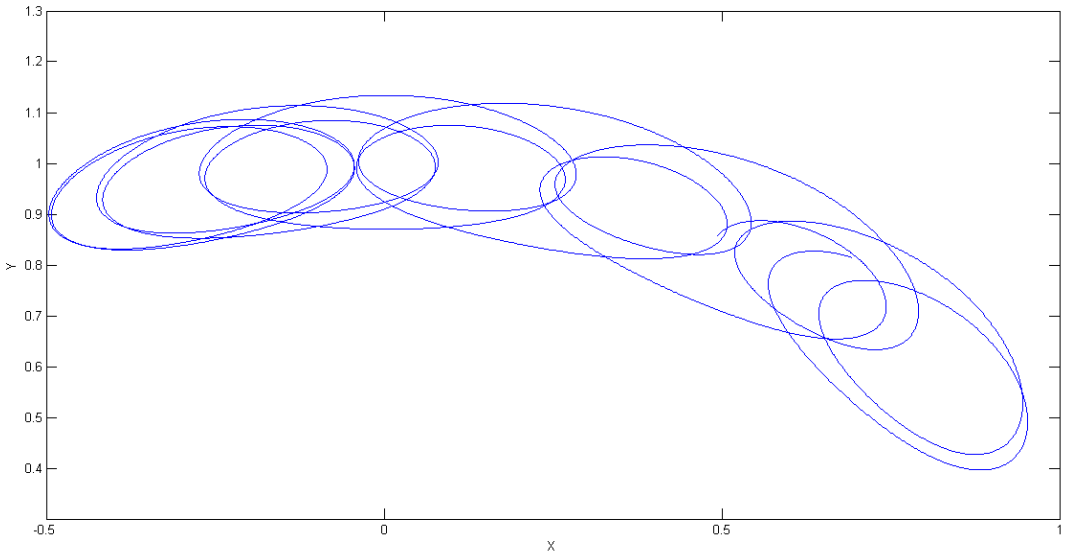


Figura 10: representació cartesiana de la resolució numèrica del R3BP per condicions inicials $[r\ \theta\ \dot{\theta}\ E] = [0.99\ ,1.047\ ,0\ ,-1.494]$ *per un temps d'integració de 83 unitats.*

2.6- Secció de Poincaré

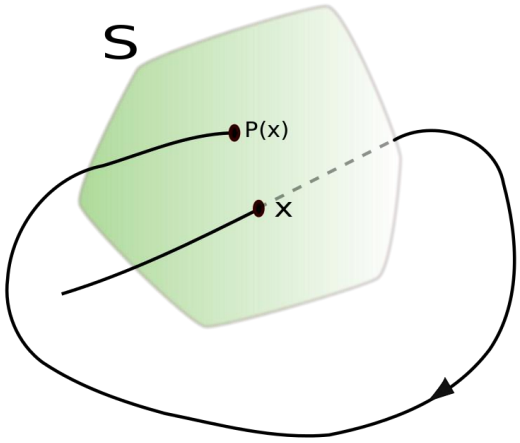


Figura 11: exemple de secció de Poincaré.

El matemàtic, físic, científic teòric i filòsof de la ciència Henri Poincaré va revolucionar l'estudi de la mecànica celeste l'any 1899 amb la publicació de *Méthodes Nouvelles*. [3]

Una secció de Poincaré és una aplicació en la que donat un punt sobre una secció calculada, calcula el següent punt d'una òrbita que torna a estar sobre la mateixa secció [6]. Un exemple entenedor és el que es mostra a la figura 11. En aquesta es pot observar una secció S que és atravessada dues vegades per una òrbita determinada (línia negre). Així, una aplicació d'aquesta òrbita sobre S calcula el punt x a partir del punt P(x).

La resolució de les equacions que componen el R3BP obté els valors $[x\ ,y\ ,\dot{x}\ ,\dot{y}]$ per diferents instants de temps. A més, es poden obtenir a partir d'aquestes dades els valors $[r\ ,\theta\ ,\dot{r}\ ,\dot{\theta}]$. Per la representació d'òrbites tadpole i horseshoe dins el R3BP és possible utilitzar una aplicació que dóna com a resposta tots aquells instants de temps, junt amb les seves coordenades cartesianes i polars, on es satisfà la igualtat $\dot{\theta} = 0$. Això és possible ja que totes dues òrbites presenten bucles on aquesta variable oscil·la entre valors positius i negatius. Així, per poder visualitzar una òrbita no és necessari

dibuixar-la tota mitjançant una línia, sinó que serà suficient dibuixant una sèrie de punts, que es corresponen a aquells que atravessen la secció $\dot{\theta} = 0$. D'aquesta manera s'aconsegueix eliminar molta informació de poca rellevància dels gràfics, donant la possibilitat de representar una gran quantitat d'òrbites diferents en una sola figura.

Com a exemple, es prenen les mateixes condicions inicials de la figura 10, se'n calculen les seccions de Poincaré i es dibuixen sobre l'òrbita completa. Els resultats es mostren a la figura 12.

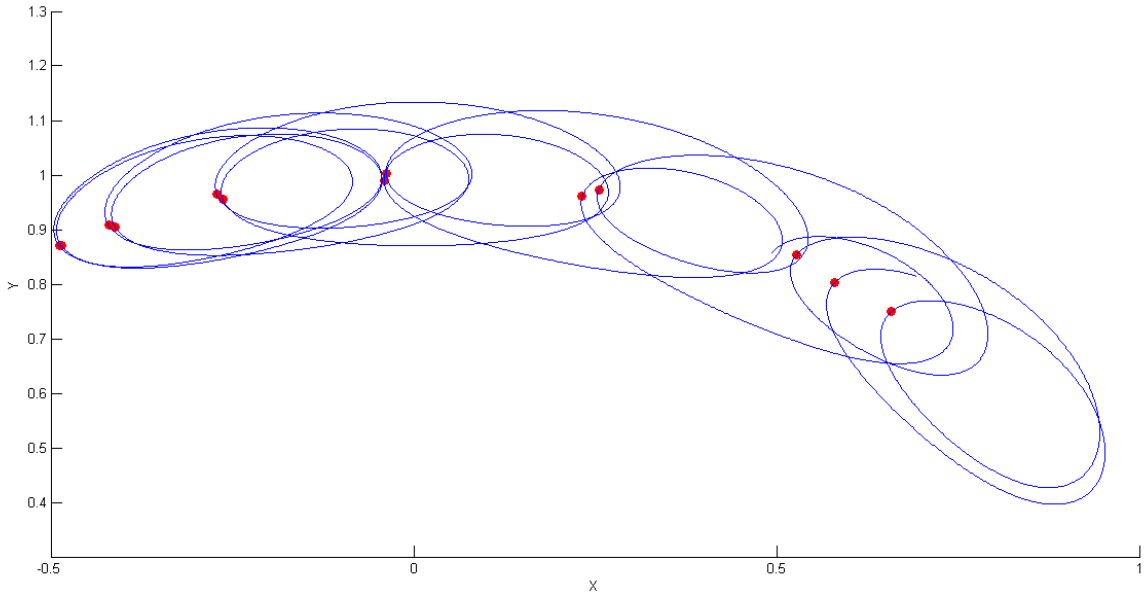


Figura 12: Seccions de Poincaré (en vermell) i òrbita completa (en blau) per les mateixes condicions inicials que a la figura 7.

Com es pot veure, les seccions de Poincaré coincideixen amb el costat esquerre de cada bucle que fa l'òrbita. Cal dir que només s'han calculat la meitat de les seccions de Poincaré ja que pel costat dret de cada bucle també es compleix $\dot{\theta} = 0$. L'objectiu de no calcular tots els punts és que per cada bucle només aparegui una secció de Poincaré. Matemàticament, això s'aconsegueix introduint una nova condició per decidir si s'ha trobat una secció de Poincaré desitjada o no. Aquesta condició és imposant que $\dot{r} > 0$.

Si s'elimina l'òrbita completa de la representació gràfica i es deixen únicament les seccions de Poincaré, el gràfic queda molt més simplificat, alhora que es segueix tenint informació sobre quina trajectòria general segueix l'asteroide.

A la figura 13 es pot comprovar que, efectivament, les seccions de Poincaré coincideixen amb $\dot{\theta} = 0$ i $\dot{r} > 0$.

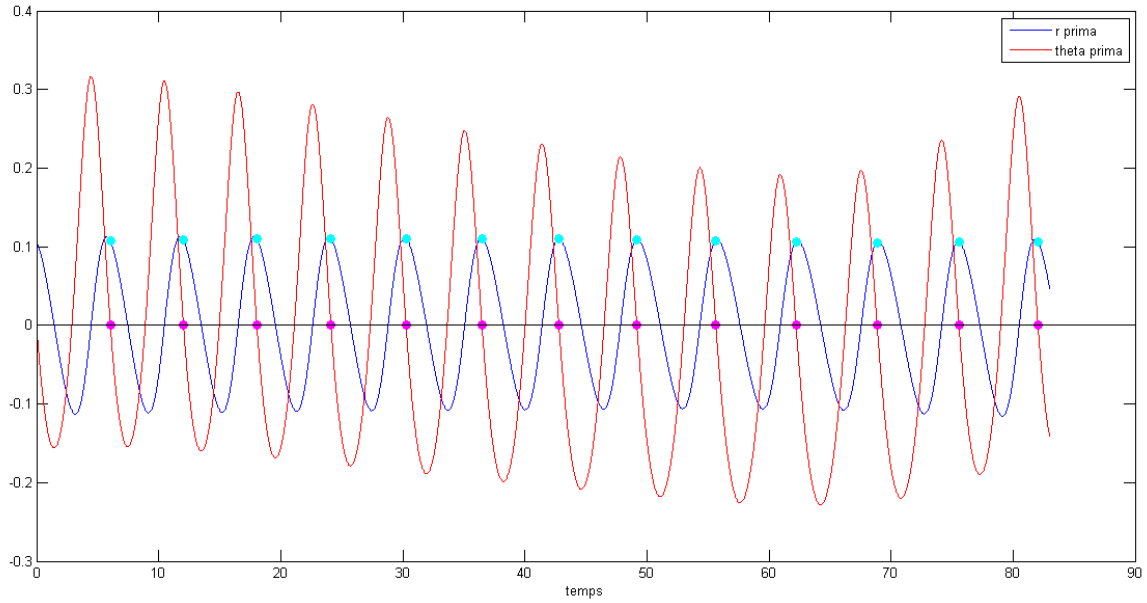


Figura 13: Representació de la evolució de $\dot{r}$ (blau) i $\dot{\theta}$(vermell) juntament amb els valors de les seccions de Poincaré (punts de color cian i magenta respectivament).

A les figures 14 i 15 es pot veure com queden les trajectòries de les figures 8 i 9 respectivament, dibuixant únicament els punts obtinguts per la secció de Poincaré. Es pot observar clarament que la secció de Poincaré permet obtenir unes figures molt més clares i senzilles. A més, a les representacions polars s'ha pogut ajustar l'eix vertical per obtenir una visió més detallada dels punts, deixant així fora del gràfic els punts L_1 i L_2, que són de poca rellevància en aquest treball.

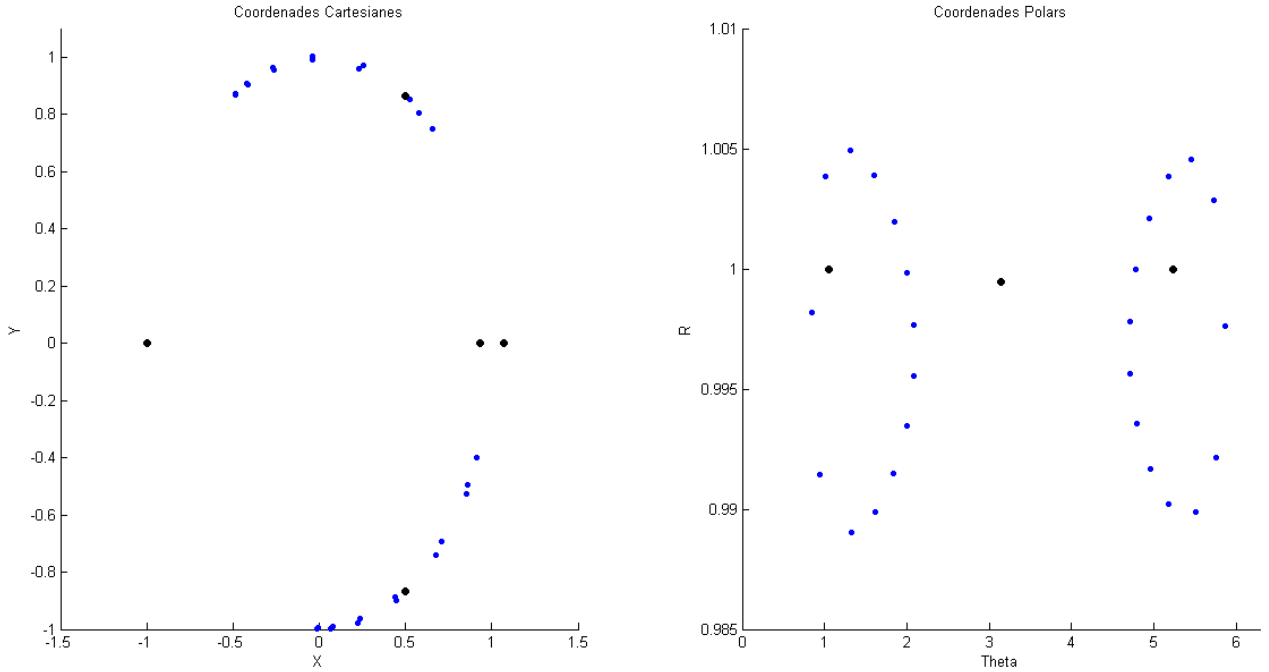


Figura 14: representació de les seccions de Poincaré corresponents a la figura 8, corresponents a dues òrbites tadpole al sistema Sol-Júpiter.

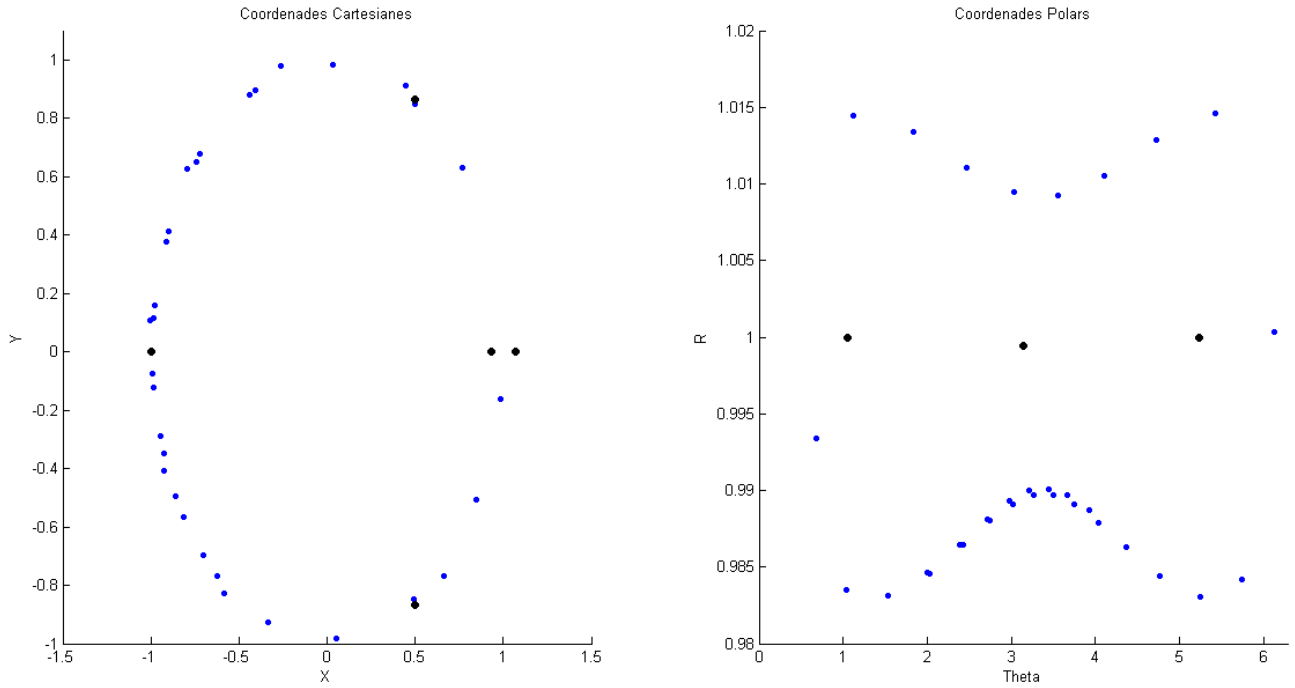


Figura 15: representació de les seccions de Poincaré corresponents a la figura 9, corresponents a una òrbita horseshoe al sistema Sol-Júpiter.

3- Metodologia

En aquest capítol s'exposen de manera concisa tots els aspectes teòrics rellevants i els conceptes metodològics aplicats en aquest treball.

3.1- Resolució d'equacions diferencials

Donat un sistema d'equacions diferencials $X = F'(X)$ on X és un vector de dimensió n, aquest pot o no tenir solució analítica [6].

En cas de no tenir-ne, és impossible determinar-ne les solucions explícites sinó que únicament es pot trobar una solució determinada a partir d'una condició inicial $X(t_0) = X_0$, de manera que per aquest vector de condicions inicials existeix una única sol·lució. Això es coneix com a problema de valor inicial.

El R3BP no té solució analítica, de manera que especificant un conjunt de condicions inicials es pot calcular numèricament la solució del sistema. Aquesta solució representa una òrbita concreta.

El mètode numèric que s'ha emprat per solucionar el R3BP a partir d'unes determinades condicions inicials s'exposa a l'apartat següent.

3.2- Mètode d'integració numèrica: Runge-Kutta-Fehlberg.

En aquest treball s'ha escollit el mètode numèric Runge Kutta Fehlberg d'ordre 7 i 8 per resoldre el R3BP. Aquest mètode es basa en els mètodes Runge-Kutta però introduint una característica important: el pas d'integració adaptatiu.

El mètode RKF78 obté la solució de les equacions diferencials per petits increments de temps utilitzant el mètode RK7 i prenent de referència per assegurar una certa precisió en els resultats el mètode RK8, calculant-ne l'error relatiu entre un i altre.

En cas que les solucions del RK7 i el RK8 divergeixin en excés, superant l'error màxim que es vulgui garantir, es torna a realitzar la integració amb un increment de temps menor. Per altra banda, si l'error relatiu és molt inferior a l'error relatiu màxim, s'incrementa el pas d'integració per tal de garantir una certa celeritat. Es defineixen, per tant, dos límits de l'error relatiu, un superior o màxim i un inferior o mínim.

Cal dir què tots els càlculs s'han fet amb el software MATLAB, treballant entorn a una funció anomenada *ode78*, dels autors *Daljeet Singh* i *Howard Wilson*, modificat per *Marc Compere* sota una llicència pública general de GNU. Amb aquest programa, s'ha especificat un pas d'integració màxim de 10^{-2} unitats i un error relatiu màxim de 10^{-9}.

3.3- Càlcul de la secció de Poincaré.

Com ja s'ha dit, les seccions de Poincaré rellevants per obtenir una correcta visualització dels resultats en aquest treball es correspon a aquells punts on la variable $\dot{\theta}$, en coordenades polars, es fa zero.

A continuació s'explica, a grans trets, el mètode emprat per trobar aquests punts en el programa MATLAB:

a) Prenent unes determinades condicions inicials, es resol el R3BP numèricament per trams de temps petits (de l'ordre de $10 \cdot pas\ d'integració\ màxim$) i consecutius, obtenint alhora les coordenades polars corresponents a cada punt obtingut durant les integracions.

b) Per cada tram de temps, es comprova el signe de $\dot{\theta}$ en el primer i l'últim punt i s'aplica el teorema de Bolzano. S'assumeix que, degut a que els trams de temps són petits, les variacions que poden haver-hi a $\dot{\theta}$ són prou petites per assegurar que, com a molt, prendrà valor 0 una vegada.
   - En cas que no hi hagi variació de signe significa que la variable no ha pres valor 0 i es continua resolent pel següent tram temporal.
   - En cas que el primer i últims valors de $\dot{\theta}$ siguin diferents indica que dins aquell tram $\dot{\theta}$ ha pres valor 0. Es passa al pas c).

c) Mitjançant el mètode de la bisecció es troba l'instant de temps i les coordenades cartesianes i polars en que $\dot{\theta} = 0$ amb una precisió de 10^{-10}. Prenent els valors inicials els de l'últim punt de l'últim tram de temps calculat, es torna al pas a).

3.4- Representació de les variables polars.

Una manera de representar les dades molt utilitzada durant l'elaboració del treball és emprant les variables polars θ per l'eix d'ordenades i r per l'eix de coordenades. Un avantatge d'aquesta representació és que algunes òrbites periòdiques o quasi-periòdiques que en coordenades cartesianes poden semblar de vaivé i dificultar la visualització de totes les seccions de Poincaré, en coordenades

polars es pot distingir clarament que no és així i les seccions queden ben espaiades. Això es pot apreciar amb claredat a la figura 16, on es representen les seccions de la aplicació de Poincaré de la mateixa òrbita que a la figura 10 tant en coordenades cartesianes com en polars. Cal dir que la representació en coordenades cartesianes pot portar a confusió a l'hora de discernir alguns punts molt propers entre ells de la secció de Poincaré mentre que les coordenades polars n'ofereixen una representació més clara.

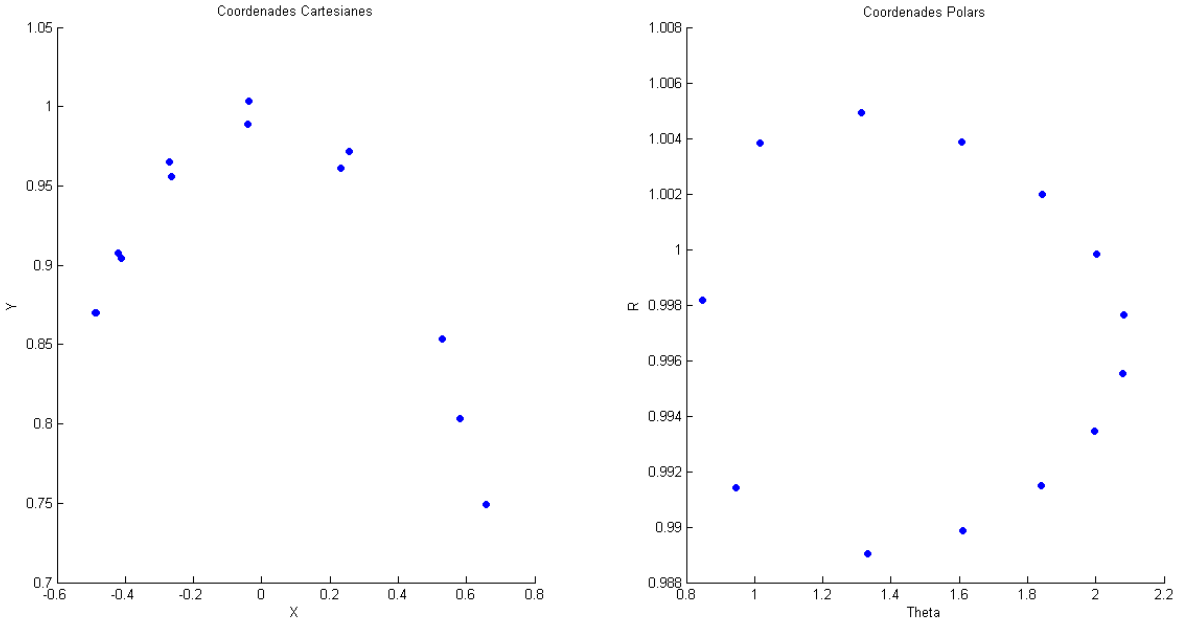


Figura 16: Comparació de la aplicació de Poincaré entre la representació cartesiana (esquerra) i polar (dreta) de la mateixa òrbita de la figura 10.

En la resolució d'aquest treball s'ha emprat únicament la secció de Poincaré i les coordenades polars per a la representació d'òrbites. Així, es pretén obtenir major claredat en les figures on hi hagi representades una gran quantitat d'òrbites diferents.

3.5- Processament i visualització de dades.

