

Computation and Visualization of Invariant Manifolds

Zhikai Wang

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montréal, Québec, Canada

October 2009

© Zhikai Wang, 2009



Library and Archives
Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file *Votre référence*
ISBN: 978-0-494-67284-6
Our file *Notre référence*
ISBN: 978-0-494-67284-6

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.


Canada

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Zhikai Wang**

Entitled: **Computation and Visualization of Invariant Manifolds**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards
with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. N. Shiri

_____ Examiner
Dr. A. Krzyzak

_____ Examiner
Dr. B. Oldeman

_____ Supervisor
Dr. E. Doedel

Approved by _____
Chair of Department or Graduate Program Director

_____ 20 _____
Dr. Robin A. L. Drew, Dean
Faculty of Computer Science and Software Engineering

Computation and Visualization of Invariant Manifolds

Zhikai Wang

October 20, 2009

Abstract

Computation and Visualization of Invariant Manifolds

Zhikai Wang

In this thesis, we start with the basic concepts of dynamical systems. Then we introduce the general types of problems that the well-known software package AUTO solves. AUTO uses a boundary value algorithm with Gauss collocation and pseudo-arclength continuation. These two features distinguish AUTO from other general ODE solvers for dynamical systems. In order to compute 2D solution manifolds, AUTO uses orbit continuation. With these tools, we study two famous problems, the Lorenz system and the Circular Restricted Three-Body Problem (CR3BP). We briefly discuss the basic bifurcation and stability analysis of general ODE systems. The numerical analysis of the two problems leads to the newest algorithm to compute the 2D stable manifold of the origin of the Lorenz system and the 2D unstable manifold of appropriate periodic orbits of the CR3BP. We utilize Python for the flow control of AUTO. We also implement two visualization packages, QTPlaut and MATPlaut. They make possible the processing of large quantities of AUTO solution data with the OpenGL graphical library, dynamic memory allocation and interpolation methods. We conclude with prospect for future research.

Acknowledgments

I would like to express my sincere appreciation to my supervisor Professor Eusebius J. Doedel, for his guidance, support, documents, and patience, for completing my thesis. Life is dynamic. I should get hints from Eusebius' instructions. The instructions are about trajectories, stability and methods.

I would also thank my parents for their love, understanding and support.

Contents

Contents	v
List of Figures	ix
List of Tables	xiv
1 Introduction	1
1.1 Dynamical systems	1
1.2 Differential equations and the problems AUTO solves	2
1.3 Stable and unstable invariant manifolds	3
1.4 Numerical methods for the computation of stable and unstable manifolds	4
1.5 Different approaches for computing 2D stable/unstable manifold	5
1.6 Graphical methods to visualize AUTO solutions	7
1.7 Organization of the thesis	8
2 Orbit Continuation	9
2.1 Stability of equilibria	9
2.2 Continuation of solutions	11
2.2.1 Regular solutions	11
2.2.2 Parameter continuation	11
2.2.3 Keller's pseudo-arclength continuation	13
2.3 Boundary value problems	14
2.3.1 Orthogonal collocation	15
2.4 Implementation of the orthogonal collocation method	17

2.5	Orbit continuation	18
2.5.1	Example 1, AUTO demo um2	20
2.5.2	Example 2, AUTO demo um3	22
2.5.3	Discussion	22
3	Elementary Bifurcation Concepts	28
3.1	Bifurcation and bifurcation diagrams	28
3.2	Hopf bifurcation	31
3.3	Homoclinic and heteroclinic orbits	33
4	Computing the Lorenz Manifold	35
4.1	Introduction	35
4.1.1	Formalization of the system	35
4.1.2	Elementary analysis	36
4.2	Previous work	40
4.3	Computing the stable manifold of the origin	41
4.3.1	The method for computing the stable manifold of the origin	41
4.3.2	Numerical results	44
4.4	Locating heteroclinic connections in the Lorenz manifold	44
4.4.1	The existence of homoclinic orbits and heteroclinic orbits	45
4.4.2	The continuation procedure	45
4.4.3	The symbol sequence of the heteroclinic connections	47
5	The Stability of Periodic Solutions	62
5.1	Periodic solutions of autonomous systems	62
5.2	The monodromy matrix	63
5.3	The Poincaré map	65
5.4	Mechanism of losing stability for periodic orbits	70
5.4.1	Branch points of periodic solutions	71
5.4.2	Period doubling	74
5.4.3	Bifurcation to a torus	76