La aplicació de Poincaré és molt útil per veure quina és la tendència de les variables eliminant dels gràfics l'excés d'informació no desitjada, pel que es facilita molt la visualització d'òrbites de múltiples asteroides (és a dir, les òrbites que segueixen múltiples condicions inicials). Tot i així, es pot donar el cas que la quantitat d'òrbites a visualitzar és tan elevada i les seccions de Poincaré són tan properes entre elles, que es fa necessari l'ús des tècniques complementàries per visualitzar correctament les òrbites.

L'eina escollida en aquest treball per resoldre el problema de l'excés d'informació és el de mapa de color. Per realitzar un mapa de color es divideix l'espai on es troben les seccions de Poincaré a dibuixar

en rectangles regulars. Seguidament, es calcula la quantitat de seccions de Poincaré que pertoquen a cada rectangle segons les seves coordenades. D'aquesta manera s'obté un histograma en dues dimensions que representa la densitat de seccions de Poincaré, és a dir, el nombre de seccions per unitat de superfície.

La representació d'aquest histograma es realitza en un mapa de color pla, que associa un color a cada nivell de densitat.

Una altra utilitat a l'hora de dibuixar vàries òrbites en el mateix gràfic és emprant diferents colors per cada òrbita. Així, en cas que hi hagi òrbites que es creuin o tinguin parts del seu recorregut similars, sempre es podrà identificar la trajectòria de cada òrbita.

La figura 17 representa les seccions de Poincaré de 5 òrbites amb condicions inicials properes en diferents colors. D'aquesta manera es pot distingir perfectament quins punts es corresponen a cada òrbita.

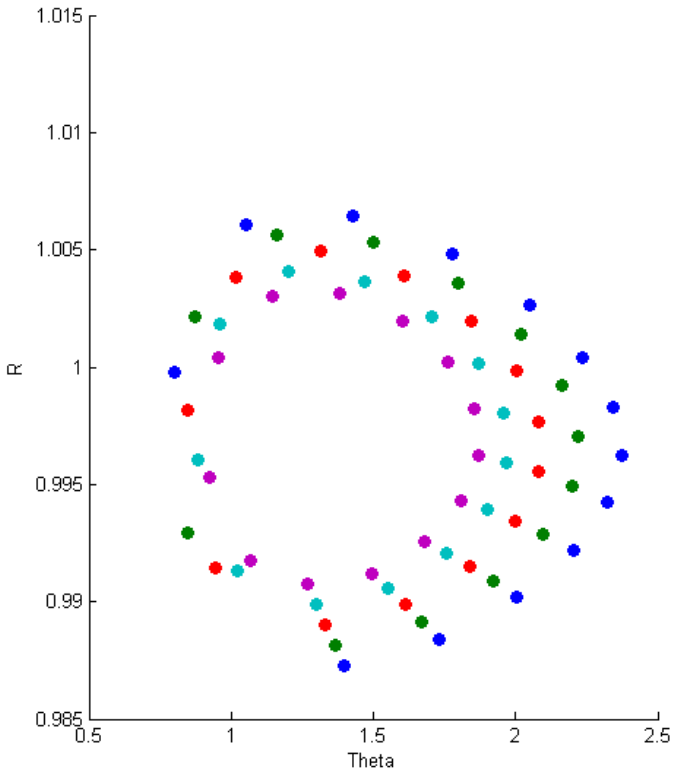


Figura 17: Representació de 5 òrbites tadpole properes en diferents colors.

4- Jumping Trojans

Hi ha asteroides que segueixen òrbites tadpole, però estudiant la seva trajectòria futura o passada, s'observa que en algun moment canvien el seu comportament i salten d'un camp a l'altre, recorrent una òrbita de tipus horseshoe. Per tant, es podria dir que segueixen òrbites tadpole quasi estables al voltant d'un punt d'equilibri fins que aconsegueixen marxar, convertint-se així en horseshoe únicament per tornar-se a convertir en tadpoles un cop arriben a l'altre punt estable.

Aconseguir trobar les condicions inicials propícies per simular aquest tipus d'asteroide requereix un anàlisi més profund de la mecànica rere els asteroides troians. Si s'uneixen les representacions polars de les figures 14 i 15 s'obté un gràfic on queden representades dues òrbites tadpole (una per cada punt de Lagrange estable) juntament amb una òrbita horseshoe. El resultat es veu a la figura 18.

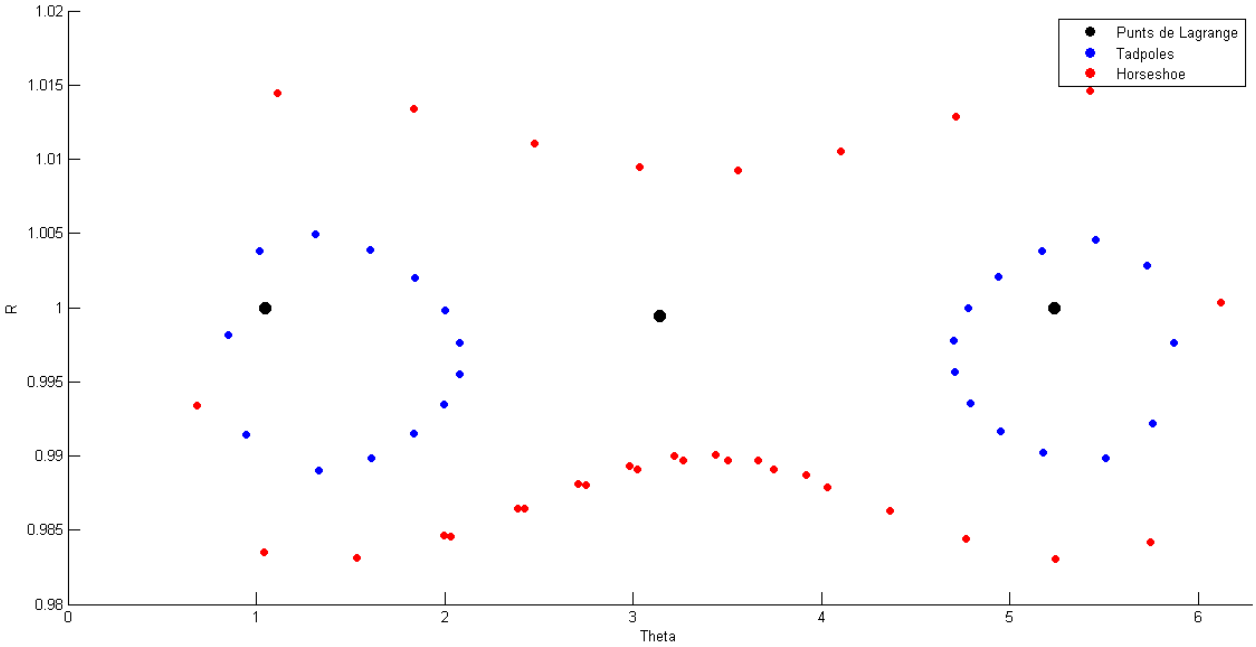


Figura 18: representació de dues òrbites tadpole (blau) i horseshoe (vermell) corresponents a les figures 13 i a4 juntament amb els punts de Lagrange L_3, L_4 i L_5.

S'observa com l'òrbita horseshoe (en vermell) envolta tant els tres punts de Lagrange representats (d'esquerra a dreta: L_4, L_3 i L_5) com les òrbites tadpole (en blau), que envolten cada una un punt estable. Així, a l'hora de buscar *jumping trojans* cal estudiar el que passa entremig d'aquestes dues òrbites. Per aquest motiu cal veure com és aquesta zona de transició entre tadpoles i horseshoes.

Dinàmicament, en aquesta zona intermèdia s'hi troben un conjunt d'òrbites que estan associades a òrbites periòdiques al voltant de L_3 i que en representen les varietats invariants [7]. Aquestes queden fora de l'abast d'aquest treball, que pretén avaluar si és possible estimar les probabilitats de trobar jumping trojans emprant únicament la secció de Poincaré. En cas de ser possible, seria una possible

comprovació prèvia de si cal calcular les varietats invariants per realitzar un estudi en profunditat o no.

A la figura 19 s'observa la simulació de dues varietats invariants anomenades estables (de color blau) i dues inestables (de color vermell), realitzat a l'article esmentat, pel sistema Sol-Júpiter.

En les interseccions entre les varietats estables i inestables es poden obtenir punts inicials que formin part d'òrbites troianes.

D'aquesta imatge es pot deduir, a més, que les òrbites tadpole estan contingudes al voltant dels punts L_4 i L_5 sense intersecar amb les varietats invariants. Les òrbites horseshoe, per altra banda, envolten tots dos punts sense entrar dins els lòbuls formats entre les varietats estables i inestables.

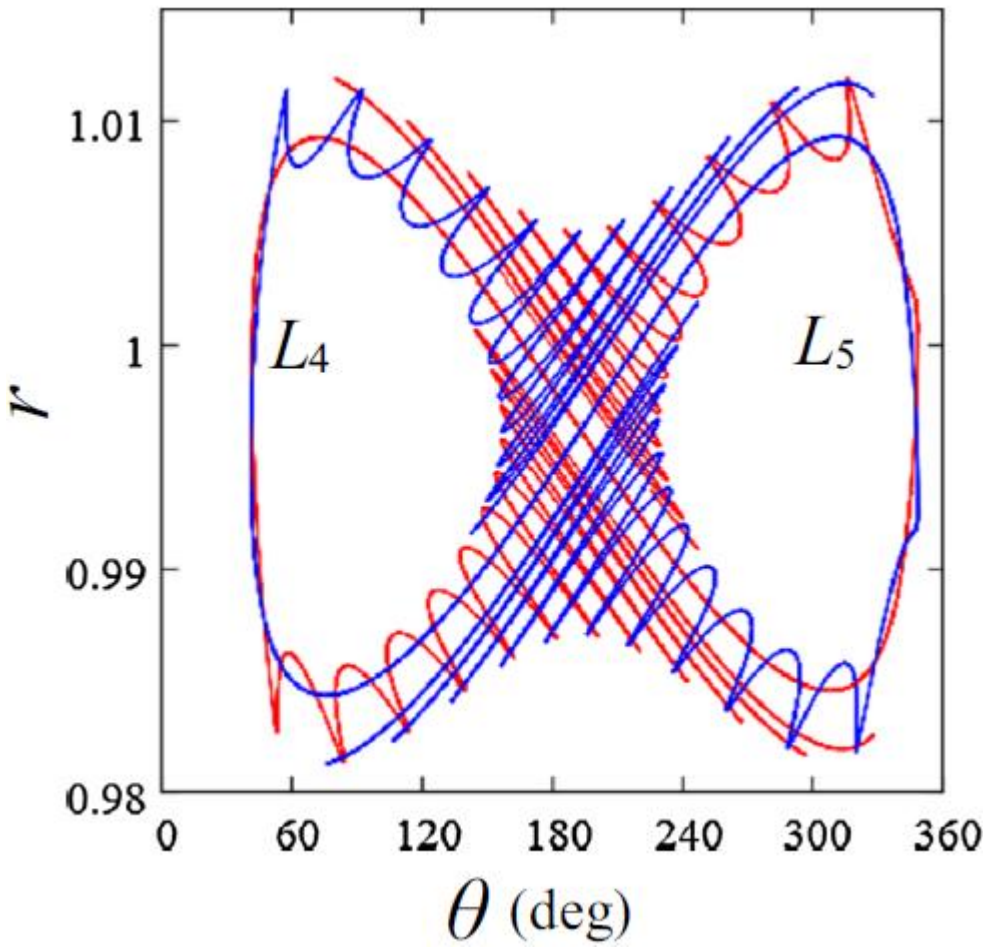


Figura 19: projeccions de dues varietats estables (blau) i dues inestables (vermell) de l'òrbita de Lyapunov del sistema Sol-Júpiter. FONT: [7]

5- Resultats

En aquest capítol s'exposen les simulacions realitzades dels asteroides troians dels diferents planetes estudiats. Per tots els casos s'ha estudiat un total de 1001 condicions inicials. Aquestes condicions s'han obtingut variant únicament la variable r entre un valor mínim i un valor màxim diferents per cada planeta. Per altra banda, les altres condicions inicials $(\theta\,,\dot{\theta}\,,E)$ han pres els següents valors:

- $\theta = pi/2$ coincidint amb el seu valor pel punt estable L_4. D'aquesta manera s'aconsegueix garantir que dins els resultats hi haurà tant òrbites tadpole com horseshoe depenent únicament del valor de r.
- $\dot{\theta} = 0$ per començar des d'una secció de Poincaré. D'aquesta manera s'assegura que no es tria un valor massa gran o massa petit que podria corrompre el comportament esperat de les òrbites
- E pren diferents valors en funció del planeta a estudiar i ha d'estar dins els marges especificats a la taula 2, a l'apartat 2.4. Així, els diferents valors de E escollits són:
    - $E_{1\,,Júpiter} = -1.494$ $\quad E_{2\,,Júpiter} = -1.49$
    - $E_{Terra} = -1.49$
    - $E_{Neptú} = -1.49$

D'aquesta manera, és d'esperar que apareguin representades òrbites tadpole del punt L_4 i horseshoe. Les òrbites tadpole de L_5 no apareixeran ja que no es calculen condicions inicials per aquest punt d'equilibri, tot i que no caldria ja que el R3BP és simètric respecte l'eix x, pel que les òrbites tadpole de L_4 i L_5 són també simètriques. A més, així s'aconsegueix visualitzar les òrbites horseshoe al voltant de L_5 sense dificultat per discernir-les, en principi, d'òrbites tadpole.

Les imatges més complexes d'aquest capítol es poden trobar ampliades a l'annex corresponent.

5.1- Sol-Júpiter

Per el cas Sol-Júpiter $_1$, s'han estudiat el rang de valors $r = [0.98\,,1.02]$. El temps d'integració ha estat de 1000 (el que representa el temps en què Júpiter dóna aproximadament 159 voltes al voltant del Sol). El gràfic resultant és un mapa de color de 1000x1000 caselles representant totes les seccions de Poincaré obtingudes i s'observa a la figura 20. Es pot veure perfectament com els punts L_3, L_4 i L_5

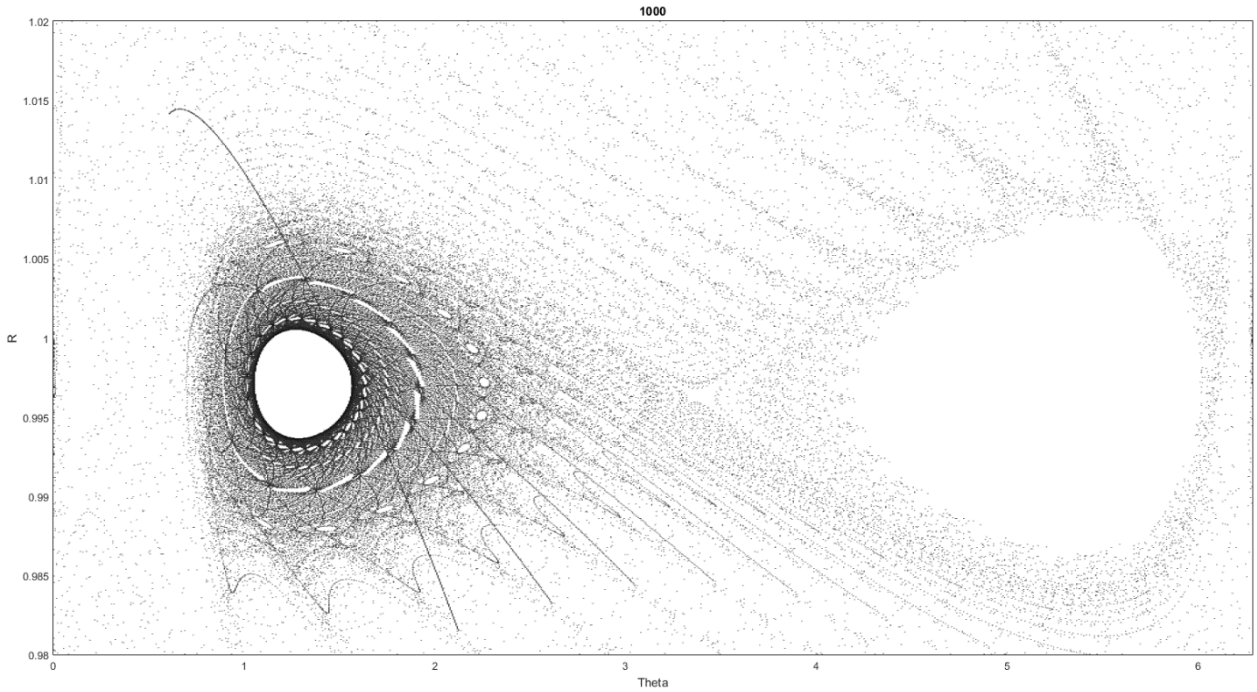


Figura 20: Mapa de color de 1001 condicions inicials del sistema Sol-Júpiter $_1$ integrades numèricament per t=1000. E=-1.494.

queden envoltats per multitud d'òrbites. Així, hi ha un conjunt d'òrbites tadpole confinades al voltant de L_4 i òrbites horseshoe que envolten tant L_4 com L_5. També es poden intuir les varietats invariants de les òrbites periòdiques de L_3.

A més a més, es pot veure com es creen una espècie de línies de tendència de trajectòries que uneixen els punts L_4 i L_5. Els asteroides presents en aquestes zones emanen de les proximitats de L_4 per la part inferior de la imatge i s'uneixen a les proximitats de L_5. A partir d'aquí es veu com apareixen un seguit d'òrbites tadpole al voltant de L_5. Per altra banda, també apareixen aquestes trajectòries emanant de L_5 per la part superior de la imatge i unint-se per L_4. S'ha comprovat que les seccions de Poincaré dels asteroides troians avancen en sentit antihorari. Per tant, per la part inferior els asteroides tendeixen a la dreta i per la part superior tendeixen a l'esquerra.

Concretament, les zones comentades coincideixen amb les varietats inestables de l'òrbita de Lyapunov al voltant de L_3. Els asteroides que segueixen aquestes trajectòries s'han hagut de trobar, en primera instància, entre les ondulacions creades per les varietats inestables. No tots aquests asteroides són potencialment *jumping trojans* ja que, per poder-ho ser, han d'haver estat alhora dins un lòbul entre les varietats inestables i estables.

Escollint un punt inicial dins la zona de *jumping trojans* inferior de la figura 20, concretament $[r\,,\theta\,,\dot{\theta}\,,E\,] = [0.991955\,,\ 3.326894\,,0\,,-1.494]$, es calculen les seccions de Poincaré per un període de t=200 tant per temps futurs com passats. Els resultats es poden observar a la figura 21.

En aquesta figura es pot observar clarament com un punt inicial proper a L_3 prové d'una òrbita tadpole al voltant de L_4 i s'acaba convertint en una tadpole de L_5.

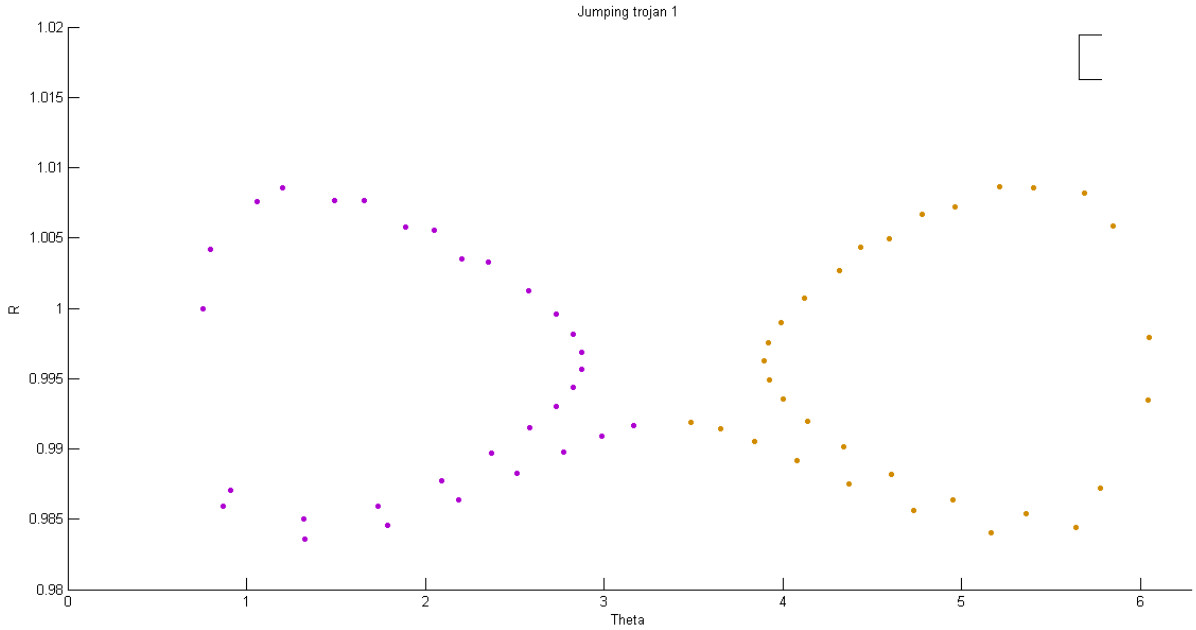


Figura 21: representació de la transició de L_5 a L_4 d'un jumping trojan per 200 unitats de temps futur (groc) i 200 passat (lila)

Per altra banda, si s'escull un punt de la zona de *jumping trojans* de la part superior del gràfic, concretament $[r\,,\theta\,,\dot{\theta}\,,E\,] = [1.00173\,,\ 3.43498\,,0\,,-1.494]$ per els mateixos períodes de temps, s'obtenen els resultats de la figura 22 en que una tadpole de L_5 es converteix en una tadpole de L_4.

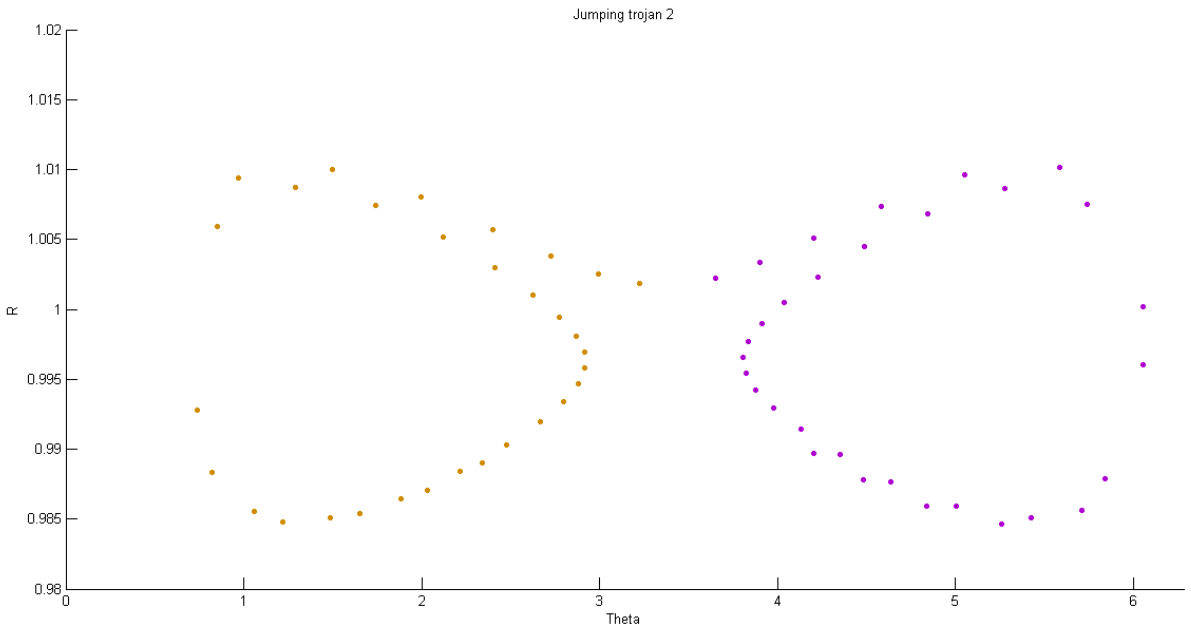


Figura 22: representació de la transició de L_4 a L_5 d'un jumping trojan per 200 unitats de temps futur (groc) i 200 passat (lila)

Unint ara les figures 21 i 22 sobre la representació gràfica directe de totes les trajectòries calculades pel sistema Sol-Júpiter s'obté la figura 23.