6	The Circular Restricted Three-Body Problem	78
6.1	Introduction	78
6.2	The circular restricted Three-Body Problem	80
6.3	Computing periodic orbits with AUTO	83
6.4	The CR3BP implementation in AUTO	85
6.5	Some facts about the CR3BP	86
6.6	Computing unstable manifolds and heteroclinic orbits of the CR3BP with AUTO	87
6.6.1	The general procedure	87
6.6.2	Results	88
7	Conclusions and Prospects	108
	Bibliography	110
A	Computing Heteroclinic Connections of the Lorenz Manifold	120
A.1	The equation file man.f	120
A.2	Computing the candidate orbits	129
A.2.1	File start.auto	129
A.2.2	Constants file c.man.1	130
A.2.3	Constants file c.man.2	130
A.3	Computing the candidate orbits	131
A.3.1	Script run.auto	131
A.3.2	Utility file count.f	134
A.3.3	Constants file c.man.3	137
A.3.4	Constants file c.man.4	137
A.4	Checking if the symbol sequences are complete	139
A.4.1	Python script mlvl.py (max level)	139
A.4.2	Record the sequences	141
A.4.3	AUTO script trace.auto	141
A.5	Supplement missing sequences manually	145

A.5.1	Constants file c.man.21	145
A.6	Extract the connections to one solution	146
A.6.1	AUTO script extract.auto	146
A.6.2	Constants file c.man.32	147
A.6.3	Constants file c.man.42	148
B	Computing CR3BP Unstable Manifold	149
B.1	The general procedure	149
B.2	Scripts	150
B.2.1	AUTO script work.auto	150

List of Figures

2.1	Basic equilibria types in R^2	12
2.2	Graphical interpretation of parameter continuation	13
2.3	Graphical interpretation of pseudo-arclength continuation	13
2.4	The mesh $\{ 0 = t_0 < t_1 < \dots < t_N = 1 \}$	16
2.5	AUTO Demo um2: $\epsilon = 1.0$	24
2.6	AUTO Demo um2: $\epsilon = 0.5$	24
2.7	AUTO Demo um2: $\epsilon = 0.05$	25
2.8	AUTO Demo um2: $\epsilon = 0.005$; Runge-Kutta $\Delta t = 0.001$	25
2.9	AUTO Demo um2: $\epsilon = 0.005$; Runge-Kutta $\Delta t = 0.0001$	26
2.10	AUTO Demo um3 $\epsilon = 1.0$	26
2.11	AUTO Demo um3 $\epsilon = 0.5$	27
2.12	AUTO Demo um3 $\epsilon = 0.05$	27
3.1	An example bifurcation diagram	30
3.2	The path of eigenvalues related to the Hopf bifurcation	32
3.3	Heteroclinic and homoclinic orbits	33
3.4	A homoclinic bifurcation at λ_0	34
4.1	Bifurcation diagram of the Lorenz system	38
4.2	The Lorenz manifold with different ρ	50
4.3	Comparing the Lorenz manifolds for $\rho = 15$ and $\rho = 60$	51
4.4	Detecting a heteroclinic orbit	52
4.5	The path of candidate orbits' end points	53