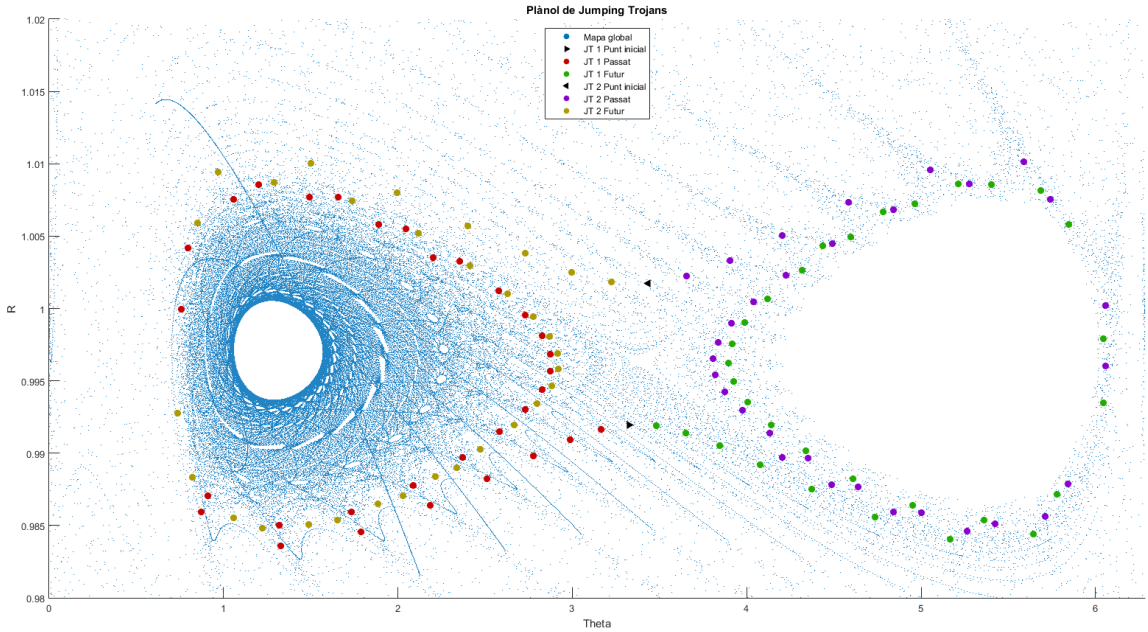


Figura 23: representació de dos punts inicials de jumping trojans. Els punts inicials estan representats en negre. Els colors rosa i groc corresponen a integracions temps enrere i els colors vermell i verd, temps futurs.

En aquesta figura es pot observar els punts inicials seleccionats (fletxes en negre) per realitzar les simulacions de dos *jumping trojans* i les trajectòries que segueixen, de la mateixa manera que a les figures 21 i 22. A més, es pot apreciar com les trajectòries es mantenen sempre entre els plecs que fan les varietats invariants que separen les òrbites tadpole de les horseshoe.

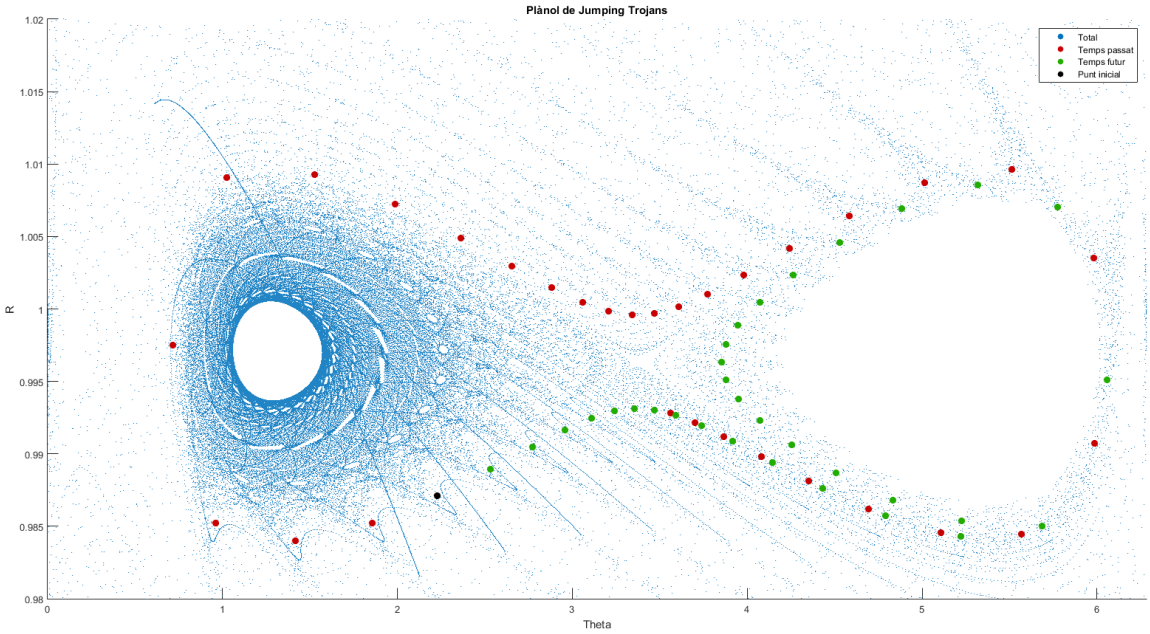


Figura 24: Representació de les òrbites futures i passades un temps d'integració de 200. El punt inicial està representat en negre.

Per altra banda, a la figura 24 es pot veure la propagació per temps de 200 unitats en el futur i en el passat del punt inicial $[r\,,\theta\,,\dot{\theta}\,,E\,] = [0.9871\,,\ 2.229\,,0\,,-1.494]$. Es pot observar com, a partir d'un punt inicial dins un plec de la varietat invariant, una òrbita horseshoe acaba convertint-se en tadpole de L_5. Un altre aspecte interessant és que, per temps passats, es pot comprovar que l'asteroide no ha adoptat una òrbita tadpole en un sol trajecte (mitja òrbita horseshoe) de L_4 a L_5 sinó que ha actuat com a horseshoe completant, com a mínim, una òrbita sencera. En aquest cas no es tracta pròpiament d'un *jumping trojan*, sinó d'un asteroide en òrbita horseshoe que ha adoptat una òrbita tadpole, a diferència d'un *jumping trojan,* on un asteroide en òrbita tadpole salta d'un camp a un altre seguint una òrbita horseshoe.

A la figura 25 es troben representades les òrbites 200 unitats de temps futures i passades pel punt inicial $[r\,,\theta\,,\dot{\theta}\,,E\,] = [0.9848\,,\ 2.407\,,0\,,-1.494]$. Aquest punt és molt proper al punt inicial de la figura 19 amb la diferència que es troba fora del plec de la varietat invariant.

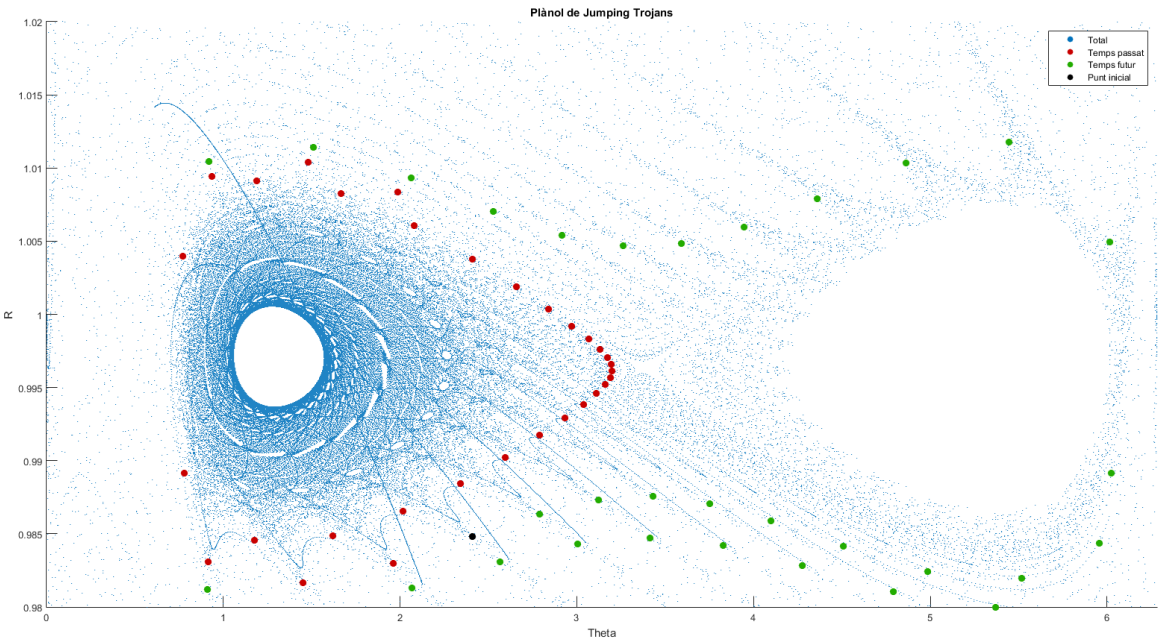


Figura 25: Representació de les òrbites futures i passades un temps d'integració de 200. El punt inicial està representat en negre.

En aquest cas, al contrari del que passa a la figura 24, l'asteroide passa de tenir una òrbita tadpole de L_4 a ser una horseshoe.

La figura 26 representa una part de la figura 20 en la que s'observa la part més propera a L_3 de les òrbites tadpole de L_5. L_3 es troba a la zona buida de seccions de Poincaré de l'esquerra de la figura.

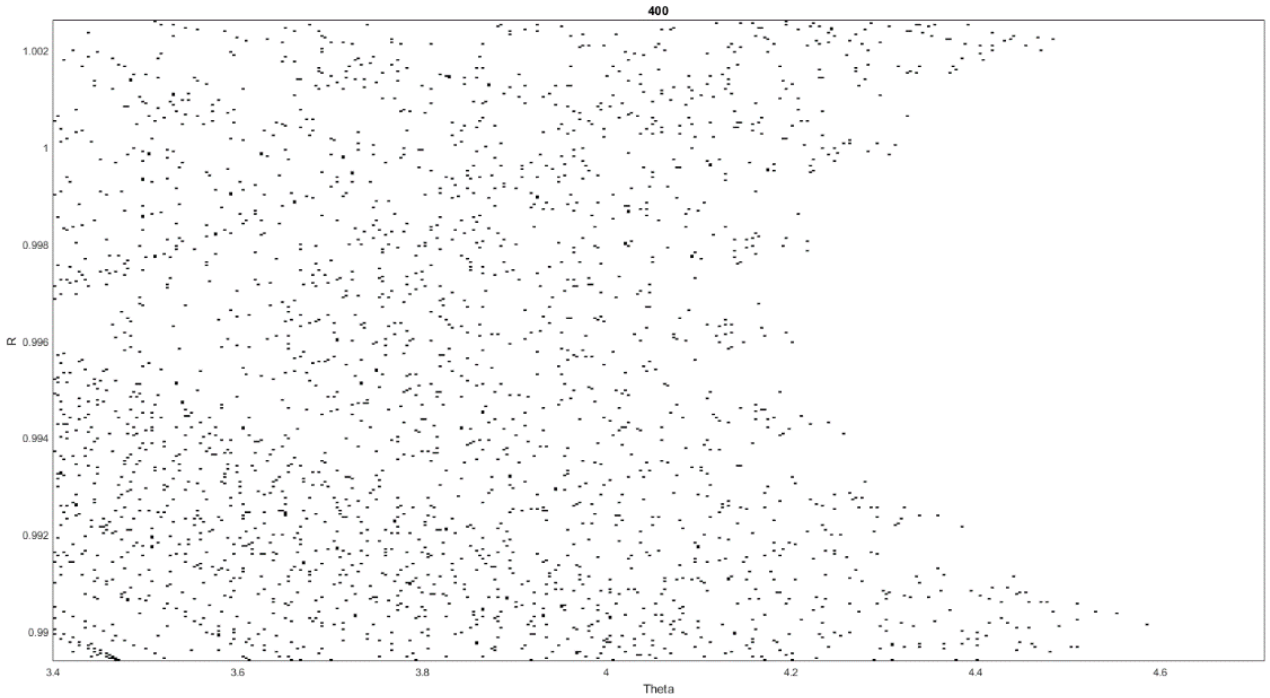


Figura 26: Detall de la part més propera a L_3 de les òrbites tadpole de L_5 de la figura 20.

A la figura 27 es pot observar la representació d'una vuitena part de les òrbites calculades. En ella, cada òrbita s'ha dibuixat en un color concret (hi ha òrbites que repeteixen els colors ja que n'hi ha una gran quantitat). S'ha realitzat aquesta figura per poder realitzar observacions generals d'òrbites concretes sense necessitat de conèixer-ne les coordenades inicials. Així, s'hi poden realitzar un seguit

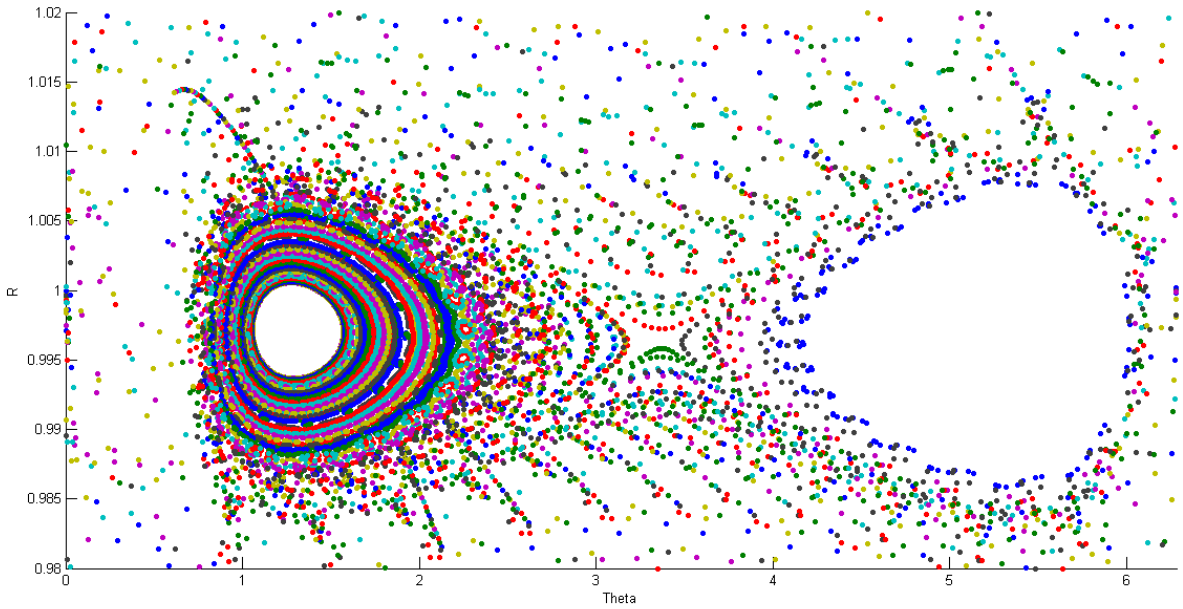


Figura 27: Coloració d'una vuitena part del total d'òrbites calculades pel sistema Sol-Júpiter per E=-1.494.

d'observacions interessants. Tenint en compte que totes les òrbites representades tenen el seu punt inicicial a prop de L_4, existeixen un seguit d'òrbites de color blau, entre altres, que prenen un comportament de tadpole al voltant de L_5. Aquestes òrbites semblen provenir de zones lleugerament per fora de les òrbites tadpole de L_4, tal i com era d'esperar. També es poden discernir unes òrbites colorades de vermell, verd i blau clar passant molt a prop de L_3 traçant òrbites horseshoe.

El segon estudi realitzat, Sol-Júpiter $_2$, ha pres valors inicials de $r = [0.975\ , 1.015]$. Per aquest cas, el mapa de color de 900x900 resultant es veu a la figura 28. Cal recordar que en aquest cas, $E = -1{,}490$.

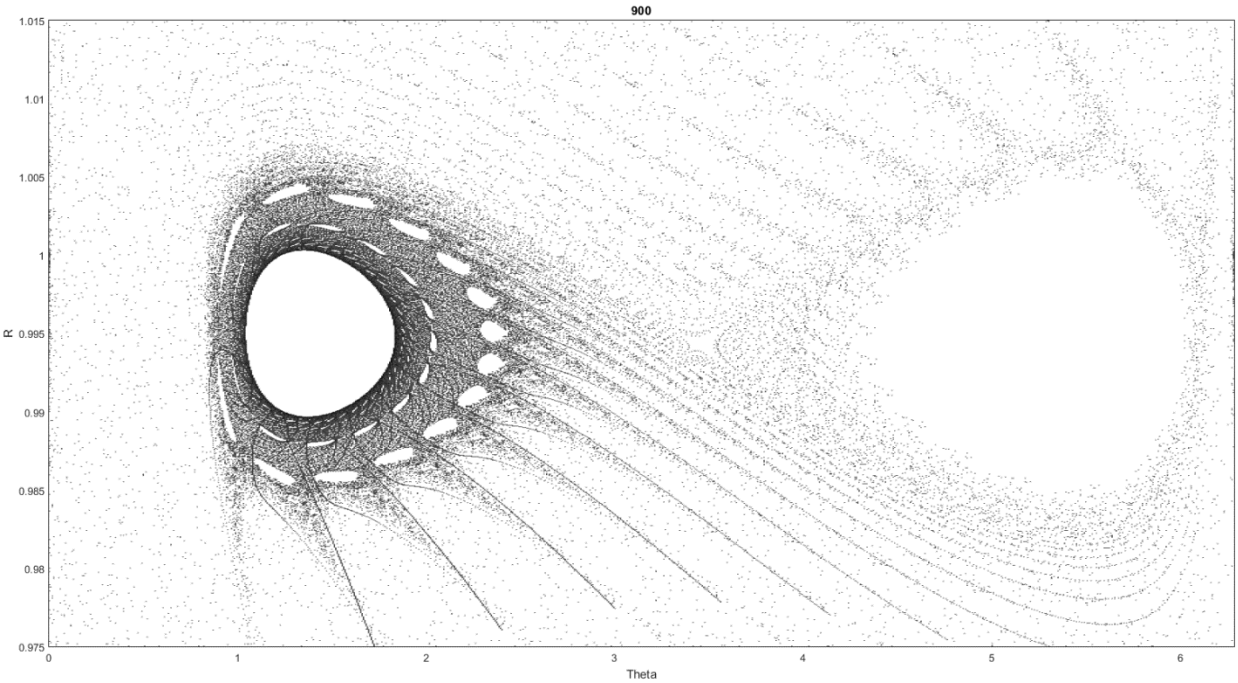


Figura 28: mapa de color representant els resultats de 1001 punts inicials pel sistema Sol-Júpiter $_2$, amb E=-1.490.

Comparant les figures 20 i 28 es pot observar com les àrees al voltant de L_4 i L_5 amb absència d'asteroides incrementa a mesura que incrementa l'energia amb la que s'ha calculat cada gràfic. Tot i que la diferència d'energia entre un gràfic i l'altre és de 0.004, suposa una variació sensible considerant els rangs d'energia per la qual té sentit parlar d'asteroides troians de Júpiter.

La conclusió més rellevant al comparar les dues figures és que, qualitativament, les òrbites que s'hi representen varien molt poc. Així, s'observa que la presència de *jumping trojans* segueix sent important.

5.2- Sol-Terra

Per el cas Sol-Terra s'ha estudiat un rang de valors de $r = [0.994\ , 0.996]$ amb un temps d'integració de 4500 unitats. El resultat està representat en un mapa de color a la figura 29.

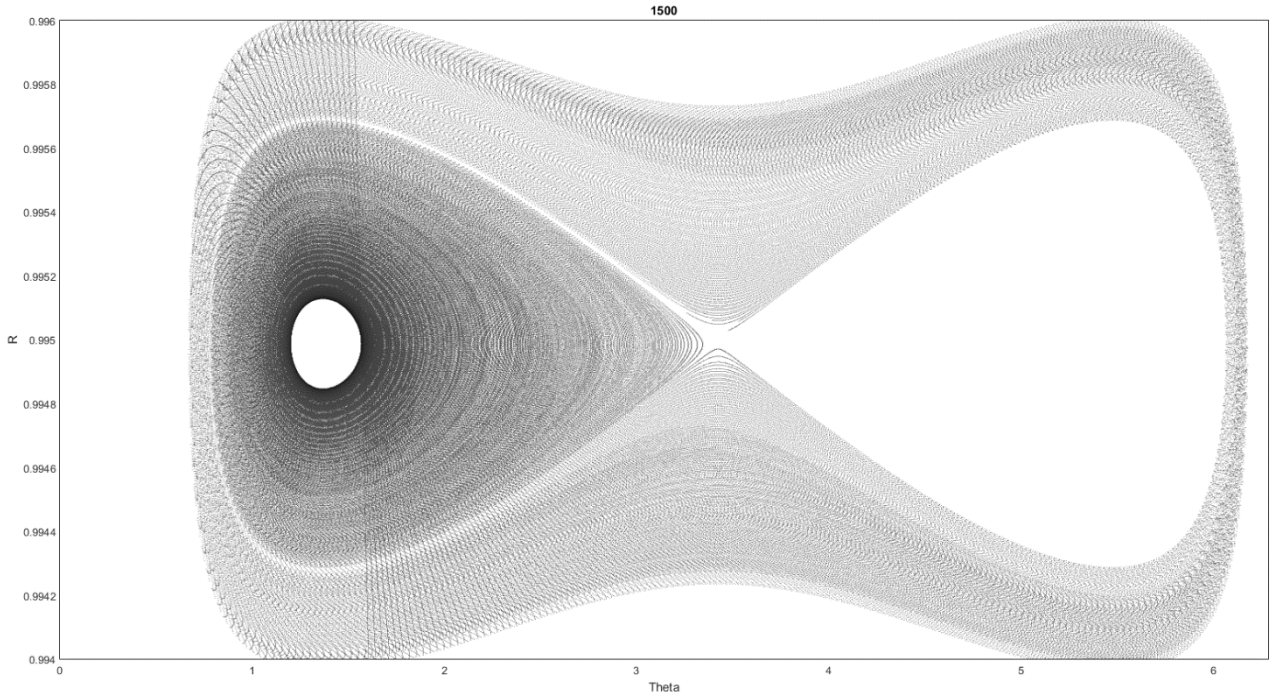


Figura 29: Mapa de color de la trajectòria seguida per 1001 punts inicials en un lapse de temps de 4500 unitats.

A contraposició de les figures 20 i 28, es pot observar clarament que la simulació pels asteroides del sistema Sol-Terra té una separació entre les òrbites tadpole i horseshoe molt reduïda i no s'aprecien a simple vista les varietats invariants, pel que es podria pensar que aquestes, en el sistema Sol-Terra, són molt planes (sense ondulacions importants) i l'àrea dels lòbuls deguts al creuament entre elles és mínima.

Cal concretar que es veu una separació molt clara entre les òrbites tadpole i horseshoe a la part superior dreta del punt L_4. Això és degut a que el temps d'integració no ha estat suficient perquè les òrbites horseshoe més properes a les tadpole poguessin completar un cicle complet. En aquest sentit, fixar-se en la part inferior del gràfic resulta més representatiu per veure la separació entre els dos tipus d'òrbites.

Una altra imatge d'interès és la figura 30, en què es dibuixen per colors una vuitena part de les òrbites calculades. Cal destacar una òrbita en concret, de tipus horseshoe, a la part més interior, just per sobre el punt L_3 i representada de color blau clar. Aquesta òrbita es troba incompleta per falta de temps d'integració, però es pot apreciar que té un punt d'inflexió prop del punt L_3, indicant que no es converteix en una tadpole de L_5 sinó que manté el seu caràcter original.

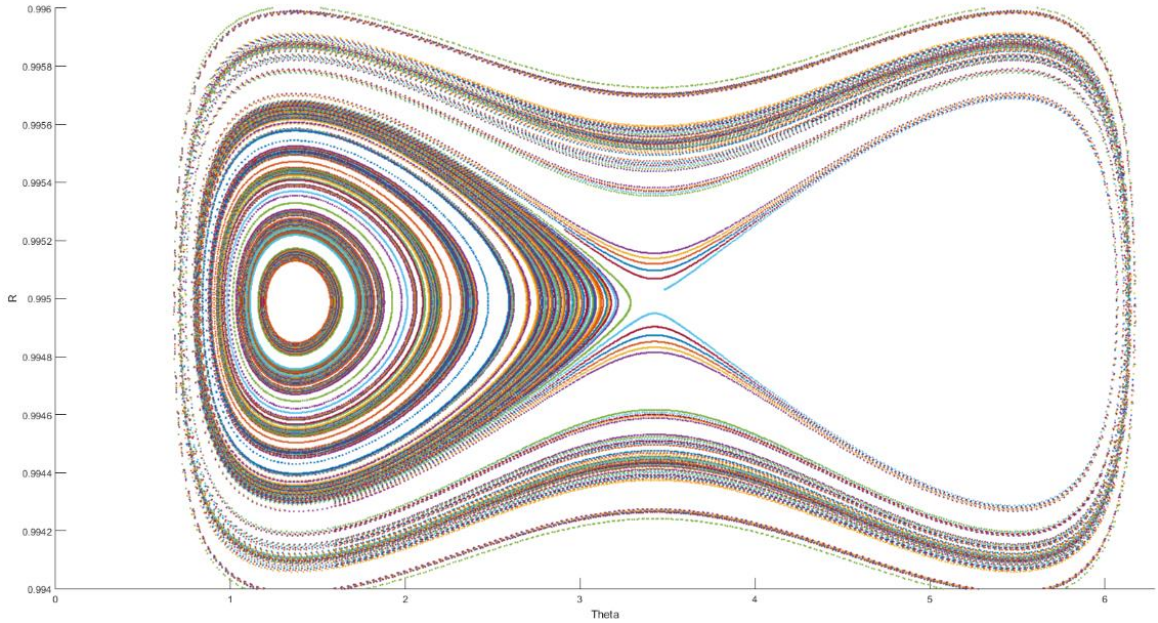


Figura 30: Coloració d'una vuitena part de les òrbites calculades pel sistema Sol-Terra.

5.3- Sol-Neptú

Per el cas Sol-Neptú s'ha estudiat un rang de valors de $r = [0.98\ , 1.01]$ amb un temps d'integració de 4350 unitats. El resultat està representat en un mapa de color a la figura 31, ajustant lleugerament el rang de valors de r representats.

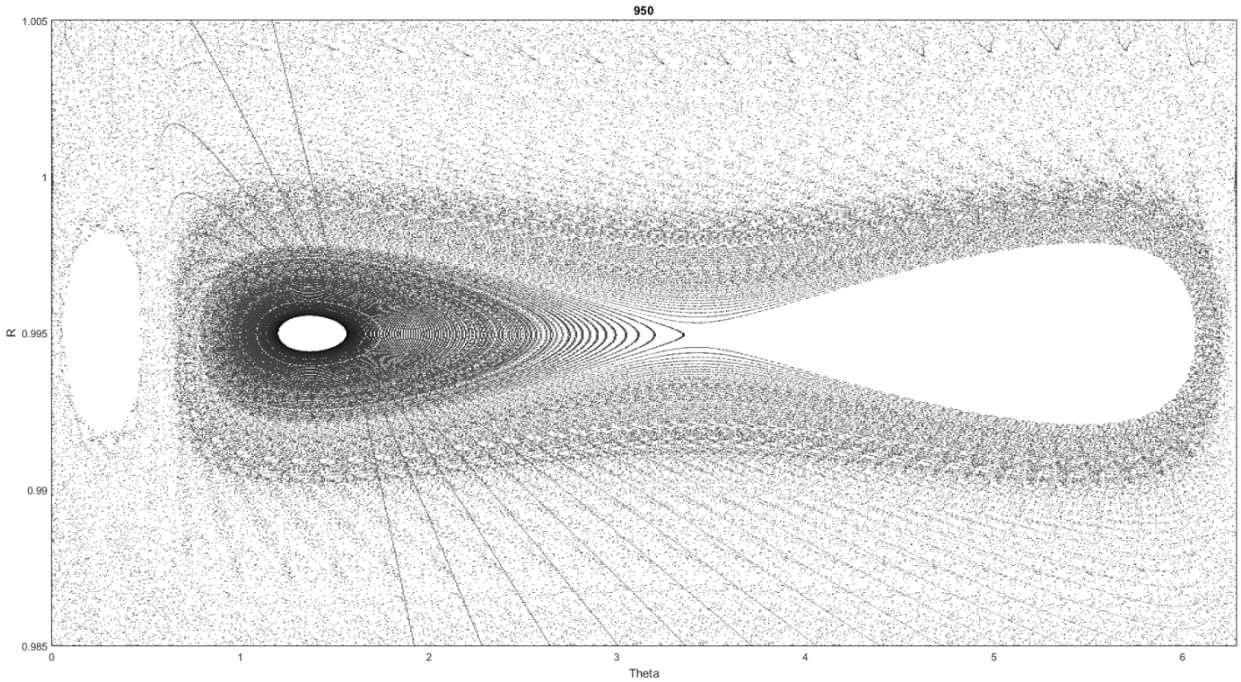


Figura 31: mapa de color de la propagació de 1001 punts inicials durant 4350 unitats de temps pel sistema Sol-Neptú.

En aquesta figura no s'aprecien *jumping trojans* i el marge entre òrbites tadpole i horseshoe segueix sent molt estret, de manera similar a la Terra. Per altra banda, a diferència del cas Sol-Terra sí que s'aprecien les varietats invariants.

La figura 32 il·lustra en diferents colors per les propagacions d'una quarta part dels punts inicials calculats. En aquesta figura es pot comprovar que, certament, les òrbites horseshoe són molt properes a les tadpole, sense haver-hi cap asteroide que adopti els dos tipus de trajectòries (*jumping trojans* o canvis d'òrbita entre horseshoe i tadpoles).

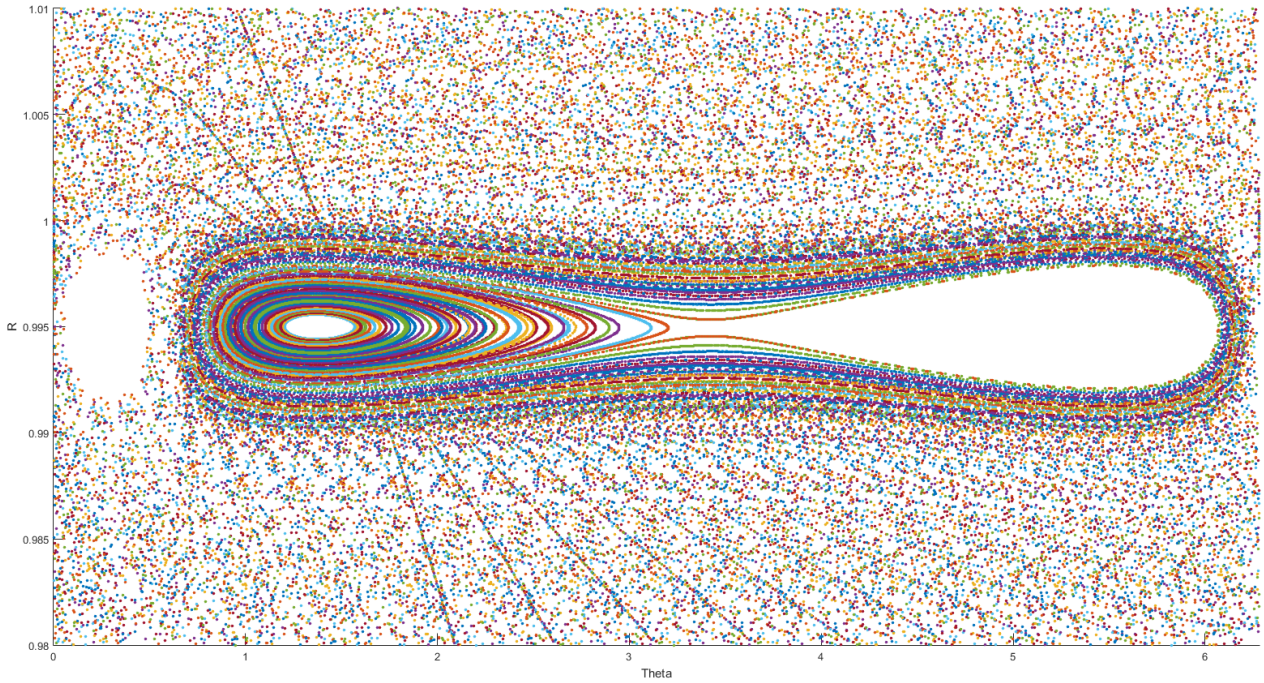


Figura 32: representació en diferents colors per una quarta part de les òrbites calculades al sistema Sol-Neptú.

El paràmetre de masses del sistema Sol-Neptú és de l'òrdre de 10^{-5}, es troba entre els valors del sistema Sol-Terra (d'òrdre de 10^{-6}) i del sistema Sol-Júpiter (d'òrdre de 10^{-4}). Per tant, el resultat del càlcul per aquest sistema és consistent: s'hi aprecien les varietats invariants tal i com passa a Júpiter, però les òrbites horseshoe i tadpole estan molt juntes, deixant poc marge per detectar *jumping trojans*.

Un estudi més detallat de la frontera entre òrbites tadpole i horseshoe pel sistema Sol-Neptú es troba representat a la figura 33.

En aquesta figura s'han triat 21 valors inicials pertanyents a la frontera entre els dos tipus d'òrbites de la figura 31. D'aquesta manera, es pretén comprovar si la frontera entre les òrbites és mes estreta del que realment s'ha vist a la figura 31 o, pel contrari, apareixen diferències importants.

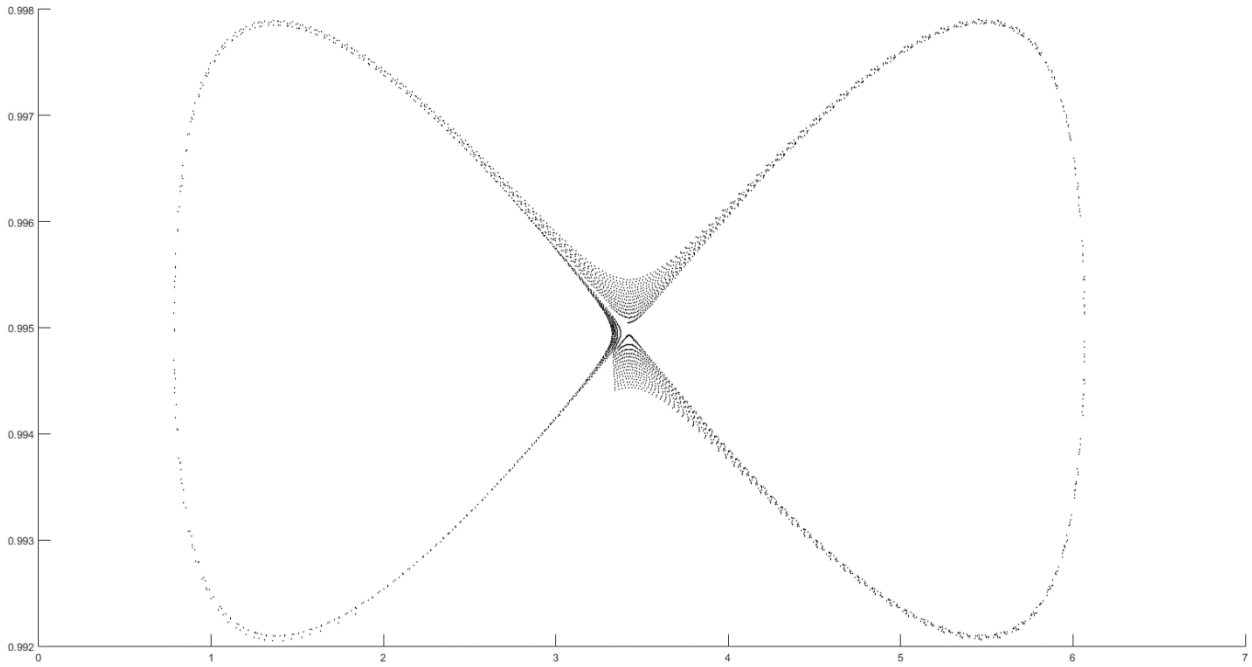


Figura 33: Estudi en detall de la frontera entre òrbites tadpole i horseshoe pel sistema Sol-Neptú. Theta s'ha fixat a $\theta = 3.324$ i s'han calculat 21 condicions inicials cooresponents a valors de $r = [\ 0.9944\ , 0.9949\ ]$, amb un temps d'integració de 1000 unitats (aproximadament 159 voltes de Neptú al Sol).

Com es pot veure a la imatge, sembla que la frontera entre horseshoes i tadpoles és molt estreta, de manera molt similar al sistema Sol-Terra. Per tant, es pot afirmar que una primera aproximació a la recerca de *jumping trojans* de Neptú mostra que la seva existència dins el R3BP és limitada, pel que es recomanaria estudiar el sistema amb un model més elaborat que tingui en compte la pertorbació gravitacional d'algun altre planeta, com podria ser Júpiter per la seva massa o Urani per la seva proximitat.

La figura 34 és un detall de les proximitats de L_3 de la figura 33, on es pot comprovar clarament la separació entre òrbites horseshoe i tadpole de L_4, així com la absència de *jumping trojans.*

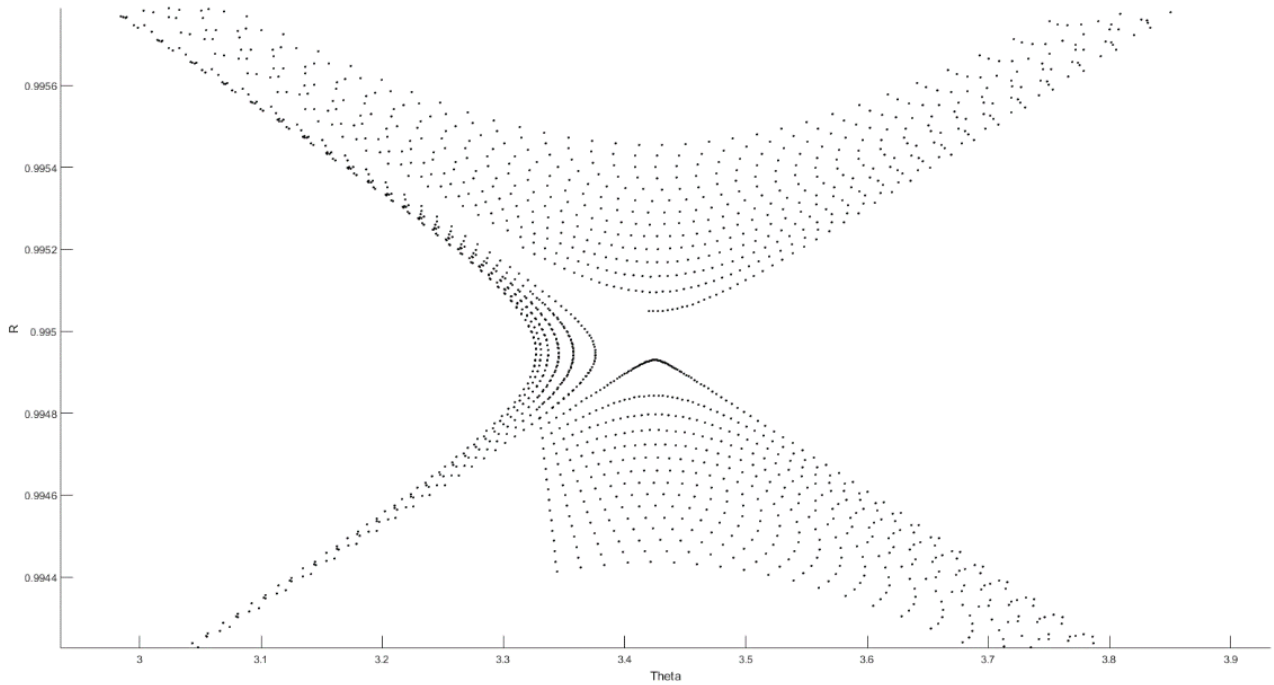


Figura 34: Detall de les proximitats de L_3 de la figura 33.

5.4- Comparació entre els sistemes Sol-Júpiter, Sol-Terra i Sol-Neptú

Comparant les figures 29 i 31, corresponents als mapes de color dels sistemes Sol-Terra i Sol-Neptú respectivament, es pot observar que pel segon cas apareixen clarament representats els plecs de la varietat invariant així com multitud d'asteroides que segueixen trajectòries no estudiades en aquest treball per fora de la zona de horseshoes. Tot i així, la diferència no és suficient per permetre l'aparició de *jumping trojans*.

Introduint dins la comparació la figura 28, corresponent al sistema Sol-Júpiter, es pot observar clarament la presència de *jumping trojans*. Com que la única variable que ha variat per a l'obtenció d'aquestes tres figures és el paràmetre de masses, es pot concloure que aquest és molt rellevant per determinar les probabilitats de trobar-ne, sent directament proporcional el paràmetre de masses i la quantitat de *jumping trojans*.

Cal dir que tots tres casos s'han estudiat dins el R3BP. Com que aquest mètode no considera altres forces gravitacionals més enllà dels dos cossos principals i pren les òrbites d'aquests com circulars, la validesa de les solucions obtingudes i les conclusions que se n'extreuen depenen de la validesa del propi model. Així, els resultats seran més representatius amb la realitat com menys excentricitat tingui l'òrbita real del planeta que s'estudiï i menys pertorbacions existeixin a causa d'altres cossos propers.

El sistema Sol-Júpiter pot estar influenciat per Saturn, que es troba a 4,34 U.A. (1 U.A. es correspon a la distància entre el Sol i la Terra). Tot i la distància, és el segon planeta amb més massa del sistema solar pel que, probablement, sigui el cos que més significatiu a tenir en compte si es volgués obtenir uns resultats més precisos.

El sistema Sol-Terra pot estar fortament influenciat, a part de Júpiter, per la gravetat de Venus, sent la massa d'aquest molt propera a la de la Terra i trobant-se només a 0,28 U.A.

El sistema Sol-Neptú, per altra banda, es troba molt allunyat de tots els altres planetes del sistema solar. Tot i així, Júpiter pot seguir influint fortament en el comportament dels seus asteroides degut a la gran massa que té.

6- Resum del pressupost

Resum del pressupost general d'execució:

PRESSUPOST D'EXECUCIÓ	44.927,40	€
14% de despeses generals d'empresa	6289,836	€
6% de benefici industrial	2695,644	€
PRESSUPOST D'EXECUCIÓ TOTAL A FALTA D'IVA	**53.912,88**	**€**

El pressupost general d'execució a falta d'iva és de cinquanta-tres mil nou-cents dotze euros amb vuitanta-vuit cèntims.

7- Conclusions

En el problema restringit de tres cossos o R3BP, les proximitats dels punts estables de Lagrange L_4 i L_5 s'hi troben caracteritzades per dos tipus d'òrbites ben diferenciades.

Per una banda, les òrbites tadpole orbiten al voltant d'un sol punt estable (L_4 o L_5) i es troben molt a prop d'aquest.

Per altra banda, les òrbites horseshoe engloben els punts L_3, L_4 i L_5, passant a una distància major dels punts estables. Tots dos tipus d'òrbites existeixen dins un rang molt estret d'energia, pel que un asteroide amb una energia fora d'aquest rang seguirà altres tipus de trajectòries.

Existeixen, a més, un tipus d'asteroide capaç de realitzar òrbites tadpole al voltant d'un punt estable i adoptar, en un moment donat, òrbites de tipus horseshoe per acabar adoptant altra vegada una òrbita tadpole al voltant de l'altre punt estable (són anomenats *jumping trojans*) i un tipus d'asteroide capaç de passar de tenir una òrbita tadpole (horseshoe) i adoptar una òrbita horseshoe (tadpole) de manera permanent.

Les òrbites tadpole i horseshoe estan separades per una sèrie d'òrbites associades a òrbites periòdiques al voltant de L_3, anomenades varietats invariants. La detecció de les àrees on es poden donar un o altre tipus d'òrbites es realitza analitzant aquestes varietats, tal com s'ha vist a la figura 19, al capítol 4.

Les òrbites tadpole es donen al voltant dels dos punts estables sense arribar a creuar les varietats invariants. Les òrbites horseshoe, per altra banda, no creuen mai els plecs de les varietats invariants. Per últim, els *jumping trojans* s'originen quan un asteroide creua un lòbul creat entre un plec d'una varietat inestable (o estable) amb una varietat estable (o inestable).

La mida d'aquests lòbuls i, en conseqüència, la quantitat de *jumping trojans* és directament proporcional al paràmetre de masses del sistema i l'energia que s'estudia en varia poc els resultats. En aquest sentit, s'ha trobat que el valor del paràmetre de masses que permeti observar en una primera aproximació *jumping trojans* ha de ser superior al del sistema Sol-Neptú, ja que per aquest no se n'han trobat.

Cal dir que el R3BP no és una representació exacta del sistema Solar, pel que la validesa dels resultats queda subjecta a l'aptitud del model pel sistema que es vulgui estudiar.

7.1- Propostes de millora

Seria interessant realitzar una recerca exhaustiva utilitzant el R3BP per establir una cota inferior del paràmetre de masses a partir de la qual puguin existir *jumping trojans*. Aquest estudi, però, requeriria de gran capacitat de còmput degut a la gran quantitat de càlculs a realitzar. També es podria incloure un estudi sobre les varietats invariants de les òrbites de Lyapunov al voltant del punt L_3.

Per tal d'obtenir uns resultats més propers a la realitat, seria convenient realitzar l'estudi per altres models matemàtics més complexes com podrien ser el model restringit de 4 cossos. A mode d'exemple, pel cas Sol-Terra es podrien considerar les pertorbacions provocades, bé per Júpiter per ser el planeta més massiu del sistema solar, o bé de Venus, per ser molt proper a la Terra.

8- Relació de documents

La relació de documents que componen aquest projecte és:

1- Memòria i Annexes

9- Bibliografia

[1] - AUSTRALIAN SPACE ACADEMY. Asteroid Belt. (http://www.spaceacademy.net.au/index.htm , 26 DE JUNY DE 2015).

[2] - BARRABÉS E. i OLLÉ M. Invariant Manifolds of L_3 and horseshoe motion in the restricted three-body problema. 2006.

[3] DICAU, F. The solution of the n-body Problem, The Mathematical Intelligencer. Vol. 18, p. 66–70. 1996

[4] DR. EDWARD J. WOLLACK. The Lagrange Points. (http://map.gsfc.nasa.gov/mission/observatory_l2.html , 15 D'ABRIL DE 2015).

[5] - E. I. CHIANG & Y. LITHWICK. Neptune Trojans as a Testbed for Planet Formation, The Astrophysical Journal. Vol. 628, p. 520–532. 2005.

[6] - HIRSCH, SMALE, DEVANEY. Differential equations, dynamical systems, and an introduction to chaos (2ed., Elsevier). , 2004.