4.6	1024 heteroclinic orbits	54
4.7	Selected heteroclinic orbits	55
4.8	The relation between symbol sequences and their lengths	56
4.9	Scaled orbit indices <i>vs.</i> time	57
4.10	Scaled orbit indices <i>vs.</i> L_2 norm	58
4.11	Scaled orbit indices <i>vs.</i> θ	59
4.12	Scaled orbit indices <i>vs.</i> arclength	60
4.13	Scaled orbit indices <i>vs.</i> $(L_{nom} - L) \times T - T_{nom} $	61
5.1	A periodic trajectory \mathbf{u}^* and a neighboring trajectory	63
5.2	An example Poincaré map	65
5.3	Two trajectories intersecting a Poincaré section Ω	68
5.4	Three Poincaré sections	69
5.5	Multipliers (eigenvalues of the monodromy matrix) for three different values of λ	70
5.6	Three ways multipliers leave the unit circle	71
5.7	Illustration of a pitchfork bifurcation in a Poincaré section	72
5.8	Close to a pitchfork bifurcation; corresponding to λ_2 in Figure (5.7)	73
5.9	Close to a fold	73
5.10	Phase planes illustrating how a stable orbit collapses at a fold for λ_0	74
5.11	Close to period doubling	75
5.12	A “Torus” trajectory encircles an unstable periodic orbit	76
6.1	An unstable manifold of a periodic orbit in the family \mathbf{H}_1 at energy -1.465584	89
6.2	An unstable manifold of a periodic orbit in the family \mathbf{H}_1 at energy -1.673174	89
6.3	An unstable manifold of a periodic orbit in the family \mathbf{V}_1 at energy -1.660129	90
6.4	An unstable manifold of a periodic orbit in the family \mathbf{V}_1 at energy -1.507751	90
6.5	A heteroclinic connection from a periodic orbit in family \mathbf{H}_1 to a torus at equilibrium L_3 at energy -1.465585	91
6.6	A heteroclinic connection from a periodic orbit in family \mathbf{H}_1 to a torus at equilibrium L_3 at energy -1.673190	92

6.7	A homoclinic connection from a periodic orbit in family \mathbf{H}_1 to \mathbf{H}_1 at energy -1.673174	92
6.8	A heteroclinic connection from a periodic orbit in family \mathbf{V}_1 to a torus at equilibrium L_1 at energy -1.507750	93
6.9	A heteroclinic connection from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_3 at energy -1.520881	93
6.10	A heteroclinic connection from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_2 at energy -1.506991	94
6.11	A heteroclinic connection from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_3 at energy -1.430828	94
6.12	An unstable manifold of a periodic orbit in the family \mathbf{L}_1 at energy -1.766528	95
6.13	A homoclinic connection from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_1 at energy -1.764280	95
6.14	An unstable manifold containing homoclinic connections from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_1 around the Moon at energy -1.764280	96
6.15	An unstable manifold containing homoclinic connections from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_1 at energy -1.761132	96
6.16	An unstable manifold containing homoclinic connections from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_1 at energy -1.761125	97
6.17	A homoclinic connection from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_1 around the moon at energy -1.754963	97
6.18	An unstable manifold containing homoclinic connections from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_1 at energy -1.754963	98
6.19	An unstable manifold containing homoclinic connections from a periodic orbit in family \mathbf{L}_1 to \mathbf{L}_1 at energy -1.749074	98
6.20	A heteroclinic connection from a periodic orbit in family \mathbf{L}_2 to \mathbf{L}_1 at energy -1.724892	99
6.21	A heteroclinic connection from a periodic orbit in family \mathbf{L}_2 to \mathbf{L}_1 at energy -1.708555	99

6.22	A heteroclinic connection from a periodic orbit in family \mathbf{L}_2 to \mathbf{L}_1 at energy -1.691739	100
6.23	An unstable manifold containing homoclinic connections from a periodic orbit in family \mathbf{L}_2 to \mathbf{L}_2 around the Moon at energy -1.691739	100
6.24	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.559424	101
6.25	An unstable manifold containing heteroclinic connections from a periodic orbit in family \mathbf{L}_3 to family \mathbf{L}_1 at energy -1.559424	101
6.26	An unstable manifold containing heteroclinic connections from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_1 at energy -1.548840	102
6.27	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.548840	102
6.28	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.548840	103
6.29	An unstable manifold containing homoclinic connections from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.548840	103
6.30	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.506991	104
6.31	A heteroclinic connection from a periodic orbit in the family \mathbf{L}_3 to \mathbf{L}_2 at energy -1.506991	104
6.32	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.453254	105
6.33	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.453254	105
6.34	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.430830	106
6.35	A heteroclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_1 at energy -1.506991	106
6.36	A homoclinic connection from a periodic orbit in family \mathbf{L}_3 to \mathbf{L}_3 at energy -1.506991	107