[7] - OSHIMA K. i YANAO T. Jumping Mechanisms of Trojan asteroids in the planar restricted three- and four- body problems. Vol. 122. P. 53-74

Annexes

A- Codi informàtic

A.1- BiseccioDef

```
function [ coordCart , coordPol , aproximacions ] = BiseccioDef( resu,
resuPol, hmax, variablePS, u )
%Bisecció troba l'instant de temps pel qual una variable es fa zero
%   Està pensat per la variable theta' (resuPol(5))
precisio = 1e-10;  tol=1e-15;  hmax=hmax/20; hmaxOriginal=hmax;
loopLimit=1500;  loopBS=0;  aproximacions=0;

BSspan=[resu(1,1) , resu(1,end)];

while loopBS<loopLimit
    loopBS=loopBS+1; if mod(loopBS,375)==0; display(loopBS); end

    %Valor de thetaP intermedi
    midaP=size(resuPol); esSenar=mod(midaP(2),2);
    switch esSenar
        case 0;    pos=midaP(2)/2;
        otherwise; pos=midaP(2)/2+0.5;
    end
    %Tria dels valors extrems i intermedis de theta prima.
    a=resuPol(variablePS,1);  b=resuPol(variablePS,end);
c=resuPol(variablePS,pos); %Valors de la variable
    %Registre de la secció en cas que es compleixi el criteri de precisió
    if abs(c)<precisio
        coordCart = resu(:,pos);
        coordPol  = resuPol(:,pos);
        break
    end %if

    if         a*c<0 && c*b>0
        BSspan=[resu(1,1) , resu(1,pos)]; xy=resu(2:5,1);
     elseif    a*c>0 && c*b<0
         BSspan=[resu(1,pos) , resu(1,end)]; xy=resu(2:5,pos);
     elseif    a*c<0 && c*b<0
         nota='Hi ha més de 1 tall a secció \n'; display(nota);
acb=[a;c;b]; display(acb);
     elseif    a*c>0 && c*b>0
         nota='Aparentment no hi ha tall a secció \n';display(nota);
    end %if
    %Selecció de hmax per assegurar com a mínim dos passos d'integració.
    if    hmax>(BSspan(2)-BSspan(1))/3;
          hmax=(BSspan(2)-BSspan(1))/3;
     else hmax=hmaxOriginal;
    end
    %Integració amb seguretat contra bucles infinits
    if BSspan(2)-BSspan(1)>=1e-11
        [resu2] = nInt(xy,BSspan,hmax,tol,u);
        [resu2] = ham(resu2,u);
        [resuPol2] = convPol(resu2,u);
    else
```

```
        resu2=[]; resuPol2=[];
    end
    %Registre de la secció en cas que no es pugui continuar integrant.
    sizeTest=size(resuPol2);    punts=[a,c,b];
    if sizeTest(2)<3;  [~,posicio]=min(abs(punts));
        switch posicio
            case 1;    coordCart = resu(:,1);
                       coordPol  = resuPol(:,1);

            case 2;    coordCart = resu(:,pos);
                       coordPol  = resuPol(:,pos);

            case 3;    coordCart = resu(:,end);
                       coordPol  = resuPol(:,end);
        end
        aproximacions=aproximacions+1;
        display(punts);display(coordCart); display(coordPol);
        break
    else
        resu=resu2;
        resuPol=resuPol2;
    end %if
end
if loopBS==loopLimit
    nota='Error a Bisecció';  display(nota);
end %if
end
```

A.2- BiseccioDef_JT

```
function [ coordCart , coordPol , aproximacions ] = BiseccioDef_JT(
resu,resuPol,variablePS,u,fun,tolerancies)
%Bisecció troba l'instant de temps pel qual una variable es fa zero
%   Està pensat per la variable theta' (resuPol(5))
precisio = 1e-10; %hmax=hmax/20; hmaxOriginal=hmax;
loopLimit=1500;  loopBS=0;  aproximacions=0;

BSspan=[resu(1,1) , resu(1,end)];

while loopBS<loopLimit
    loopBS=loopBS+1; if mod(loopBS,250)==0; display(loopBS); end

    %Valor de thetaP intermedi
    midaP=size(resuPol); esSenar=mod(midaP(2),2);
    switch esSenar
        case 0;    pos=midaP(2)/2;
        otherwise; pos=midaP(2)/2+0.5;
    end
    %Tria dels valors extrems i intermedis de theta prima.
    a=resuPol(variablePS,1);  b=resuPol(variablePS,end);
c=resuPol(variablePS,pos); %Valors de la variable
    %Registre de la secció en cas que es compleixi el criteri de precisió
    if abs(c)<precisio
        coordCart = resu(:,pos);
        coordPol  = resuPol(:,pos);
        display('Bisecció finalitzada al '); display(loopBS);
        break
    end %if
```

```
    if          a*c<0 && c*b>0;    BSspan=[resu(1,1) , resu(1,pos)];
xy=resu(2:5,1);
     elseif     a*c>0 && c*b<0;    BSspan=[resu(1,pos) , resu(1,end)];
xy=resu(2:5,pos);
     elseif     a*c<0 && c*b<0;    nota='Hi ha més de 1 tall a secció \n';
display(nota); acb=[a;c;b]; display(acb);
     elseif     a*c>0 && c*b>0;    nota='Aparentment no hi ha tall a secció
\n';display(nota);
    end %if
    %Integració amb seguretat contra bucles infinits
    if abs(BSspan(2)-BSspan(1))>=1e-11
        [T,Y]=ode45(fun,BSspan,xy,tolerancies); resu2=[T,Y]';
        [resu2]=ham(resu2,u);  [resuPol2]=convPol(resu2,u);
    else
        resu2=[]; resuPol2=[];
    end
    %Registre de la secció en cas que no es pugui continuar integrant.
    sizeTest=size(resuPol2);    punts=[a,c,b];
    if sizeTest(2)<3;  [~,posicio]=min(abs(punts));
        switch posicio
            case 1;    coordCart = resu(:,1);
                       coordPol  = resuPol(:,1);

            case 2;    coordCart = resu(:,pos);
                       coordPol  = resuPol(:,pos);

            case 3;    coordCart = resu(:,end);
                       coordPol  = resuPol(:,end);
        end
        aproximacions=aproximacions+1;
display(punts);display(coordCart);display(coordPol);
        break
    else
        resu=resu2;
        resuPol=resuPol2;
    end %if
end
if loopBS==loopLimit
    nota='Error a Bisecció';  display(nota);
end %if
end
```

A.3- Builder

```
%Uneix varis documents en cartesianes i en polars.
%El nom dels fitxers ha de ser 01nom.d fins (com a màxim) 99nom.d
askfNameCart='Part text del nom dels fitxers Cartesians\n';
askfNamePol='Part text del nom dels fitxers en Polars\n';
askNumArxius='Quants arxius hi ha de cada tipus?\n';
askTitolCart='Escriu el títol del fitxer Cartesianes resultant (amb la
extensió .d o .txt)\n';
askTitolPol='Escriu el títol del fitxer Polars resultant (amb la extensió
.d o .txt)\n';

fNameCart=input(askfNameCart,'s');
fNamePol=input(askfNamePol,'s');
numArxius=input(askNumArxius);
```

```
titolCartesianes=input(askTitolCart,'s');
titolPolars=input(askTitolPol,'s');

regMides=zeros(numArxius,1);
for i=1:numArxius
    if i<10;  indexP=['0',num2str(i),fNamePol];
    else      indexP=[num2str(i),fNamePol];
    end

    fName = indexP ;
    fId = fopen( fName, 'r' ) ;
    headerP = strsplit( strtrim( fgetl( fId ) ), ' ' ) ;
    taulaP = fscanf( fId, '%f' ) ;
    taulaP = reshape( taulaP, 5, [] )' ;
    fclose( fId ) ;

    regMides(i)=length(taulaP);
end %Medeix els arxius per predimensionar

indexFinal=cumsum(regMides);
trojans_Cart=zeros(6,indexFinal(end));
trojans_Pol=zeros(5,indexFinal(end));

for i=1:numArxius
    display(i)
    if i<10; indexC=['0',num2str(i),fNameCart];
             indexP=['0',num2str(i),fNamePol];
    else     indexC=['c',num2str(i),'.d'];
             indexP=['p',num2str(i),'.d'];
    end

        fName = indexC ;
    fId = fopen( fName, 'r' ) ;
    headerT = strsplit( strtrim( fgetl( fId ) ), ' ' ) ;
    taulaT = fscanf( fId, '%f' ) ;
    taulaT = reshape( taulaT, 6, [] )' ;
    fclose( fId ) ;
        fName = indexP ;
    fId = fopen( fName, 'r' ) ;
    headerP = strsplit( strtrim( fgetl( fId ) ), ' ' ) ;
    taulaP = fscanf( fId, '%f' ) ;
    taulaP = reshape( taulaP, 5, [] )' ;
    fclose( fId ) ;
    switch i
        case 1
            trojans_Cart(:,1:indexFinal(i))=taulaT';
            trojans_Pol(:,1:indexFinal(i))=taulaP';
        otherwise
            trojans_Cart(:,indexFinal(i-1)+1:indexFinal(i))=taulaT';
            trojans_Pol(:,indexFinal(i-1)+1:indexFinal(i))=taulaP';
    end
end %Uneix els arxius en una sola variable

rotulacioCartesianes = ['Temps ',' X ',' Y ',' xPrima ',' yPrima ',' E'];
rotulacioPolars = ['Temps ',' R ',' Theta ',' rPrima ',' ThetaPrima '];

fileID = fopen(titolCartesianes,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');
fprintf(fileID,'\r\n %24.12f %24.12f %24.12f %24.12f %24.12f
%24.12f',trojans_Cart);fprintf(fileID,'\r\n\r\n');fclose(fileID);
```

```
fileID = fopen(titolPolars,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');
fprintf(fileID,'\r\n %24.12f %24.12f %24.12f %24.12f
%24.12f',trojans_Pol);fprintf(fileID,'\r\n\r\n');fclose(fileID);
```

A.4- condIni_PtoC

```
function [ ab ] = condIni_PtoC( r,theta,thetaP,E,u )
%condIni_PtoC Pren valors inicials de r,theta,thetaP,E,funE,funPol,funPPol
i calcula els valors
%inicials cartesians per començar a resoldre les EDOs
nomF='funX';  [ funX ] = formulari( nomF );
nomF='funY';  [ funY ] = formulari( nomF );
nomF='funRp';  [ funRp ] = formulari( nomF );
nomF='funXp';  [ funXp ] = formulari( nomF );
nomF='funYp';  [ funYp ] = formulari( nomF );

x = feval(funX,r,theta,u);    y = feval(funY,r,theta,u);
rp = feval(funRp,E,x,y,u);
xp = feval(funXp,r,rp,theta,thetaP,u);    yp =
feval(funYp,r,rp,theta,thetaP,u);

ab = [ x,y,xp,yp ];
end
```

A.5- convPol

```
function [ resuPol ] = convPol(resu,u)
nomF='funPol';  [ funPol ] = formulari( nomF ); %Funcions de 'r' i 'theta'
nomF='funPPol';  [ funPPol ] = formulari( nomF ); %Funcions de 'rp' i
'thetap'

mida=size(resu);
resuPol=zeros(mida(2),mida(1)-2);

for i=1:1:mida(2)
    x=resu(2:5,i); %Posicions: x, y, x', y'
    resuP = feval(funPol,x,u); %Posicions: r, theta, r', theta'
    %Cal col·locar cada angle al quadrant correcte.
    if resu(2,i)+u<0 && resu(3,i)<0 %3r Q, tangent positiva
        resuP(2)=resuP(2)+pi;
    elseif resu(2,i)+u>0 && resu(3,i)<0 %4t Q, tangent negativa
        resuP(2)=resuP(2)+2*pi;
    elseif resu(2,i)+u<0 && resu(3,i)>0 %2n Q, tangent negativa
        resuP(2)=resuP(2)+pi;
    end
    resuPP = feval(funPPol,x,resuP(2),u);
    resuPol(i,:) = [resuP ; resuPP];
end
resuPol = [resu(1,:).',resuPol]';
end
```

A.6- formulari

```
function [ funcio ] = formulari( nomF )
%'formulari' Organitza totes les funcions necessàries per la resolució del
%programa principal i les seves subrutines.
%ED a integrar i Hamiltonià, respectivament
fun = @(t,x,u) [ x(3) ; x(4) ; 2.0*x(4)+x(1)+(u-
1)*(u+x(1))/sqrt((x(2)^2+(u+x(1))^2)^3)-u*(u+x(1)-1)/sqrt((x(2)^2+(u+x(1)-
1)^2)^3) ; -2.0*x(3)+x(2)+(u-1)*x(2)/sqrt((x(2)^2+(u+x(1))^2)^3)-
u*x(2)/sqrt((x(2)^2+(u-1+x(1))^2)^3) ];
funE = @(x,u)  0.5*(x(3)^2+x(4)^2)-0.5*(x(1)^2+x(2)^2)-(1-
u)/sqrt((x(1)+u)^2+x(2)^2)-u/sqrt((x(1)-1+u)^2+x(2)^2) ;
%Conversió de r, phi i r' i phi' a cartesianes, respectivament
funPol = @(x,u) [ sqrt((x(1)+u)^2+(x(2)^2)) ; atan(x(2)/(x(1)+u)) ];
funPPol = @(x,phi,u) [ ((x(1)+u)*x(3)+x(2)*x(4))/sqrt((x(1)+u)^2+x(2)^2) ;
(x(4)/(x(1)+u)-x(2)*x(3)/((x(1)+u)^2))*(cos(phi))^2 ];
%Segona derivada de theta (theta'')
funTheta2p = @(x,x2p,y2p,theta,thetaP,u) ( -
(cos(theta))^2*(x(3)*x(4))/(x(1)+u)^2-
y2p/(x(1)+u)+x(4)*x(3)/(x(1)+u)^2+x2p*x(2)/(x(1)+u)^2-
2*x(3)*x(3)*x(2)/(x(1)+u)^3-2*thetaP*cos(theta)*sin(theta)*x(4)/(x(1)+u)-
x(3)*x(2)/(x(1)+u)^2 );
%Variables cartesianes al script condIni
funX = @(r,theta,u) r*cos(theta)-u ;
funY = @(r,theta,u) r*sin(theta) ;
funRp = @(E,x,y,u) sqrt(2.0*E+x^2+y^2+2.0*(1-
u)/sqrt((x+u)^2+y^2)+2*u/sqrt((x-1+u)^2+y^2)) ;
funXp = @(r,rp,theta,thetaP,u) rp*cos(theta)-r*thetaP*sin(theta) ;
funYp = @(r,rp,theta,thetaP,u) rp*sin(theta)+r*thetaP*cos(theta) ;
%Càlcul de yp en funció de x,y,xp i E
funYpCart = @(x,y,xp,E,u) sqrt(E+2*(1-
u)/sqrt((x(1)+u)^2+x(2)^2)+2*u/sqrt((x(1)-1+u)^2+x(2)^2)+(x(1)^2+x(2)^2)-
x(3)) ;
%Coordenades lagrangianes.
funLagrange = @(cg) [1-(cg/3)^(1/3)+(1/3)*(cg/3)^(2/3)-(26/9)*(cg/3) 0 ;
1+(cg/3)^(1/3)+(1/3)*(cg/3)^(2/3)-(28/9)*(cg/3) 0 ; -1-5*cg/12 0 ; 1/2-cg
sqrt(3)/2 ; 1/2-cg -sqrt(3)/2];
funELag = @(x,cg) -0.5*(x(1)^2+x(2)^2)-(1-cg)/sqrt((x(1)+cg)^2+x(2)^2)-
cg/sqrt((x(1)-1+cg)^2+x(2)^2) ;

%Instruccions de crida de les funcions
switch    nomF
    case 'fun';          funcio = fun;
    case 'funE';         funcio = funE;
    case 'funPol';       funcio = funPol;
    case 'funPPol';      funcio = funPPol;
    case 'funTheta2p';   funcio = funTheta2p;
    case 'funX';         funcio = funX;
    case 'funY';         funcio = funY;
    case 'funRp';        funcio = funRp;
    case 'funXp';        funcio = funXp;
    case 'funYp';        funcio = funYp;
    case 'funLagrange';  funcio = funLagrange;
    case 'funELag';      funcio = funELag;
    case 'funYpCart';    funcio = funYpCart;
    otherwise;           disp('ERROR al cridar una funció \n');
end %'swtich'
end
```

A.7- ham

```
function [resu]=ham(resu,u)

nomF='funE';  [ funE ] = formulari( nomF );
mida=size(resu);
H=zeros(1,mida(2));

x=resu(2:5,:);
H(1,:)=feval(funE,x(:,:),u);
resu = [resu ; H];
end
```

A.8- integrador_cart_pol_E

```
%Resolució de sistemes d'equacions diferencials per n equacions i càlcul
del hamiltonià.
format long e; clear;
tol = 1e-12;
% hmax = pas màxim
hmax = 1e-2;
u=9.537e-4;
%'ab' vector fila de les condicions inicials x(1),..,x(n) en cartesianes
r = 0.983;    theta = pi/2;    thetaP = 0;    E=-1.494;
[ ab ] = condIni_PtoC(r,theta,thetaP,E,u);

%tspan = [tInicial , tfinal]
tspan = [0 , 200];

%Execució del programa
[ resu ] = nInt( ab , tspan, hmax, tol,u);
[ resu ] = ham(resu,u);
[ resuPol ] = convPol(resu,u);

%-------GUARDAR ELS RESUPTATS OBTINGUTS EN ARXIUS EXTERNS-------%
%'titol' string de títol del .txt on es guardaran les dades
%'titolP' string de títol del .txt de les coordenades polars
titol='HS_J_C.d';    titolP='HS_J_P.d';
%'rotulacio' títols de les columnes de 'Cartesianes'
%'rotulacioP' títols de les columnes de 'Polars'
rotulacio = ['     temps          ','X            ','Y            ','vX
','vY          ','E(H)'];
rotulacioP = ['     temps          ','R            ','Phi          ','Rp
','Phi p'];
%Creació i escriptura dels arxius .d finals
fileID = fopen(titol,'w');
fprintf(fileID,'%20s %15s %15s %15s %15s %7s\r\n',rotulacio);
fprintf(fileID,'\r\n');
fprintf(fileID,'\r\n %24.12f %24.12f %24.12f %24.12f %24.12f
%24.12f',resu);
fclose(fileID);

fileID = fopen(titolP,'w');
fprintf(fileID,'%20s %15s %15s %7s\r\n',rotulacioP);
fprintf(fileID,'\r\n');
fprintf(fileID,'\r\n %24.12f %24.12f %24.12f %24.12f %24.12f',resuPol);
```

```
fclose(fileID);

plot(resu(2,:),resu(3,:)); xlabel('X'); ylabel('Y');
```

A.9- jumpingTrojans

```
clear
% Recull strings dels inputs
askR='Valor de "r" inicial\n';
askTheta='Valor de "theta" inicial\n';
askE='Valor de "E"\n';
askU='Planeta:\n t = Terra\n j = Júpiter\n n = Neptú\n';
askTF='Temps futur que es vol calcular\n';
askTP='Temps passat es vol calcular\n';
askCalc='Calcular orbites(o), PS(p) o ambdues(a) ?\n';
askNPS='Nombre superior al màxim de seccions de poincaré\n';
askTitol='Títol general pels fitxers .d\n';
askTitolCF='Títol dels resultats de la òrbita futura en cartesianes\n';
askTitolPF='Títol dels resultats de la òrbita futura en polars\n';
askTitolCP='Títol dels resultats de la òrbita passada en cartesianes\n';
askTitolPP='Títol dels resultats de la òrbita passada en polars\n';
askTitolPS_CF='Títol dels resultats de la PS futura en cartesianes\n';
askTitolPS_PF='Títol dels resultats de la PS futura en polars\n';
askTitolPS_CP='Títol dels resultats de la PS passada en cartesianes\n';
askTitolPS_PP='Títol dels resultats de la PS passada en polars\n';

% Establir i carregar funcions i constants
[fun] = formulari('fun');
tolerancies = odeset('RelTol',1e-12,'AbsTol',[1e-13 1e-13 1e-13 1e-
13],'Refine',8);
variablePS=5;
vControl=4;
rotulacioCartesianes = ['Temps ',' X ',' Y ',' xPrima ',' yPrima ',' E'];
rotulacioPolars = ['Temps ',' R ',' Theta ',' rPrima ',' ThetaPrima '];

% Introduïr dades punt inicial, temps futur (F), passat (P) i
especificacions de càlcul
r=input(askR);
theta=input(askTheta);
E=input(askE);
tF=input(askTF); tF=abs(tF);
tP=input(askTP); tP=abs(tP)*-1;
nPS=input(askNPS);
planeta=input(askU,'s');
switch planeta %Establir la 'u' pertinent en funció del planeta escollit
    case {'t','Terra','terra'}
        u=3.036e-6;
    case {'j','Jupiter','jupiter'}
        u=9.537e-4;
    case {'n','Neptu','neptu'}
        u=5.151e-5;
end
calc=input(askCalc,'s');
tGral=input(askTitol,'s');
switch calc %Demanar noms de fitxers i creació d'aquests en funció delc
càlculs escollits
    case 'o'
        titolCF=[tGral,'_F_cart.d'];
```

```
        titolPF=[tGral,'_F_pol.d'];
        titolCP=[tGral,'_P_cart.d'];
        titolPP=[tGral,'_P_pol.d'];
        fileID = fopen(titolCF,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPF,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolCP,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPP,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);

    case 'p'
        titolPS_CF=[tGral,'_PS_F_cart.d'];
        titolPS_PF=[tGral,'_PS_F_pol.d'];
        titolPS_CP=[tGral,'_PS_P_cart.d'];
        titolPS_PP=[tGral,'_PS_P_pol.d'];
        fileID = fopen(titolPS_CF,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPS_PF,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPS_CP,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPS_PP,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);

    case 'a'
        titolCF=[tGral,'_F_cart.d'];
        titolPF=[tGral,'_F_pol.d'];
        titolCP=[tGral,'_P_cart.d'];
        titolPP=[tGral,'_P_pol.d'];
        titolPS_CF=[tGral,'_PS_F_cart.d'];
        titolPS_PF=[tGral,'_PS_F_pol.d'];
        titolPS_CP=[tGral,'_PS_P_cart.d'];
        titolPS_PP=[tGral,'_PS_P_pol.d'];
        fileID = fopen(titolCF,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPF,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolCP,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPP,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPS_CF,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPS_PF,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPS_CP,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
        fileID = fopen(titolPS_PP,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);

end

% Preparar condicions inicials
ab = condIni_PtoC(r,theta,0,E,u);
fun=@(t,x) fun(t,x,u);

% Calcular òrbites (F i P)
```

```
if strcmpi(calc,'o') || strcmpi(calc,'a'); % Càlcul de les òrbites
"contínues"
        [TF,YF]=ode45(fun,[0 tF],ab,tolerancies); resu_F=[TF,YF]';
            [resu_F]=ham(resu_F,u);     [resuPol_F]=convPol(resu_F,u);
        [TP,YP]=ode45(fun,[0 tP],ab,tolerancies); resu_P=[TP,YP]';
            [resu_P]=ham(resu_P,u);     [resuPol_P]=convPol(resu_P,u);
        % Guardar els resultats en .d
        fileID = fopen(titolCF,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f %24.12f
%24.12f',resu_F);fprintf(fileID,'\r\n\r\n');fclose(fileID);
        fileID = fopen(titolPF,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f
%24.12f',resuPol_F);fprintf(fileID,'\r\n\r\n');fclose(fileID);
        fileID = fopen(titolCP,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f %24.12f
%24.12f',resu_P);fprintf(fileID,'\r\n\r\n');fclose(fileID);
        fileID = fopen(titolPP,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f
%24.12f',resuPol_P);fprintf(fileID,'\r\n\r\n');fclose(fileID);
if calc=='o';return;end
end

% Calcular de les seccions de poincaré (F i P)
[resuPS_F , resuPSPol_F] =
Tim_PS_solver_JT(ab,tF,nPS,variablePS,vControl,u,fun,tolerancies,'F');
[resuPS_P , resuPSPol_P] =
Tim_PS_solver_JT(ab,tP,nPS,variablePS,vControl,u,fun,tolerancies,'P');
    % Guardar els resultats en .d
    fileID = fopen(titolPS_CF,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f %24.12f
%24.12f',resuPS_F);fprintf(fileID,'\r\n\r\n');fclose(fileID);
    fileID = fopen(titolPS_PF,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f
%24.12f',resuPSPol_F);fprintf(fileID,'\r\n\r\n');fclose(fileID);
    fileID = fopen(titolPS_CP,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f %24.12f
%24.12f',resuPS_P);fprintf(fileID,'\r\n\r\n');fclose(fileID);
    fileID = fopen(titolPS_PP,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f
%24.12f',resuPSPol_P);fprintf(fileID,'\r\n\r\n');fclose(fileID);
```

A.10- moulder

```
%Elimina les seccions de Poincaré amb una energia erronea
energia=-1.490;
fNameCart='trojans_ST_total_Cart.d';
fNamePol='trojans_ST_total_Pol.d';

%Codi fixe
fId = fopen( fNameCart, 'r' ) ;
headerC = strsplit( strtrim( fgetl( fId ) ), ' ' ) ;
brutC = fscanf( fId, '%f' ) ;
brutC = reshape( brutC, 6, [] )' ;
fclose( fId ) ;

fId = fopen( fNamePol, 'r' ) ;
headerP = strsplit( strtrim( fgetl( fId ) ), ' ' ) ;
brutP = fscanf( fId, '%f' ) ;
```

```
brutP = reshape( brutP, 5, [] )' ;
fclose( fId ) ;

neteja=find(mod(brutC(:,6),energia)~=0);
netC=removerows(brutC,neteja);
netP=removerows(brutP,neteja);
midaBC=size(brutC);    midaNC=size(netC);
display(midaBC(1)); display(midaNC(1));
Percentatge_Eliminat=100*(midaBC(1)-midaNC(1))/midaBC(1);
display(Percentatge_Eliminat);
switch Percentatge_Eliminat
    case 0
        nota='No hi ha valors erronis. als arxius originals';
        display(nota); beep;
    otherwise
askTitolCart='Escriu el títol del fitxer Cartesianes resultant (amb la
extensió .d o .txt)\n';
askTitolPol='Escriu el títol del fitxer Polars resultant (amb la extensió
.d o .txt)\n';
titolCartesianes=input(askTitolCart,'s');
titolPolars=input(askTitolPol,'s');
rotulacioCartesianes = ['Temps ',' X ',' Y ',' xPrima ',' yPrima ',' E'];
rotulacioPolars = ['Temps ',' R ',' Theta ',' rPrima ',' ThetaPrima '];

fileID = fopen(titolCartesianes,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');
fprintf(fileID,'\r\n %24.12f %24.12f %24.12f %24.12f %24.12f
%24.12f',netC');fprintf(fileID,'\r\n\r\n');fclose(fileID);

fileID = fopen(titolPolars,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');
fprintf(fileID,'\r\n %24.12f %24.12f %24.12f %24.12f
%24.12f',netP');fprintf(fileID,'\r\n\r\n');fclose(fileID);
end
```

A.11- nInt

```
function [ resu] = nInt( ab , tspan, hmax, tol, u)
%Resol un sistema d'ED's mitjançant l'algorisme ode78
%Serveix per sistemes de n ED's (de 1 a inf).
ode_fcn_format = 0;
trace = 0;
count = 0;

nomF='fun';  [ fun ] = formulari( nomF );

[t , y] = ode78 (fun, tspan, ab, ode_fcn_format, tol, trace, count,
hmax,u);
resu = [t , y]; resu=resu';
end
```

A.12- ode78

```
function [tout,xout] =
ode78(FUN,tspan,x0,ode_fcn_format,tol,trace,count,hmax,u)
```

```
% Copyright (C) 2001, 2000 Marc Compere
% This file is intended for use with Octave.
% ode78.m is free software; you can redistribute it and/or modify it
% under the terms of the GNU General Public License as published by
% the Free Software Foundation; either version 2, or (at your option)
% any later version.
%
% ode78.m is distributed in the hope that it will be useful, but
% WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
% General Public License for more details at www.gnu.org/copyleft/gpl.html.
%
% ---------------------------------------------------------------------
%
% ode78 (v1.14) Integrates a system of ordinary differential equations
using
% 7th order formulas.
%
% This is a 7th-order accurate integrator therefore the local error
normally
% expected is O(h^8).  However, because this particular implementation
% uses the 8th-order estimate for xout (i.e. local extrapolation) moving
% forward with the 8th-order estimate will yield errors on the order of
O(h^9).
%
% The order of the RK method is the order of the local *truncation* error,
d,
% which is the principle error term in the portion of the Taylor series
% expansion that gets dropped, or intentionally truncated.  This is
different
% from the local error which is the difference between the estimated
solution
% and the actual, or true solution.  The local error is used in stepsize
% selection and may be approximated by the difference between two estimates
of
% different order, l(h) = x_(O(h+1)) - x_(O(h)).  With this definition, the
% local error will be as large as the error in the lower order method.
% The local truncation error is within the group of terms that gets
multipled
% by h when solving for a solution from the general RK method.  Therefore,
the
% order-p solution created by the RK method will be roughly accurate to
O(h^(p+1))
% since the local truncation error shows up in the solution as h*d, which
is
% h times an O(h^(p)) term, or rather O(h^(p+1)).
% Summary:   For an order-p accurate RK method,
%            - the local truncation error is O(h^p)
%            - the local error used for stepsize adjustment and that
%              is actually realized in a solution is O(h^(p+1))
%
% This requires 13 function evaluations per integration step.
%
% Relevant discussion on step size choice can be found on pp.90,91 in
% U.M. Ascher, L.R. Petzold, Computer Methods for  Ordinary Differential
Equations
% and Differential-Agebraic Equations, Society for Industrial and Applied
Mathematics
% (SIAM), Philadelphia, 1998
%
```

```
% More may be found in the original author's text containing numerous
% applications on ordinary and partial differential equations using Matlab:
%
%     Howard Wilson and Louis Turcotte, 'Advanced Mathematics and
%     Mechanics Applications Using MATLAB', 2nd Ed, CRC Press, 1997
%
%
% [tout, xout] = ode78(FUN,tspan,x0,ode_fcn_format,tol,trace,count,hmax)
%
% INPUT:
% FUN   - String containing name of user-supplied problem description.
%         Call: xprime = fun(t,x) where FUN = 'fun'.
%         t      - Time (scalar).
%         x      - Solution column-vector.
%         xprime - Returned derivative COLUMN-vector; xprime(i) = dx(i)/dt.
% tspan - [ tstart, tfinal ]
% x0    - Initial value COLUMN-vector.
% ode_fcn_format - this specifies if the user-defined ode function is in
%         the form:     xprime = fun(t,x)   (ode_fcn_format=0, default)
%         or:           xprime = fun(x,t)   (ode_fcn_format=1)
%         Matlab's solvers comply with ode_fcn_format=0 while
%         Octave's lsode() and sdirk4() solvers comply with
ode_fcn_format=1.
% tol   - The desired accuracy. (optional, default: tol = 1.e-6).
% trace - If nonzero, each step is printed. (optional, default: trace = 0).
% count - if nonzero, variable 'rhs_counter' is initalized, made global
%         and counts the number of state-dot function evaluations
%         'rhs_counter' is incremented in here, not in the state-dot file
%         simply make 'rhs_counter' global in the file that calls ode78
% hmax  - limit the maximum stepsize to be less than or equal to hmax
%
% OUTPUT:
% tout  - Returned integration time points (row-vector).
% xout  - Returned solution, one solution column-vector per tout-value.
%
% The result can be displayed by: plot(tout, xout).

%   Daljeet Singh & Howard Wilson
%   Dept. Of Electrical Engg., The University of Alabama.
%   11-24-1988.
%
% modified by:
% Marc Compere
% CompereM@asme.org
% created : 06 October 1999
% modified: 19 May 2001


% The Fehlberg coefficients:
alpha_ = [ 2./27., 1/9, 1/6, 5/12, 0.5, 5/6, 1/6, 2/3, 1/3, 1, 0, 1 ]';
beta_  = [ 2/27, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ;
          1/36, 1/12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ;
          1/24, 0, 1/8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ;
          5/12, 0, -25/16, 25/16, 0, 0, 0, 0, 0, 0, 0, 0, 0 ;
          0.05, 0, 0, 0.25, 0.2, 0, 0, 0, 0, 0, 0, 0, 0 ;
          -25/108, 0, 0, 125/108, -65/27, 125/54, 0, 0, 0, 0, 0, 0, 0 ;
          31/300, 0, 0, 0, 61/225, -2/9, 13/900, 0, 0, 0, 0, 0, 0 ;
          2, 0, 0, -53/6, 704/45, -107/9, 67/90, 3, 0, 0, 0, 0, 0 ;
          -91/108, 0, 0, 23/108, -976/135, 311/54, -19/60, 17/6, -1/12, 0,
0, 0, 0 ;
```

```
          2383/4100, 0, 0, -341/164, 4496/1025, -301/82, 2133/4100, 45/82,
45/164, 18/41, 0, 0, 0 ;
          3/205, 0, 0, 0, 0, -6/41, -3/205, -3/41, 3/41, 6/41, 0, 0, 0 ;
          -1777/4100, 0, 0, -341/164, 4496/1025, -289/82, 2193/4100, 51/82,
33/164, 12/41, 0, 1, 0 ]';
chi_  = [ 0, 0, 0, 0, 0, 34/105, 9/35, 9/35, 9/280, 9/280, 0, 41/840,
41/840]';
psi_  = [ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, -1 ]';

pow = 1/8; % see p.91 in the Ascher & Petzold reference for more
infomation.


if nargin < 8, hmax = (tspan(2) - tspan(1))/2.5; end
if nargin < 7, count = 0; end
if nargin < 6, trace = 0; end
if nargin < 5, tol = 1.e-6; end
if nargin < 4, ode_fcn_format = 0; end

% Initialization
t0 = tspan(1);
tfinal = tspan(2);
t = t0;
% the following step parameters are used in ODE45
% hmax = (tfinal - t)/5;
% hmin = (tfinal - t)/20000;
% h = (tfinal - t)/100;
% The following parameters were taken because the integrator has
% higher order than ODE45. This choice is somewhat subjective.
%hmin = (tfinal - t)/10000;
hmin = (tfinal - t)/1e20;
h = (tfinal - t)/50;
x = x0(:);          % the '(:)' ensures x is initialized as a column vector
f = x*zeros(1,13);  % f needs to be an Nx13 matrix where N=number of rows
in x
tout=t;
xout=x.';
tau = tol * max(norm(x,'inf'), 1);
% ¡¡¡ MODIFICACIÓ DE FLL PER MILLORAR VELOCITAT !!! LINIES seguents CTRL+R
% if count==1,
%  global rhs_counter
%  if ~exist('rhs_counter'),rhs_counter=0; end
% end % if count

if trace
%  clc, t, h, x
%   clc, t, x
   clc, t
end

% The main loop
   while t<tfinal && h>=hmin %NOTA: no hauria de ser 'abs' a tot?
      if (t + h) > tfinal, h = tfinal - t; end %NOTA: 'abs'

      % Compute the slopes
      if (ode_fcn_format==0), % (default)
         f(:,1) = feval(FUN,t,x,u);
         for j = 1: 12,
            f(:,j+1) = feval(FUN, t+alpha_(j)*h, x+h*f*beta_(:,j),u);
         end
```

```
      else % ode_fcn_format==1
         f(:,1) = feval(FUN,x,t,u);
         for j = 1: 12,
            f(:,j+1) = feval(FUN, x+h*f*beta_(:,j), t+alpha_(j)*h,u);
         end
      end % if (ode_fcn_format==1)


      % increment rhs_counter
      if count==1, rhs_counter = rhs_counter + 13; end

      % Truncation error term
      gamma1 = h*41/840*f*psi_;

      % Estimate the error and the acceptable error
      delta = norm(gamma1,'inf');
      tau = tol*max(norm(x,'inf'),1.0);

      % Update the solution only if the error is acceptable
      if delta <= tau
         t = t + h;
         x = x + h*f*chi_;  % this integrator uses local extrapolation
         tout = [tout; t];
         xout = [xout; x.'];
      end
      if trace
%        home, t, h, x
%        home, t, x
         home, t, h
      end

      % Update the step size
      if delta == 0.0
       delta = 1e-16;
      end
      h = min(hmax,0.8*h*(tau/delta)^pow); %NOTA: cal revisar això per
integrar enrere?

   end;

   if (t < tfinal) %NOTA: 'abs'
      disp('SINGULARITY LIKELY.')
      t
   end
```

A.13- plotter

```
%Permet graficar arxius externs amb diferents opcions.
%Càrrega dels fitxers de dades:
askImport='Carregar fitxers? \n    y  --> yes \n   [ ] --> no \n';
import=input(askImport,'s');
if strcmpi(import,'y') || strcmpi(import,'yes');
    clear all
    askTitolFitxer='Escriu el títol del fitxer que vols carregar\n';
    fName=input(askTitolFitxer,'s');
    askColumnes='Quantes columnes té (Cart --> 6   ||   Pol --> 5)\n';
    nColumn=input(askColumnes);
```

```
     tic
     fId = fopen( fName, 'r' ) ;
     header = strsplit( strtrim( fgetl( fId ) ), ' ' ) ;
     taula = fscanf( fId, '%f' ) ;
     taula = reshape( taula, nColumn, [] )' ;
     fclose( fId ) ;
    timImportData=toc; display(timImportData)
end %'if' per carregar arxius
%-----------------------------------------------------------------------------
--------------------
askPlot='Realitzar plots? \n    y  --> yes \n   [ ] --> no \n';
import=input(askPlot,'s');
if strcmpi(import,'y') || strcmpi(import,'yes')
% Plots -->
    askPlaneta='Quin planeta representa? (j,t,n)\n';
    planeta=input(askPlaneta,'s');
    switch planeta
        case 'j'
            eixos=[0 2*pi 0.98 1.02]; midaColors=10;
        case 't'
            eixos=[0 2*pi 0.994 0.996]; midaColors=4;
        case 'n'
            eixos=[0 2*pi 0.985 1.005]; midaColors=6;
        otherwise
            eixos=[]; midaColors=[];
    end
    askTipusPlot='Scatter o Plot?\n';
    Tipus=input(askTipusPlot,'s');
    askEixX='Quina columna va a leix X?\n';
    EixX=input(askEixX);
    askEixY='Quina columna va a leix Y?\n';
    EixY=input(askEixY);
    askNomX='Nom de leix X\n';
    askNomY='Nom de leix Y\n';
    NomX=input(askNomX,'s');
    NomY=input(askNomY,'s');
    askParcial='Graficar tots els punts o seleccionar un grup?\n';
    parcial=input(askParcial,'s');
    switch parcial
        case {'tots','t'}
            switch Tipus
                case {'scatter','s'}
                    figure;
scatter(taula(:,EixX),taula(:,EixY),1,'k','fill');hold on
                    xlabel(NomX); ylabel(NomY); axis(eixos);
                case {'plot','p'}
                    figure; plot(taula(:,EixX),taula(:,EixY));hold on
                    xlabel(NomX); ylabel(NomY); axis(eixos);
            end
        case {'grup','g'}
            t=taula(:,1);  grups=1;  iniciGrup(grups)=1;
            for i=1:length(t)-1
                ti=t(i);    tj=t(i+1);
                if tj<ti
                    grups=grups+1;
                    iniciGrup(grups)=i+1;
                end
            end
            iniciGrup(grups+1)=length(t)+1; %Indica la posició on
començaria un últim grup imaginari
            display(grups)
```

```
            askGrup='Quin grup es vol graficar (s,p o valor)?\n';
            grup=input(askGrup,'s');
            nombre=str2double(grup);
            if isnan(nombre)
                askSalt='Cada quants grups vols dibuixar? (2, 3, 4...)\n';
                salt=input(askSalt);
            end
            switch grup
                case {'senars','s'}
                    figure; hold on
                    for nombre=1:salt:grups
                        inici=iniciGrup(nombre);
                        final=iniciGrup(nombre+1)-1;
                        switch Tipus
                        case {'scatter','s'}

scatter(taula(inici:final,EixX),taula(inici:final,EixY),midaColors,'fill');
hold on
                                xlabel(NomX); ylabel(NomY); axis(eixos);
                        case {'plot','p'}

plot(taula(inici:final,EixX),taula(inici:final,EixY));hold on
                                xlabel(NomX); ylabel(NomY); axis(eixos);
                        end
                    end
                case {'parells','p'}
                    figure; hold on
                    for nombre=2:salt:grups
                        inici=iniciGrup(nombre);
                        final=iniciGrup(nombre+1)-1;
                        switch Tipus
                        case {'scatter','s'}

scatter(taula(inici:final,EixX),taula(inici:final,EixY),midaColors,'fill');
hold on
                                xlabel(NomX); ylabel(NomY); axis(eixos);
                        case {'plot','p'}

plot(taula(inici:final,EixX),taula(inici:final,EixY));hold on
                                xlabel(NomX); ylabel(NomY); axis(eixos);
                        end
                    end
                otherwise
                inici=iniciGrup(nombre);
                final=iniciGrup(nombre+1)-1;
                switch Tipus
                    case {'scatter','s'}
                            figure;
scatter(taula(inici:final,EixX),taula(inici:final,EixY),1,'.b');hold on
                            xlabel(NomX); ylabel(NomY); axis(eixos); hold
off
                    case {'plot','p'}
                            figure;
plot(taula(inici:final,EixX),taula(inici:final,EixY));hold on
                            xlabel(NomX); ylabel(NomY); axis(eixos); hold
off
                end
            end
    end
    hold off
    % <-- Plots
```

```
end
```

A.14- plotter_moulder

```
%Elimina les seccions de Poincaré amb una energia erronea

%Codi fixe
%Obtenció dels arxius originals
askLoad='Carregar fitxers de dades? y/n\n';
load=input(askLoad,'s');
if strcmpi(load,'y')
    clear all
    askNP='Nom del document amb coord. polars\n';
    fNamePol=input(askNP,'s');

        fId = fopen( fNamePol, 'r' ) ;
    headerP = strsplit( strtrim( fgetl( fId ) ), ' ' ) ;
    brutP = fscanf( fId, '%f' ) ;
    brutP = reshape( brutP, 5, [] )' ;
    fclose( fId ) ;
end

%Elaboració del mallat i posició de cada punt dins el mallat
askPlaneta='Tria un planeta (j,t,n)\n';
askMida='Tria un tamany del mallat\n';
planeta=input(askPlaneta,'s');
switch planeta
    case 'j'
        xi=0.98; xf=1.02;
    case 't'
        xi=0.994; xf=0.996;
    case 'n'
        xi=0.985; xf=1.005;
end
mida=input(askMida);
x=linspace(0,2*pi,mida);
y=linspace(xi,xf,mida);
[X,Y]=meshgrid(x,y);

px=(2*pi-0)/mida; py=(xf-xi)/mida;
rx=floor(brutP(:,3)/px);
ry=floor((brutP(:,2)-xi)/py);
corrZerosRX=find(rx==0); rx(corrZerosRX)=1;
corrZerosRY=find(ry==0); ry(corrZerosRY)=1;

%Eliminació dels punts que es troben fora dels límits del mallat
neteja=find(rx<1 | rx>mida | ry<0 | ry>mida);
midaBrut=length(rx);
rx=removerows(rx,neteja); ry=removerows(ry,neteja);
midaNet=length(rx);
Percentatge_Eliminat=100*(midaBrut-midaNet)/midaBrut;
display(Percentatge_Eliminat);

%Còmput del nombre de punts per quadrícula del mallat.
index=sub2ind([mida,mida],ry,rx);
Z = accumarray(index,1,[mida*mida 1]);
Z = reshape(Z,mida,mida);
```

```
figure;
titol=num2str(mida);
colormap(pink)
dibuix=pcolor(X,Y,sqrt(Z));
xlabel('Theta');ylabel('R');title(titol);
set(dibuix, 'EdgeColor', 'none');
```

A.15- PS_solver

```
function [resuPS , resuPSPol] = PS_solver(ab,nPS,variablePS,vControl,u)
%Resolució de la secció de Poincaré. Troba nPS punts on variablePS es fa
%zero, juntament amb totes les seves coordenades.
format long e; tol = 1e-13;  hmax = 5e-3;
aprox=0;
%Execució del programa
iPS=0; %Nombre de seccions de Poincaré trobades
resuPS=zeros(6,nPS);  resuPSPol=zeros(5,nPS);
loopLimit = 10000;  loop=0; %Nombre màxim de loops. Evita entrar en bucle
infinit.
increment = hmax*10; %Pas d'integració per cada iterat a comprovar
t0=0;
%Primera iteració (des del punt inicial sense comprovació
PSspan=[t0 t0+increment];
[resu] = nInt(ab,PSspan,hmax,tol,u);    [resu] = ham(resu,u);    [resuPol]
= convPol(resu,u);
while iPS<nPS && loop<loopLimit
    loop=loop+1; if mod(loop,500)==0; display(loop); end
    ab=resu(2:5,end);
    t0=resu(1,end);  PSspan=[t0 t0+increment];
    [resu] = nInt(ab,PSspan,hmax,tol,u);    [resu] = ham(resu,u);
[resuPol] = convPol(resu,u);
    %Comprovació de si s'ha arribat a PS per thetaP
    p=resuPol(variablePS,1);  q=resuPol(variablePS,end);
ctrl=resuPol(vControl,1);
    if p*q<0 && ctrl>0
        %Poincaré section
        iPS=iPS+1; if mod(iPS,100)==0; display(iPS); end; loop=0;
        [ coordCart , coordPol , aproximacions ] = BiseccioDef( resu,
resuPol, hmax, variablePS, u );
        aprox=aprox+aproximacions;
        resuPS(:,iPS)=coordCart;  resuPSPol(:,iPS)=coordPol;
    end
end
if aprox~=0
    display(aprox);

beep;pause(0.2);beep;pause(0.2);beep;pause(0.2);beep;pause(0.2);beep;pause(
5);
end
if loop==loopLimit
    resuPS=[]; resuPSPol=[];
end
end
```

A.16- PSexe

```
%PSexe executa PS_solver. Permet variar una variable entre dos punts
especificats.
clear

%'ab' vector fila de les condicions inicials x(1),..,x(n) en cartesianes
per defecte
r = 0.99;      theta = pi/2;    thetaP = 0;    E=-1.494;
aproximacions=0; pTotals=0;
%'variable' indica en quina posició es troba la variable a anul·lar fent PS
%Cartesianes --> t x y xp yp E    Polars --> t r theta rP thetaP
%'vControl' indica quina variable ha de ser >0 per trobar els punts on
variablePS=5;    vControl=4;

ask_Planeta='Escull el planeta a analitzar:\n Júpiter --> j\n Terra --> t\n
Neptú --> n\n';
ask_nPS='Nombre (màxim) de seccions de Poincaré?\n';
ask_nPunts='Quants punts inicials es volen calcular?\n';
askInici='Quin és el primer punt inicial?\n';
askFinal='Quin és lúltim punt inicial?\n';
askVP='Quina variable defineix els punts inicials? (r,theta,thetaP,E)\n';
askObjectiu='Es vol integrar amb límit de temps o de seccions?\n';
askTemps='Tria el temps fins el que es vol integrar\n';
askTitolCart='Escriu el títol del fitxer Cartesianes resultant (amb la
extensió .d o .txt)\n';
askTitolPol='Escriu el títol del fitxer Polars resultant (amb la extensió
.d o .txt)\n';

planeta=input(ask_Planeta,'s'); %Planeta que especifica la u
nPS=input(ask_nPS); %Nombre de seccions de Poincaré
VP=input(askVP,'s'); %Variable que iterarà per tenir diferents punts
inicials
nombrePunts=input(ask_nPunts); %Nombre de punts inicials a partir dels
quals calcular PS
puntInici=input(askInici); %Primer valor dels punts inicials
puntFinal=input(askFinal); %Últim valor dels punts inicials
Objectiu=input(askObjectiu,'s'); %Integrar fins nPS o fins timeLimit
if strcmpi(Objectiu,'temps')==1 || strcmpi(Objectiu,'t')==1;
timeLimit=input(askTemps); %timeLimit
end
titolCartesianes=input(askTitolCart,'s');
titolPolars=input(askTitolPol,'s');
rotulacioCartesianes = ['Temps ',' X ',' Y ',' xPrima ',' yPrima ',' E'];
rotulacioPolars = ['Temps ',' R ',' Theta ',' rPrima ',' ThetaPrima '];
fileID = fopen(titolCartesianes,'w');fprintf(fileID,'%6s %3s %3s %8s %8s
%2s\r\n',rotulacioCartesianes);fprintf(fileID,'\r\n');fclose(fileID);
fileID = fopen(titolPolars,'w');fprintf(fileID,'%20s %15s %15s
%7s\r\n',rotulacioPolars);fprintf(fileID,'\r\n');fclose(fileID);

switch planeta
    case 'j'
        u=9.53875e-4;
    case 't'
        u=3.036e-6;
    case 'n'
        u=5.151e-5;
end
for it1=0:nombrePunts-1
```

```
    switch nombrePunts
        case 1;    VariablePivotant=puntInici;
        otherwise; VariablePivotant=puntInici+(puntFinal-
puntInici)*it1/(nombrePunts-1);
    end
    switch VP
        case 'r';                  ab =
condIni_PtoC(VariablePivotant,theta,thetaP,E,u);
        case 'theta';              ab =
condIni_PtoC(r,VariablePivotant,thetaP,E,u);
        case {'thetaP','thetap'}; ab =
condIni_PtoC(r,theta,VariablePivotant,E,u);
        case {'E','e'};            ab =
condIni_PtoC(r,theta,thetaP,VariablePivotant,u);
    end
    switch Objectiu
        case {'temps','t'};    [resuPS , resuPSPol] =
Tim_PS_solver(ab,timeLimit,nPS,variablePS,vControl,u);
        case {'seccions','s'}; [resuPS , resuPSPol] =
PS_solver(ab,nPS,variablePS,vControl,u);
    end
    display('Nombre de punts inicials realitzats: ');display(it1+1);
    %Escriu els resultats en els fitxers especificats
    fileID = fopen(titolCartesianes,'a');fprintf(fileID,'\r\n %24.12f
%24.12f %24.12f %24.12f %24.12f
%24.12f',resuPS);fprintf(fileID,'\r\n\r\n');fclose(fileID);
    fileID = fopen(titolPolars,'a');fprintf(fileID,'\r\n %24.12f %24.12f
%24.12f %24.12f
%24.12f',resuPSPol);fprintf(fileID,'\r\n\r\n');fclose(fileID);
end
load handel; sound(y,1.5*Fs)
```