6.37 A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.506991	107
--	-----

List of Tables

2.1	Classification of equilibrium points	11
6.1	Some abbreviations used in the CR3BP	86

Chapter 1

Introduction

In this chapter, we give some introductory concepts of dynamical systems. Then we discuss how AUTO is applied to some problems of dynamical systems. We also present graphical methods for visualizing AUTO results. An outline of the thesis is at the end of this chapter.

1.1 Dynamical systems

A dynamical system is the mathematical formalization of a deterministic process. The future and past states of many systems can be derived to some degree by knowing their present state and the governing laws. It is formed by three elements, the state space, the time set and a family of evolution operators. The *state space* is the set of points characterizing all possible states of the system. The *evolution operators* have two natural properties. First, the system does not change its state “spontaneously”. Second, the law governing the behavior of the system does not change in time, namely, the system is “autonomous”. An *orbit* is an ordered subset of the state space. It starts from a given point in the state space and has a corresponding point for all elements in the time set. An orbit is also called a *trajectory*; see [108].

In a global view, the atomicity of AUTO computations is orbit wise. Thus in this thesis, the simplification or optimization of visualizing AUTO solutions will operate upon orbits. They can be the reduction of actual orbits or interpolation operations on actual orbits.

An *equilibrium* or a *fixed point* is a point such that for all elements in the time set, the

point will be mapped to itself by the evolution operator. A *cycle* or a *periodic orbit* is an orbit such that after a certain time, each point in the orbit will return to itself.

The equilibria and the periodic orbits are like the outstanding landmarks or major routes of a map when we are presenting a mathematically formalized dynamical system graphically.

1.2 Differential equations and the problems AUTO solves

A *differential equation* is an equation involving derivatives. The differential equation's order is the highest order of the derivative. For example, we have the *first order equation*

$$x' = f(t, x),$$

where $x' \equiv \frac{dx}{dt}$, and $f(t, x)$ is a continuous function of t and x . Many higher order differential equations can be expressed as a system of first order equations. A system of n first order differential equations can be described by

$$\mathbf{u}'(t) = \mathbf{f}(t, \mathbf{u}), \quad \mathbf{u} \in \mathbb{R}^n, t \in \mathbb{R}, \quad (1.1)$$

where \mathbb{R} is the set of real numbers. If \mathbf{f} is linear then the system is *linear*, otherwise *nonlinear*. The dynamical systems we deal with are usually defined by n *ordinary differential equations*. ODEs for short. A differentiable function $\mathbf{u} = \mathbf{u}(t)$, $\mathbf{u} \in \mathbb{R}^n$ is the *solution* of the system if $\mathbf{u}(t)$ satisfies $\mathbf{u}'(t) = \mathbf{f}(t, \mathbf{u}(t))$. If \mathbf{f} has no explicit dependence on t , then the ODE system

$$\mathbf{u}' = \mathbf{f}(\mathbf{u}) \quad (1.2)$$

is called autonomous; see [54] and [55].