A.17- puntsLagrange

```
%Calcula la energia de cada punt de Lagrange
[ funLagrange ] = formulari( 'funLagrange' );
[ funELag ] = formulari( 'funELag' );
[ funPol ] = formulari( 'funPol' );

askPlaneta = 'Quin planeta es vol calcular?\n';
planeta=input(askPlaneta,'s');
switch planeta
    case {'terra','t'}
    cg=3.036e-6;%TERRA
    case {'jupiter','j'}
    cg=9.537e-4;%JUPITER
    case {'mart','m'}
    cg=3.228e-7;%MART
    case {'saturn','s'}
    cg=2.860e-4;%SATURN
    case {'neptu','n'}
    cg=5.151e-5;%NEPTU
end

display(planeta);
coord_xy=feval(funLagrange,cg);
midaXY=size(coord_xy); display(midaXY)
coord_rt=zeros(5,2);
```

```
for i=1:5
    coord_rt(i,:)=feval(funPol,coord_xy(i,:),cg);
    if coord_xy(i,1)+cg<0 && coord_xy(i,2)<=0 %3r Q
        coord_rt(i,2)=coord_rt(i,2)+pi;
    elseif coord_xy(i,1)+cg>0 && coord_xy(i,2)<0 %4t Q
        coord_rt(i,2)=coord_rt(i,2)+2*pi;
    elseif coord_xy(i,1)+cg<0 && coord_xy(i,2)>0 %2n Q
        coord_rt(i,2)=coord_rt(i,2)+pi;
    end
end
midaRT=size(coord_rt); display(midaRT)

E_Lagrange=zeros(length(cg),5);
for i=1:1:length(cg)
    for L=1:1:5
        E_Lagrange(i,L)=feval(funELag,coord_xy(L,:),cg(i));
    end
end
display(coord_xy)
display(coord_rt)
display(E_Lagrange)

askGraficar='Vols graficar els resultats? y/n\n';
askGuardar='Vols guardar els resultats en un fitxer .d? y/n\n';
graficar = input(askGraficar,'s');
guardar = input(askGuardar,'s');
if strcmpi(graficar,'y')
    %Graficat dels punts de Lagrange
    figure;
        s(1)=subplot(1,2,1);
    scatter(coord_xy(:,1),coord_xy(:,2),33,'fill'); hold on
    xlabel('X'); ylabel('Y'); title(s(1),'Coordenades Cartesianes');
    for i=1:5
        px=(coord_xy(i,1)+0.025); py=(coord_xy(i,2)+0.05);
        text(px,py,['L',int2str(i)]);
    end; hold off
        s(2)=subplot(1,2,2);
    scatter(coord_rt(:,2),coord_rt(:,1),33,'fill'); hold on
    xlabel('Theta'); ylabel('R'); title(s(2),'Coordenades Polars');
    for i=1:5
        px=(coord_rt(i,2)+0.05); py=(coord_rt(i,1)+0.005);
        text(px,py,['L',int2str(i)]);
    end; hold off
end
if strcmpi(guardar,'y')
    titolC=([planeta,'_Lagrange_Cart.d']);
titolP=([planeta,'_Lagrange_Pol.d']);
    %'rotulacio' títols de les columnes de 'Cartesianes'
    %'rotulacioP' títols de les columnes de 'Polars'
    rotulacioC = [' X ',' Y ',' E ']; rotulacioP = [' R ',' T ',' E '];
    %Creació i escriptura dels arxius .d finals
    fileID = fopen(titolC,'w');
    fprintf(fileID,'%3s %3s %3s\r\n',rotulacioC);  fprintf(fileID,'\r\n');
    fprintf(fileID,'\r\n %24.12f %24.12f %24.12f',[coord_xy,E_Lagrange']');
    fclose(fileID);

    fileID = fopen(titolP,'w');
    fprintf(fileID,'%3s %3s %3s\r\n',rotulacioP);  fprintf(fileID,'\r\n');
    fprintf(fileID,'\r\n %24.12f %24.12f %24.12f',[coord_rt,E_Lagrange']');
    fclose(fileID);
end
```

A.18- Tim_PS_solver

```
function [resuPS , resuPSPol] =
Tim_PS_solver(ab,timeLimit,nPS,variablePS,vControl,u)
%Resolució de la secció de Poincaré. Troba nPS punts on variablePS es fa
%zero, juntament amb totes les seves coordenades.
format long e; tol = 1e-13;  hmax = 5e-3;
aprox=0;
%Execució del programa
iPS=0; %Nombre de seccions de Poincaré trobades
resuPS=zeros(6,nPS);  resuPSPol=zeros(5,nPS);
loopLimit = 2000;  loop=0; %Nombre màxim de loops. Evita entrar en bucle
infinit.
increment = hmax*10; %Pas d'integració per cada iterat a comprovar
t0=0;
%Primera iteració (des del punt inicial sense comprovació
PSspan=[t0 t0+increment];
[resu] = nInt(ab,PSspan,hmax,tol,u);    [resu] = ham(resu,u);    [resuPol]
= convPol(resu,u);
while t0<timeLimit && loop<loopLimit
    loop=loop+1; if mod(loop,500)==0; display(loop); end
    ab=resu(2:5,end);
    t0=resu(1,end);
    if abs(t0+increment)<abs(timeLimit);  PSspan=[t0 t0+increment];
    else PSspan=[t0 timeLimit];
    end
    [resu] = nInt(ab,PSspan,hmax,tol,u);    [resu] = ham(resu,u);
[resuPol] = convPol(resu,u);
    %Comprovació de si s'ha arribat a PS per thetaP
    p=resuPol(variablePS,1);  q=resuPol(variablePS,end);
ctrl=resuPol(vControl,1);
    if p*q<0 && ctrl>0
        %Poincaré section
        iPS=iPS+1; if mod(iPS,100)==0; display(iPS); end; loop=0;
        [ coordCart , coordPol , aproximacions ] = BiseccioDef( resu,
resuPol, hmax, variablePS, u );
        aprox=aprox+aproximacions;
        resuPS(:,iPS)=coordCart;  resuPSPol(:,iPS)=coordPol;
    end
end
if aprox~=0
   display(aprox);
end
resuPS( :, ~any(resuPS,1) ) = [];  %elimina columnes no utilitzades
resuPSPol( :, ~any(resuPSPol,1) ) = [];  %elimina columnes no utilitzades
if loop==loopLimit
    resuPS=[]; resuPSPol=[];
end
end
```

A.19- Tim_PS_solver_JT

```
function [resuPS , resuPSPol] =
Tim_PS_solver_JT(ab,timeLimit,nPS,variablePS,vControl,u,fun,tolerancies,dir
eccioTemp)
```

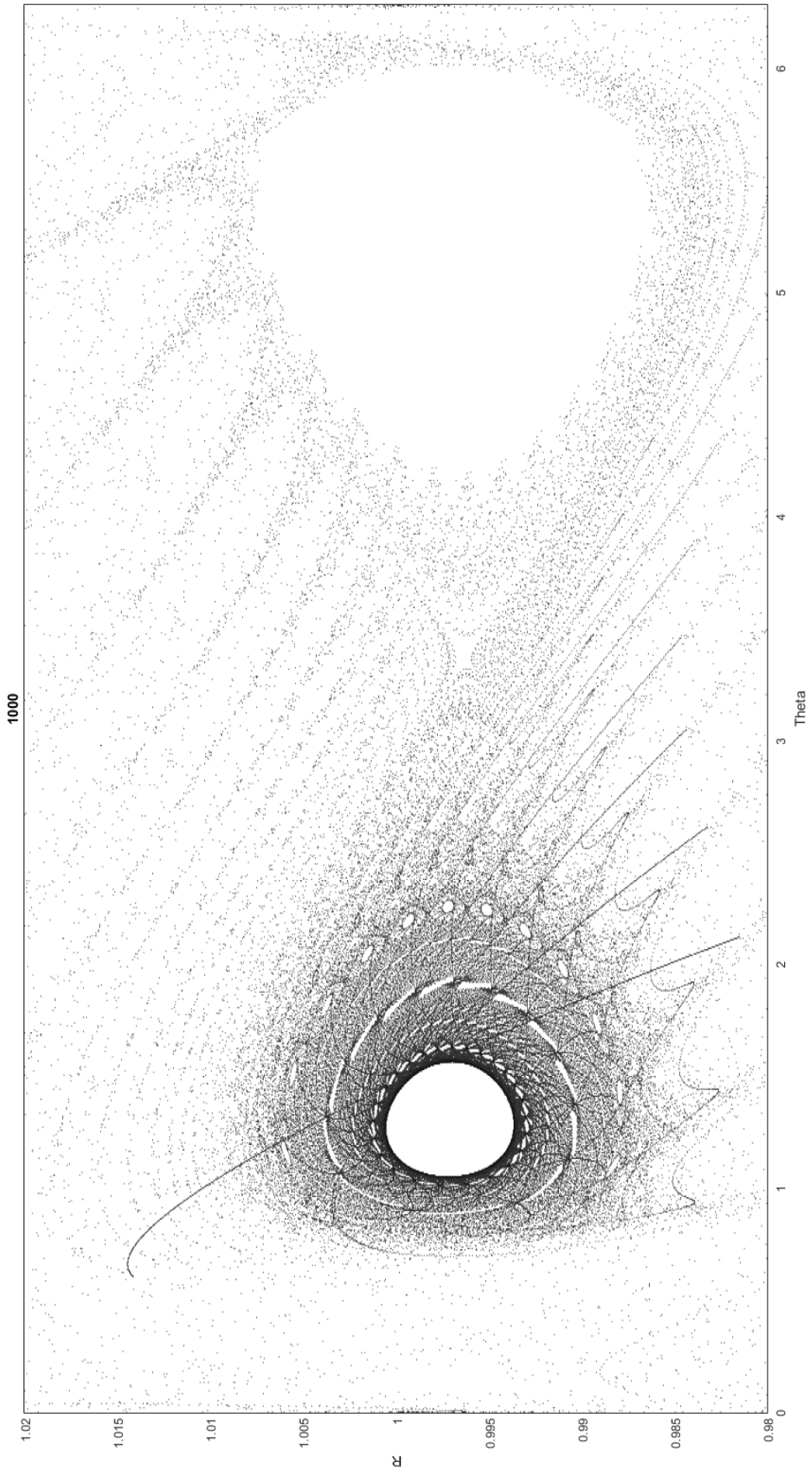
```
%Resolució de la secció de Poincaré. Troba nPS punts on variablePS es fa
%zero, juntament amb totes les seves coordenades.
format long e;
aprox=0;  hmax=5e-3;
%Execució del programa
iPS=0; %Nombre de seccions de Poincaré trobades
resuPS=zeros(6,nPS);  resuPSPol=zeros(5,nPS);
loopLimit = 2000;  loop=0; %Nombre màxim de loops. Evita entrar en bucle
infinit.
%Pas d'integració per cada iterat a comprovar
switch direccioTemp
    case 'P'
        increment = -hmax*10;
    case 'F'
        increment = hmax*10;
end
t0=0;
%Primera iteració (des del punt inicial sense comprovació
PSspan=[t0 t0+increment];
[T,Y]=ode45(fun,PSspan,ab,tolerancies); resu=[T,Y]';
    [resu]=ham(resu,u);
while abs(t0)<abs(timeLimit) && loop<loopLimit
    loop=loop+1; if mod(loop,250)==0; display(loop); end
    ab=resu(2:5,end);
    t0=resu(1,end);
    if abs(t0+increment)<abs(timeLimit);  PSspan=[t0 t0+increment];
    else PSspan=[t0 timeLimit];
    end
    [T,Y]=ode45(fun,PSspan,ab,tolerancies); resu=[T,Y]';
    [resu]=ham(resu,u);  [resuPol]=convPol(resu,u);
    %Comprovació de si s'ha arribat a PS per thetaP
    p=resuPol(variablePS,1);  q=resuPol(variablePS,end);
ctrl=resuPol(vControl,1);
    if p*q<0 && ctrl>0
        %Poincaré section
        iPS=iPS+1; display(iPS);  loop=0;

[coordCart,coordPol,aproximacions]=BiseccioDef_JT(resu,resuPol,variablePS,u
,fun,tolerancies);
        aprox=aprox+aproximacions;
        resuPS(:,iPS)=coordCart;  resuPSPol(:,iPS)=coordPol;
    end
end
if aprox~=0
   display(aprox);
end
resuPS( :, ~any(resuPS,1) ) = [];  %elimina columnes no utilitzades
resuPSPol( :, ~any(resuPSPol,1) ) = [];  %elimina columnes no utilitzades
if loop==loopLimit
    resuPS=[]; resuPSPol=[];
end
end
```

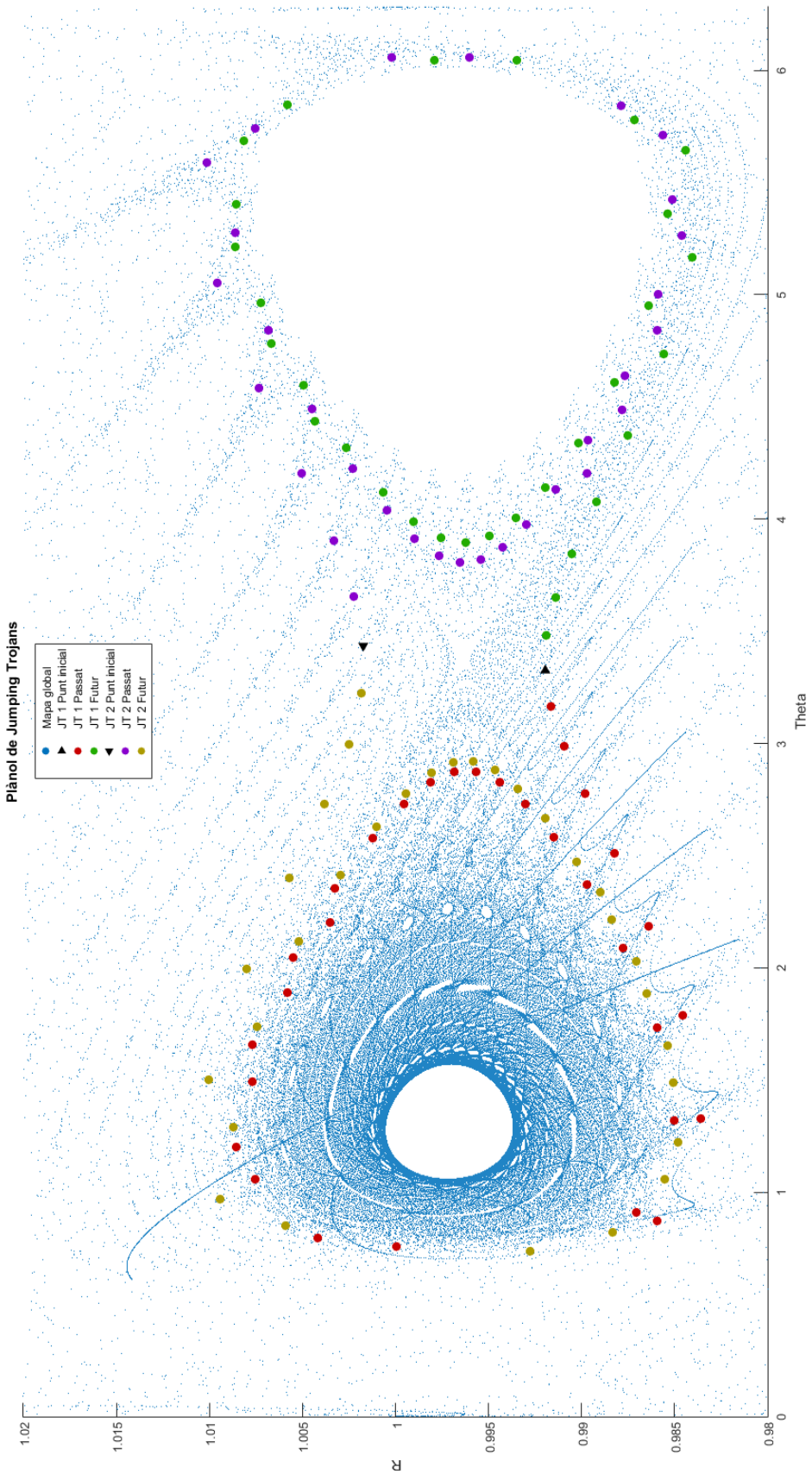
B- Figures

A continuació s'adjunten, en format ampliat, les figures més complexes del capítol 5.

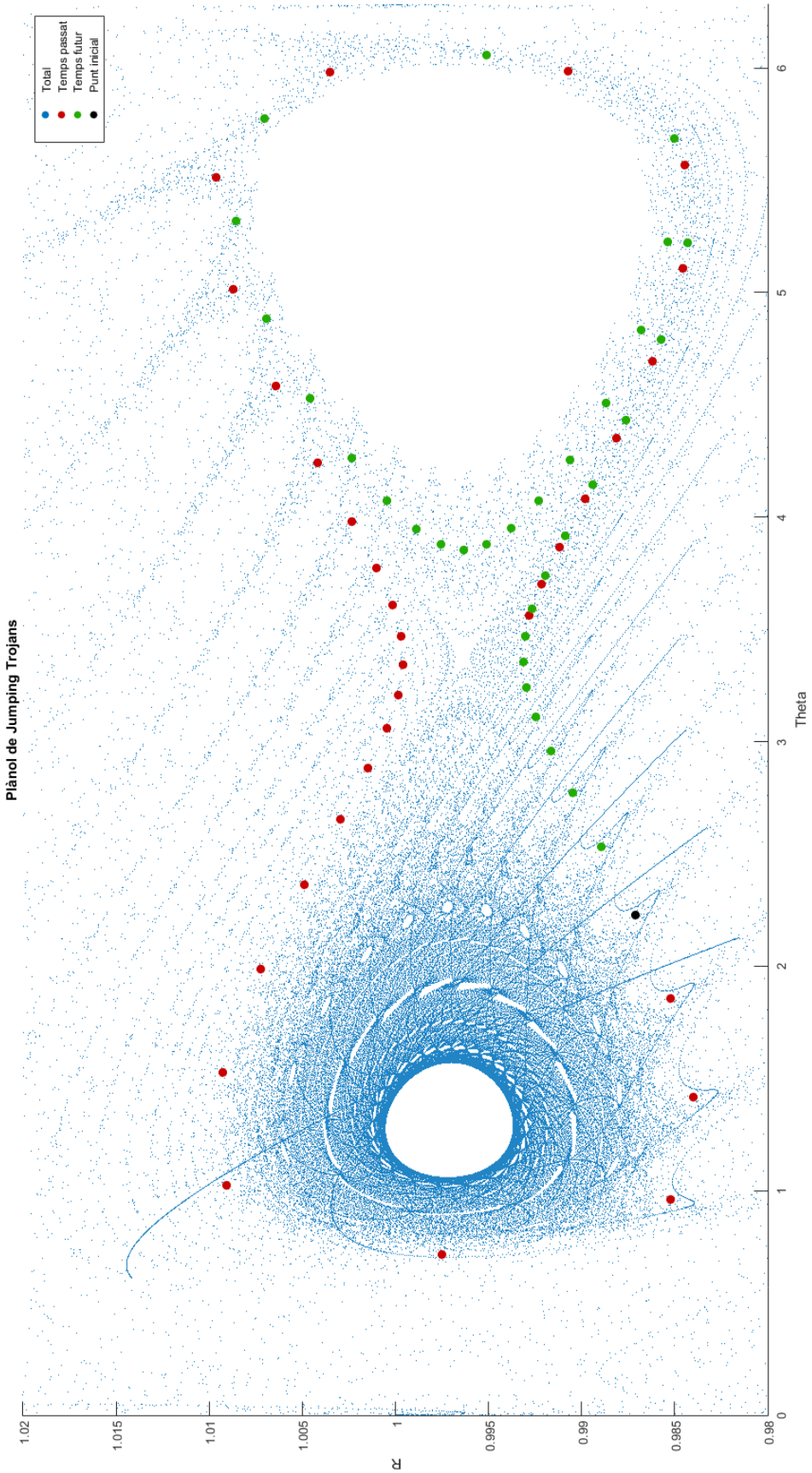
B.1- Figura 20



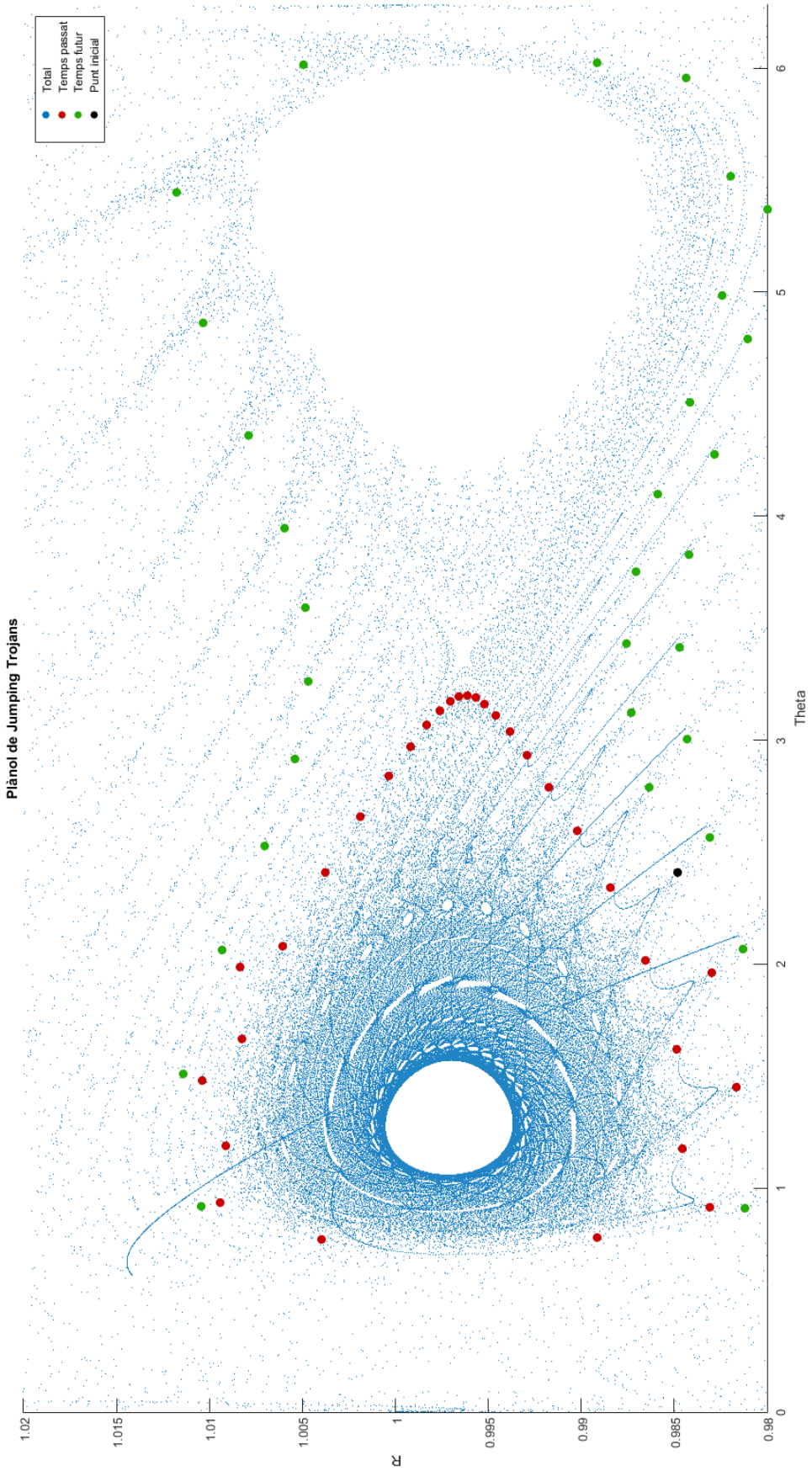
B.2- Figura 23


Plànol de Jumping Trojans
Mapa global
JT 1 Punt inicial
JT 1 Passat
JT 1 Futur
JT 2 Punt inicial
JT 2 Passat
JT 2 Futur
r
Theta
1.02
1.015
1.01
1.005
1
0.995
0.99
0.985
0.98
0
1
2
3
4
5
6