The software AUTO can do a limited bifurcation analysis of algebraic systems

$$\mathbf{f}(\mathbf{u}, \mathbf{p}) = \mathbf{0}, \quad (1.3)$$

where $\mathbf{0}$ is the zero vector in \mathbb{R}^n , and $\mathbf{f}(\cdot, \cdot)$, $\mathbf{u} \in \mathbb{R}^n$. The main algorithms in AUTO are aimed at the continuation of solutions of autonomous systems of ODEs of the form

$$\mathbf{u}'(t) = \mathbf{f}(\mathbf{u}(t), \mathbf{p}), \quad \text{where } \mathbf{f}(\cdot, \cdot), \mathbf{u}(\cdot) \in \mathbb{R}^n, \quad (1.4)$$

subject to boundary or initial conditions and integral constraints. Above, \mathbf{p} denotes the vector of *free parameters*. These boundary value algorithms also allow AUTO to do certain stationary solution and wave calculations for the partial differential equation (PDE)

$$\mathbf{u}_t = \mathbf{D}\mathbf{u}_{xx} + \mathbf{f}(\mathbf{u}, \mathbf{p}), \quad \text{where } \mathbf{f}(\cdot, \cdot), \mathbf{u}(\cdot) \in \mathbb{R}^n. \quad (1.5)$$

In this thesis, we utilize AUTO's capabilities for Equation (1.3) and especially for Equation (1.4) [34].

1.3 Stable and unstable invariant manifolds

“A *manifold* is a topological space that is locally Euclidean (*i.e.*, around every point, there is a neighborhood that is topologically the same as the open unit ball in \mathbb{R}^n)” [106]; see also [12, 37, 38, 47, 59, 65, 82].

Let \mathbf{u}_0 be an equilibrium of the system (1.2) (*i.e.*, $\mathbf{f}(\mathbf{u}_0) = \mathbf{0}$) and let \mathbf{A} denote the Jacobian matrix $d\mathbf{f}/d\mathbf{u}$ evaluated at \mathbf{u}_0 . Let n_- , n_0 , and n_+ be the number of eigenvalues of \mathbf{A} with negative, zero and positive real part, respectively. The equilibrium is called *hyperbolic* if $n_0 = 0$ and either n_- or n_+ , or both are greater than zero; see Figure (2.1) [108]. “The rank, index and signature of a matrix are called the *invariants* of the matrix, a quantity which remains unchanged under certain classes of transformations. Invariants are extremely useful for classifying mathematical objects because they usually reflect intrinsic properties of the object of study” [37]; see also [6, 76]. For example, in linear algebra, if a linear transformation takes a vector from a subspace back to the same subspace, such a transformation is invariant; see [83]. In dynamical systems, an *invariant set* is a subset of the state space such that after the evolution operation the point is still in the subset, for

all elements in the time set [108].

Definition 1.1. An invariant set $\mathbf{S} \subset \mathbb{R}^n$ is said to be a \mathbf{C}^r ($r \geq 1$) *invariant manifold* if \mathbf{S} has the structure of a \mathbf{C}^r differentiable manifold [37, 85, 89].

We give two invariant sets for an equilibrium \mathbf{u}_0 :

$$\mathbf{W}^s(\mathbf{u}_0) = \{\mathbf{u} : \mathbf{u}(t) \rightarrow \mathbf{u}_0, t \rightarrow +\infty\},$$

and

$$\mathbf{W}^u(\mathbf{u}_0) = \{\mathbf{u} : \mathbf{u}(t) \rightarrow \mathbf{u}_0, t \rightarrow -\infty\}.$$

Definition 1.2. $\mathbf{W}^s(\mathbf{u}_0)$ is called the *stable manifold* of \mathbf{u}_0 . $\mathbf{W}^u(\mathbf{u}_0)$ is called the *unstable manifold* of \mathbf{u}_0 .

\mathbf{W}^s and \mathbf{W}^u are manifolds of dimensions n_- and n_+ respectively; see [108]. “A connected topological manifold is locally homeomorphic to Euclidean n -space, and the number n is called the manifold’s dimension” [106]; see also [46, 73, 82, 37]. In most cases in this thesis, an orbit computed by AUTO is a one-dimensional manifold. The adjacent orbits form a surface that is usually a two-dimensional manifold, which we can visualize using a triangulated mesh with computer graphics packages.

1.4 Numerical methods for the computation of stable and unstable manifolds

To find $\mathbf{x} \equiv (\mathbf{u}, \mathbf{p})$ in Equation (1.3) for which $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ with \mathbf{p} fixed, one can use Newton’s method; see [81]. For a modified Newton method, namely the Newton-Chord method for nonlinear systems; see [108]. In AUTO, the tolerance settings for the Newton-Chord method are crucial in deciding the precision of the numerical results.

It is difficult to find analytical solutions for most problems in dynamical systems. To study a dynamical system, the numerical integration of an ODE system plays a key role [10]. There are some general numerical integrators, like Euler’s method, the Trapezoidal

method, Runge-Kutta methods, the BDF methods (backward differentiation formulas), *etc.* [17, 37, 40, 54, 55, 68, 81].

These methods can often be analyzed by Taylor expansion to determine the order of accuracy. It is common practice to use one of the above numerical integrators. However, such integration methods are of limited use. Key issues in the analysis of a dynamical system, *e.g.* to pass a fold or to follow the bifurcated paths, which will be explained in later chapters, need a more powerful algorithm.

Another basic numerical method is *continuation*. For the case of equilibria, the main idea is to start with locating at least one equilibrium at certain fixed parameter values, and then by varying one of the system's parameters, one follows the obtained equilibrium with respect to this parameter. Special points may be detected during this continuation process. In this thesis it will be shown that numerical continuation is also an extremely effective method for computing $2D$ stable or unstable manifolds. We generalize work on orbit continuation as given in [30].

There are two basic tasks for numerical analysis of dynamical systems. The first one is to use continuation to find a solution family and analyze this family. The second is to switch paths according to bifurcation analysis. Software for analyzing dynamical systems and revealing their nature has been developed for decades. Early versions of such software packages are for the computation of one-dimensional solution manifolds. As the computing technology and numerical methods improve, software for dynamical systems can perform more and more complex tasks. The work of Doedel's AUTO was started in the mid 1970s with H. B. Keller at Caltech. The first publication on AUTO appeared in 1981; see [30]. For a description of the evolution of AUTO and related packages see [108]. It shows the key role AUTO plays and its outstanding features compared to other packages.

1.5 Different approaches for computing $2D$ stable/unstable manifold

A stable manifold can be computed as an unstable manifold when time is reversed in the system. When we want to compute an unstable manifold, we could take a small sphere

S_δ in the appropriate eigenspace with radius δ around an equilibrium \mathbf{u}_0 , and evolve S_δ under the flow to generate the manifold $\mathbf{W}^u(\mathbf{u}_0)$. This is fine for one-dimensional unstable manifolds, as the process just evolves two points at distance δ from \mathbf{u}_0 under the flow. When it is used for computing a two or higher dimensional manifold, the mesh we acquire by discretization will deteriorate very rapidly. So the mesh cannot represent the two or higher dimensional manifold $\mathbf{W}^u(\mathbf{u}_0)$ properly. Different approaches have been developed to solve this problem. The usual practice is to grow $\mathbf{W}^u(\mathbf{u}_0)$ from a local neighborhood (S_δ) of \mathbf{u}_0 . Such methods differ in the different mesh representations of $\mathbf{W}^u(\mathbf{u}_0)$. Some of these techniques are described below. For further references see [9] and [77].

Most algorithms used for computing two-dimensional unstable manifolds are for vector fields. Guckenheimer and Worfolk [49] describe a method to compute the two-dimensional unstable manifold of the origin for a specific problem, the Lorenz system, which is also considered in this thesis. It starts with a small circle S_δ around \mathbf{u}_0 in the unstable eigenspace. Then, by iterating, a family of circles can be obtained as the approximation of the unstable manifold.

Krauskopf and Osinga [7, 8] compute $\mathbf{W}^u(\mathbf{u}_0)$ as a sequence of geodesic circles. This method computes new mesh points on the next geodesic circle by solving corresponding boundary value problems appropriately. The growth of the manifold is a sequence of discretized geodesic circles until they are no longer smooth. Adaptive coordinate systems are used to trace the manifold.