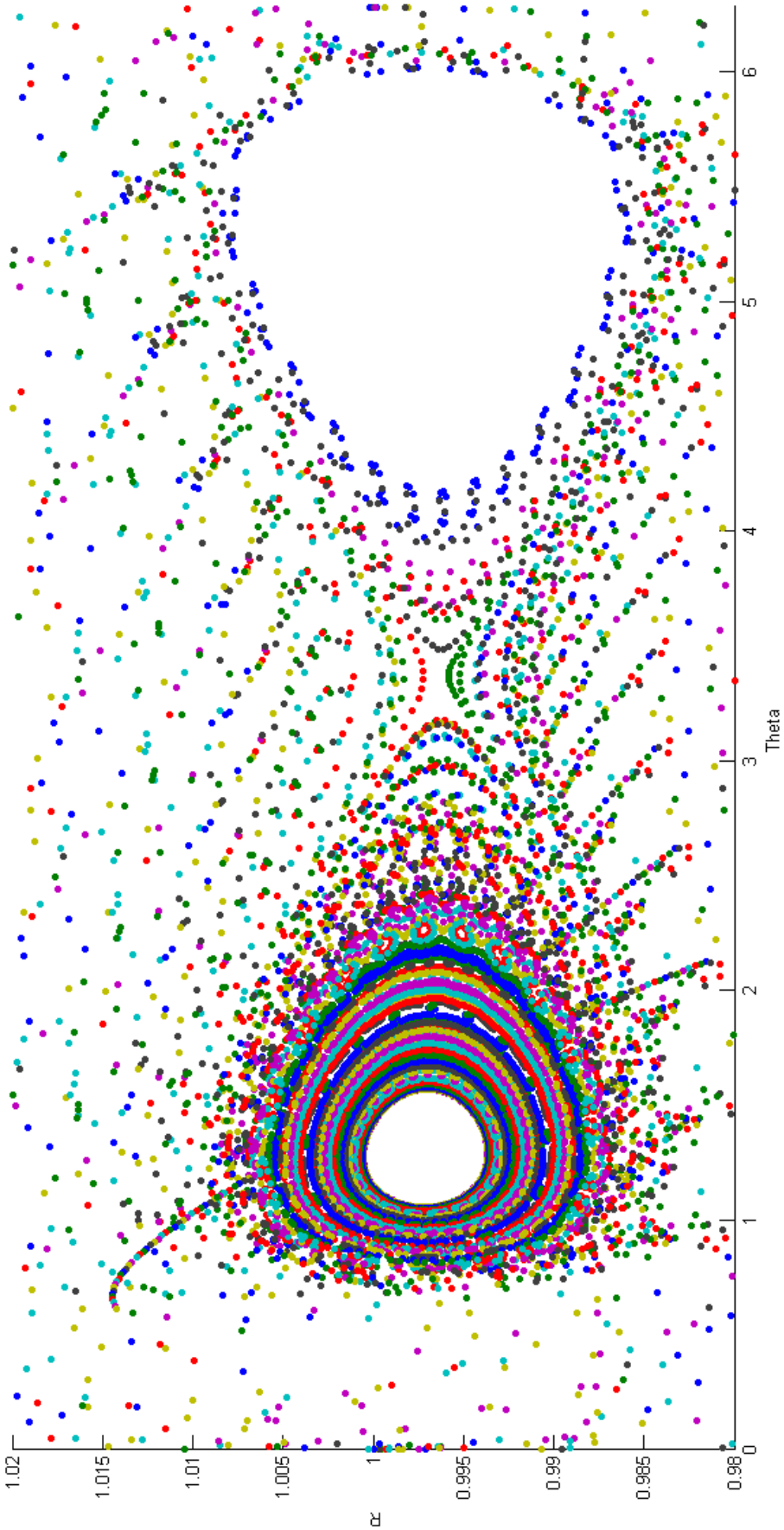
B.3- Figura 24


Plànol de Jumping Trojans
Total
Temps passat
Temps futur
Punt inicial
R
Theta
1.02
1.015
1.01
1.005
1
0.995
0.99
0.985
0.98
0
1
2
3
4
5
6

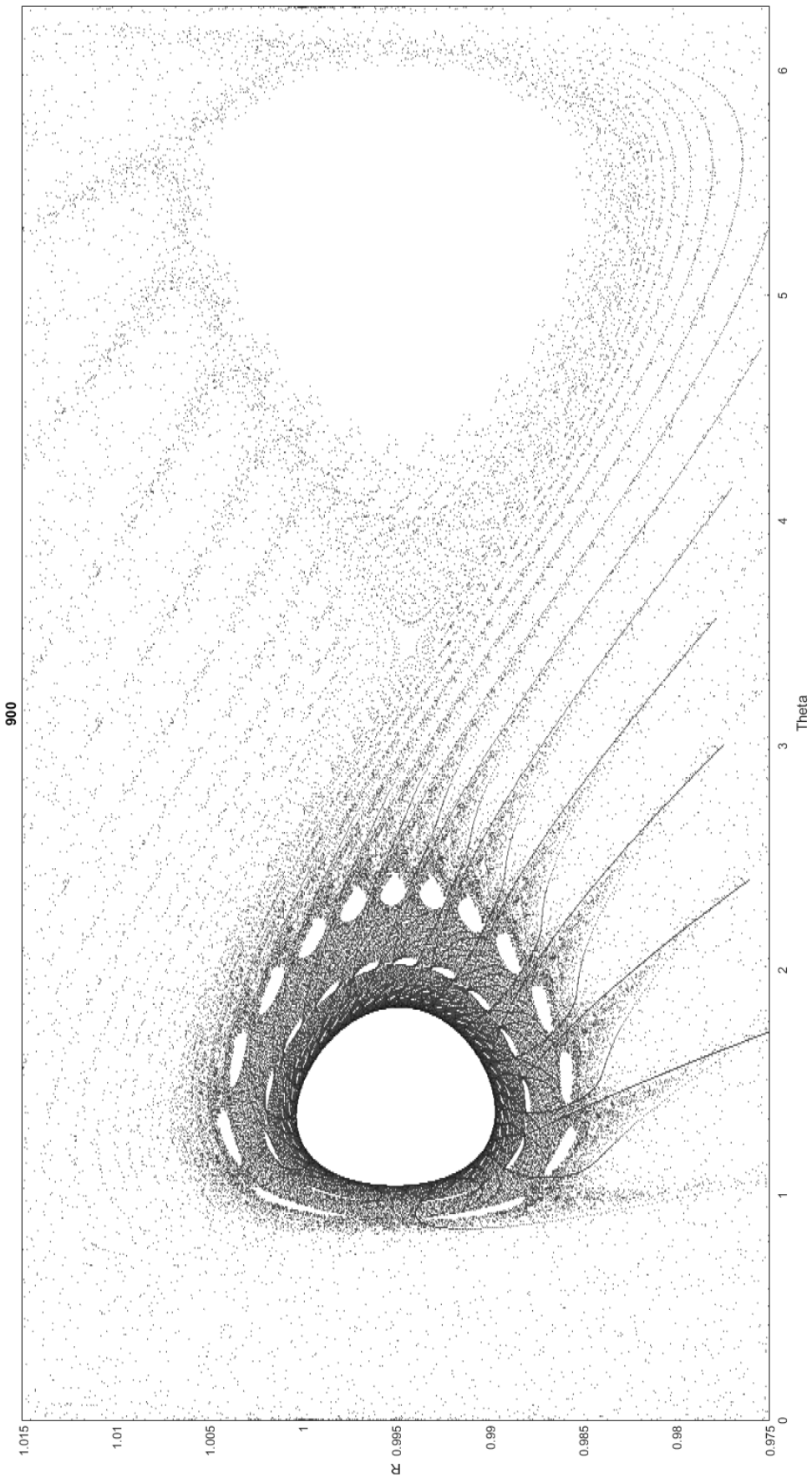
B.4- Figura 25


Plànol de Jumping Trojans
Total
Temps passat
Temps futur
Punt inicial
R
Theta
1.02
1.015
1.01
1.005
1
0.995
0.99
0.985
0.98
0
1
2
3
4
5
6

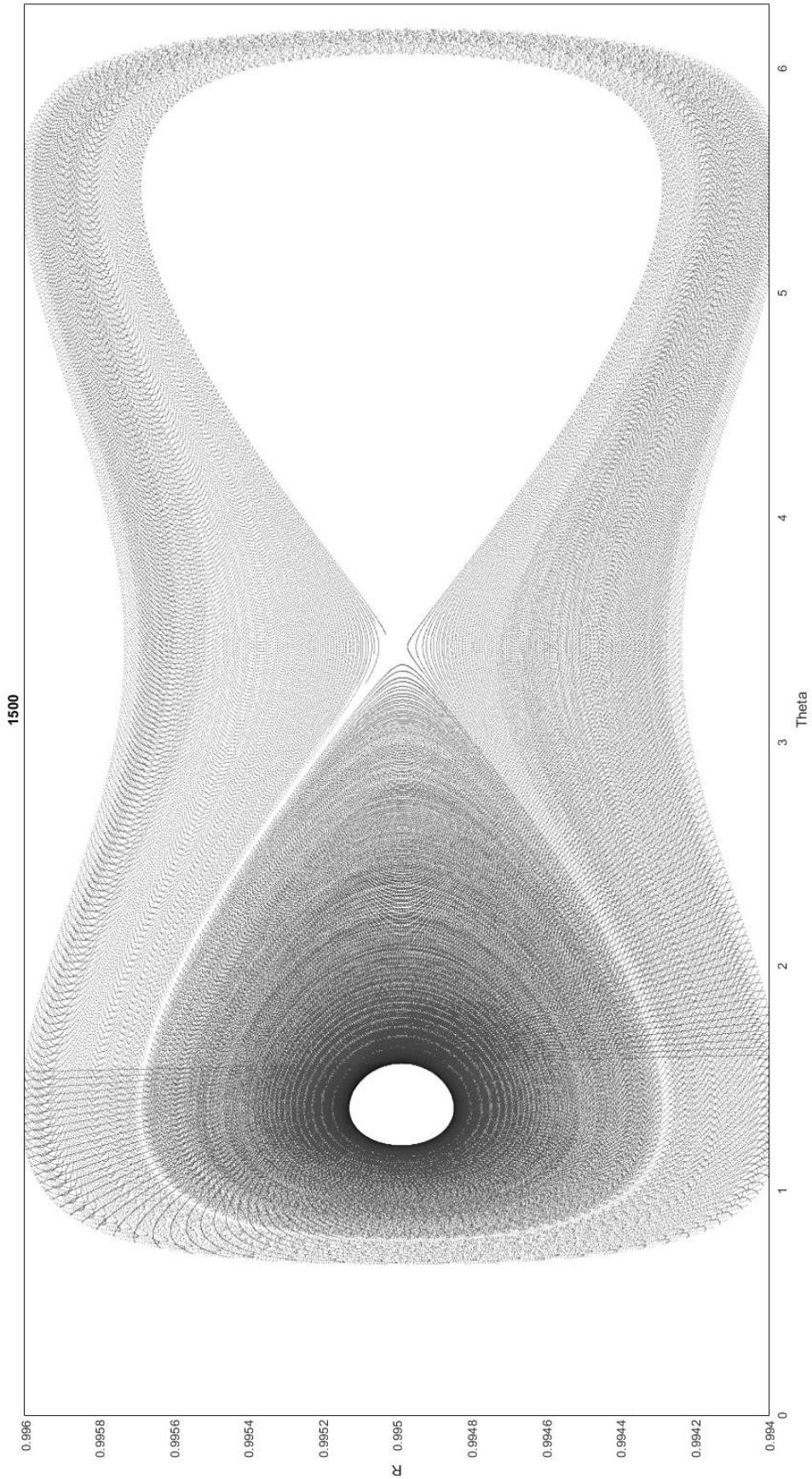
B.5- Figura 27



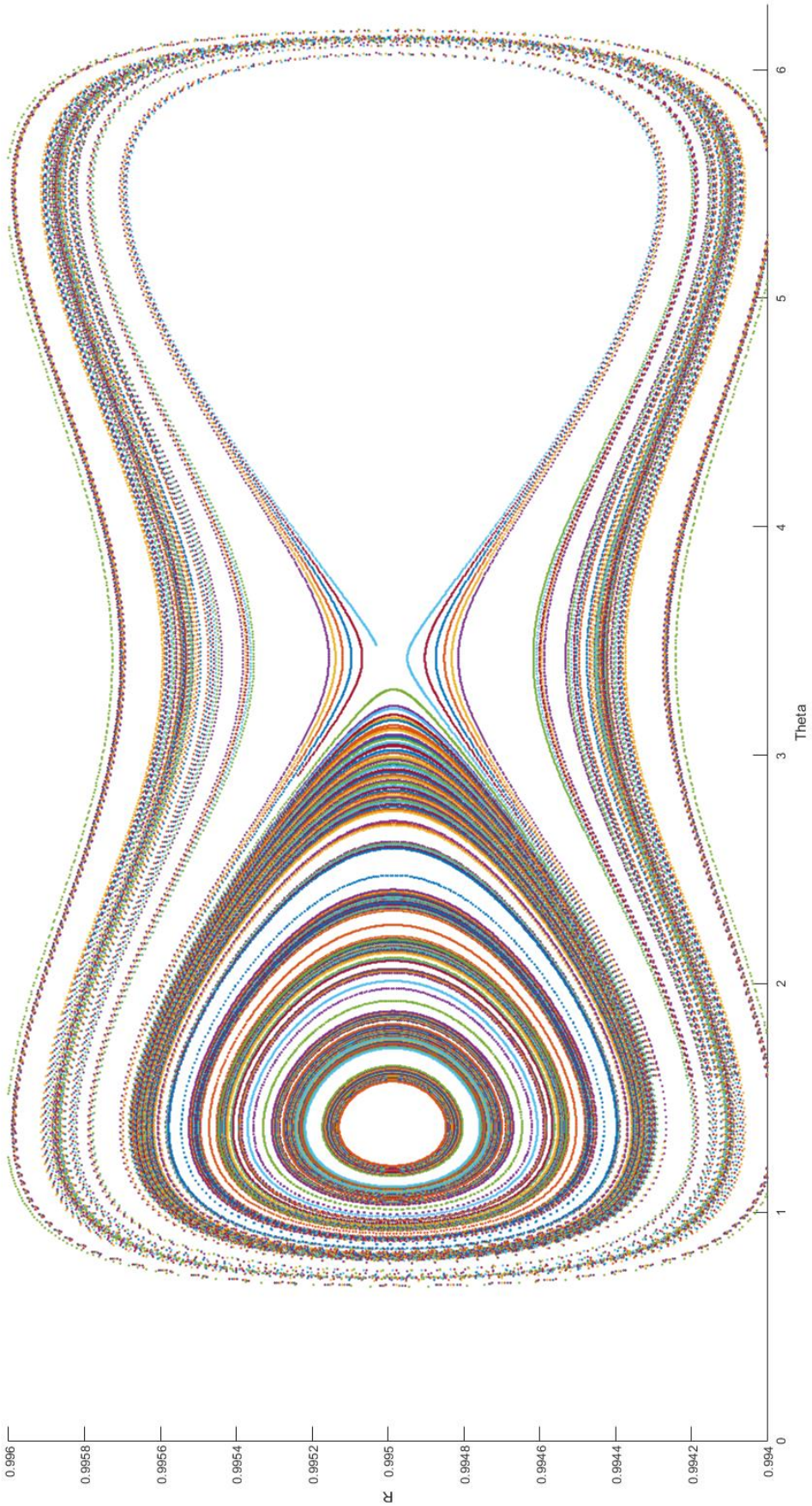
B.6- Figura 28



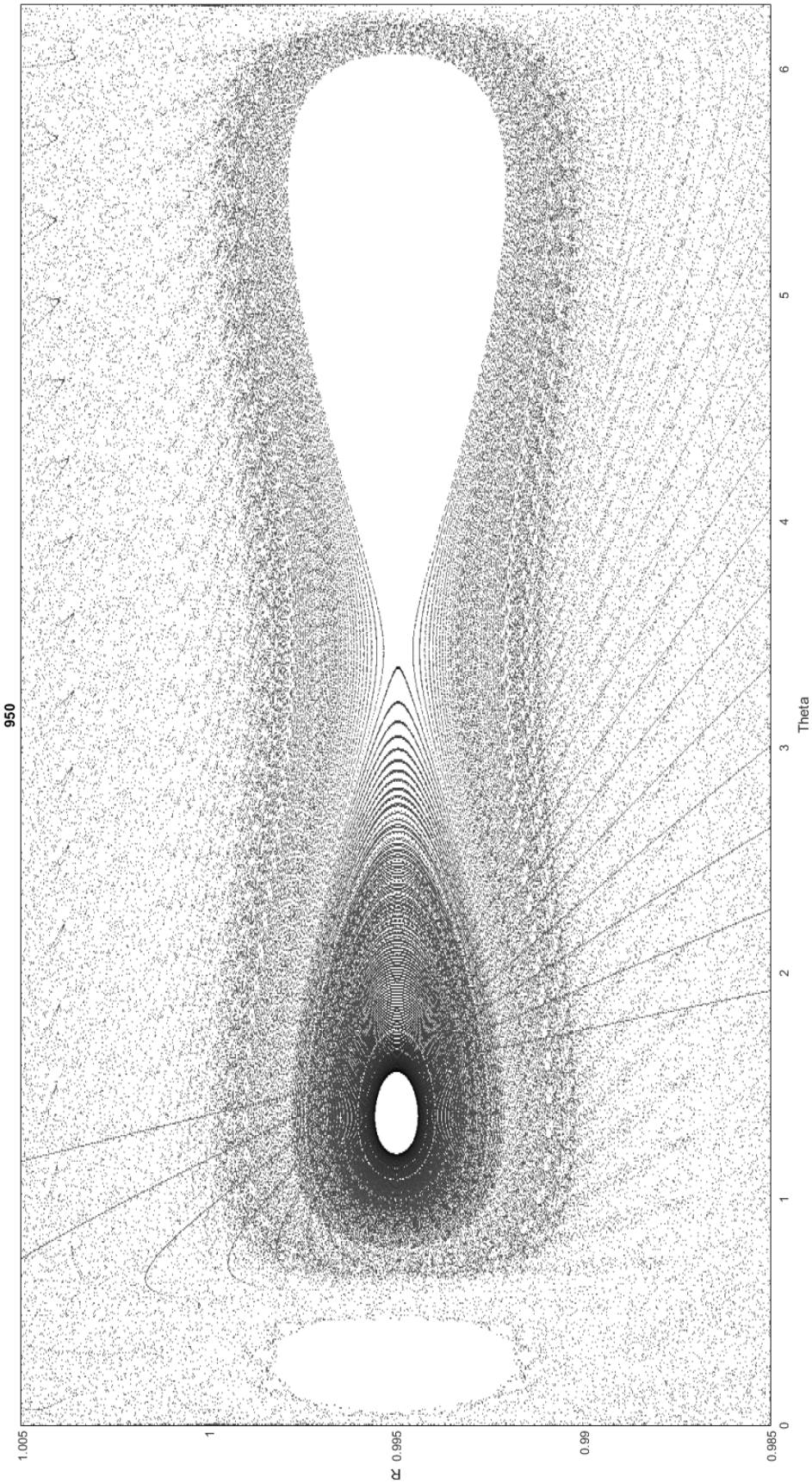
B.7- Figura 29


1500
0.996
0.9958
0.9956
0.9954
0.9952
0.995
0.9948
0.9946
0.9944
0.9942
0.994
α
0
1
2
3
4
5
6
Theta

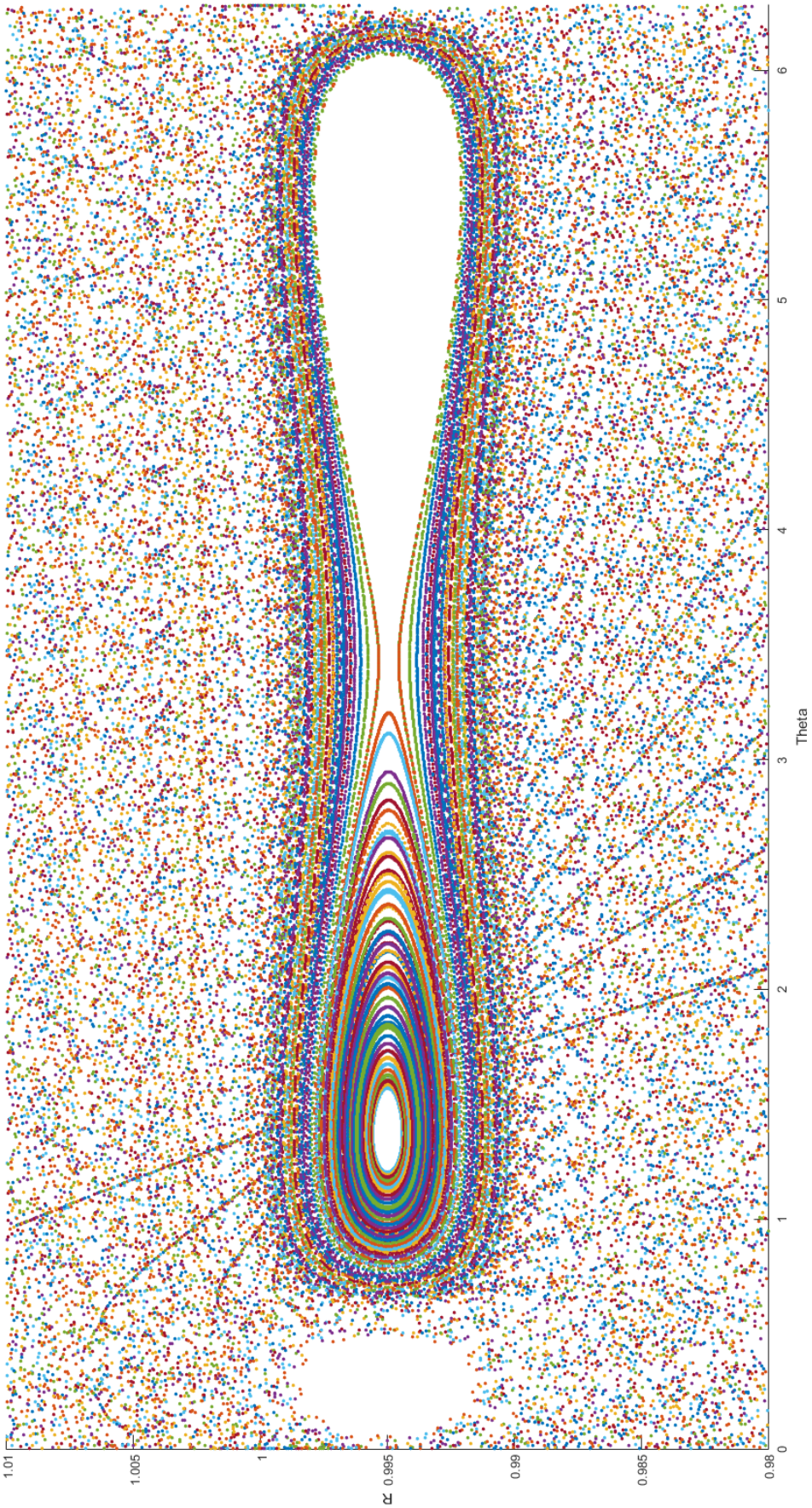
B.8- Figura 30


0.996
0.9958
0.9956
0.9954
0.9952
0.995
0.9948
0.9946
0.9944
0.9942
0.994
α
0
1
2
3
4
5
6
Theta

B.9- Figura 31



B.10- Figura 32



C- Pressupost

El pressupost desglossat es troba a la taula següent:

	Cost horari (€/h)	Hores	Import (€)
Recerca bibliogràfica	70,20	40	2.808,00
Programació de codi	70,20	450	31.590,00
Elaboració de documents	70,20	126	8.845,20
Amortització d'equips informàtics	0,90	1.248	1.123,20
Amortització de software	0,50	1.122	561,00
		Total	**44.927,40**

Cost horari de mà d'obra corresponent a la contractació d'un enginyer informàtic (oficial de 1ª) extret del base de dades BEDEC.