In the method of Guckenheimer and Johnson [69], the manifold is computed as a set of curves. If the length of these curves grows very quickly, interpolations are used to place new points on the curves such that the generation of the unstable manifold will be adequate. The remaining issue is that if there are sharp folds, it is difficult to apply an interpolation technique.

Doedel [9] computes unstable manifolds by following orbits that lie on the manifold by numerical continuation. The procedure is stepwise and each step is treated as a two point boundary value problem. The result is very accurate and the setting is flexible because different boundary and integral conditions can be specified. In the two main approaches in this thesis, we utilize Doedel's method, which we shall call *orbit continuation*.

This thesis does not include the computation of one or two-dimensional unstable manifolds of maps; such methods can be found in [63, 67, 88].

1.6 Graphical methods to visualize AUTO solutions

As mentioned in the previous section, we use numerical continuation in AUTO to compute the orbits that form the manifold. The latest version of AUTO is AUTO-07p [34]. The computational results of AUTO are formatted and written to the file system. We utilize QTPlaut based on the QT libraries [96] to interpolate and make projections of the AUTO solution files, acquiring a 2D or 3D graphically representable manifold. Then we use QTPlaut and MATPlaut based on MATLAB [94] to refine the graphical representations. Adjacent orbits in an AUTO solution file can form a smooth surface. So we can triangulate the mesh points along two adjacent orbits, which is how the triangulated surface of AUTO solutions is built. QTPlaut not only ports the manifold to the format that MATLAB can plot, but also is able to convert it into GTS [93] format. Further conversions from GTS format to various other formats like PMesh, Geomview, PLY, VRML, *etc.*, can be done using MeshViewer [95]. There are some common 3D mesh simplification methods [16] in various libraries for the above mentioned mesh formats, like vertex clustering, vertex decimation, *etc.*. The mesh we discuss here for 3D graphical models differs from the mesh mentioned in Section (1.5).

The graphical simplification methods reduce the size of graphical meshes greatly. They also retain the manifold topology very accurately. However, the manifold will be destroyed in a dynamical system sense. When we are using orbit continuation, we must keep the information of the orbits. The manifold should not be treated simply as a mesh. In the implementation, the interpolations, projections, subsettings act only upon the orbits. Thus we maintain the topology of adjacent orbits as much as possible.

1.7 Organization of the thesis

Chapter 2 Orbit Continuation

We introduce the basic numerical concepts and methods underlying AUTO. We also give computational results for two examples of dynamical systems.

Chapter 3 Elementary Bifurcation Concepts

We introduce basic but crucial concepts in bifurcation theory. They are applied in the computations of Chapter 4 and Chapter 6.

Chapter 4 Computing the Lorenz Manifold

We compute the stable manifolds of the Lorenz system with AUTO. We also compute the symbol sequence of heteroclinic connections that are encountered during the computation. Our results are visualized by QTPlaut and MATPlaut.

Chapter 5 The Stability of Periodic Solutions

We discuss the stability analysis of periodic solutions for dynamical systems. The analysis is applied in the computations of Chapter 6.

Chapter 6 The Circular Restricted Three-Body Problem

We give a brief history of this problem. Our focus is the computation of the 2D unstable manifold of the periodic orbits and the heteroclinic and homoclinic connections. We also visualize the results with QTPlaut and MATPlaut. The computational work is an extension of the AUTO demo ‘r3b’.

Chapter 7 Conclusions and Prospects

An overview is given of the work we have done and possible future improvements are suggested.

This thesis uses the methodologies and algorithms implemented in AUTO rather than contributes to bifurcation and dynamical systems theory. We “compute” and “visualize” numerical results. Crucial background material is discussed, and the sources are listed. The development of QTPlaut and MATPlaut is independent work. The study of symbol sequences of the heteroclinic connections of the Lorenz system and the study of the connections of the \mathbf{L} families in the CR3BP can be considered as independent work, but they are not possible to achieve without existing AUTO examples.

Chapter 2

Orbit Continuation

In this chapter we introduce the basic numerical concepts and methods underlying AUTO. At the end of this chapter, we will give computational results for two examples of dynamical systems.

2.1 Stability of equilibria

The solution of an ODE system gives the orbits or *trajectories* of a dynamical system. It is very important to study whether trajectories that evolve close to each other display a similar qualitative behavior. We start our analysis from an equilibrium.

An equilibrium is stable if the system will return to the same state after small disturbances. If the system does not stay at the equilibrium after small disturbances, then the equilibrium is unstable.

In Equation (1.2), we have \mathbf{u}_0 as an equilibrium point, that is, $\mathbf{f}(\mathbf{u}_0) = \mathbf{0}$. The equilibrium point \mathbf{u}_0 has the following classifications:

- It is called stable if, for each $\varepsilon > 0$ there is a $\delta > 0$ so that whenever $|\mathbf{u} - \mathbf{u}_0| < \delta$, then $|\mathbf{u}(t) - \mathbf{u}_0| < \varepsilon$ for all $t > 0$;
- It is called unstable if it is not stable;
- It is called asymptotically stable if it is stable and there is some $\delta > 0$ so that $\mathbf{u}(t) \rightarrow \mathbf{u}_0$ when $t \rightarrow \infty$ for all $\mathbf{u}(0)$ with $|\mathbf{u}(0) - \mathbf{u}_0| < \delta$.

Generally, we will only consider asymptotic stability in this thesis.

“The *phase portrait* of a dynamical system is a partitioning of the state space into orbits” [108]. One of the major tasks of this thesis is to compute and present the orbits of a dynamical system.

The solutions of a linear system can often be found explicitly. However, real life problems are usually modeled by nonlinear systems. The behavior of a dynamical system around an equilibrium can be described by a procedure called linearization, which provides the desired local qualitative description of the solution of a nonlinear system in a linear way. Consider the autonomous system (1.2) in a two-dimensional case. Let $\mathbf{u} = (u_1, u_2)^T$ and write the system as:

$$\begin{cases} u_1' &= f_1(u_1, u_2), \\ u_2' &= f_2(u_1, u_2), \end{cases} \quad (2.1)$$

where $u_1, u_2 \in \mathbb{R}$, $f_1, f_2 : \mathbb{R}^2 \mapsto \mathbb{R}$ and $(\cdot)^T$ denotes the vector transpose. The Taylor expansion at an equilibrium (u_1^0, u_2^0) of the system gives

$$\begin{cases} f_1(u_1, u_2) &= \frac{\partial f_1}{\partial u_1}(u_1^0, u_2^0)(u_1 - u_1^0) + \frac{\partial f_1}{\partial u_2}(u_1^0, u_2^0)(u_2 - u_2^0) + \text{t.h.o.}, \\ f_2(u_1, u_2) &= \frac{\partial f_2}{\partial u_1}(u_1^0, u_2^0)(u_1 - u_1^0) + \frac{\partial f_2}{\partial u_2}(u_1^0, u_2^0)(u_2 - u_2^0) + \text{t.h.o.} \end{cases} \quad (2.2)$$

where t.h.o. denotes the terms of higher order, namely $O(|\mathbf{u} - \mathbf{u}^0|^2)$ and $f_1(u_1^0, u_2^0) = 0$, $f_2(u_1^0, u_2^0) = 0$. The *Jacobian matrix* is

$$\mathbf{f}_{\mathbf{u}}^0 = \begin{pmatrix} \frac{\partial f_1}{\partial u_1}(u_1^0, u_2^0) & \frac{\partial f_1}{\partial u_2}(u_1^0, u_2^0) \\ \frac{\partial f_2}{\partial u_1}(u_1^0, u_2^0) & \frac{\partial f_2}{\partial u_2}(u_1^0, u_2^0) \end{pmatrix}. \quad (2.3)$$

The linear system $(\mathbf{u} - \mathbf{u}_0)' = \mathbf{f}_{\mathbf{u}}^0(\mathbf{u} - \mathbf{u}_0)$ locally approximates the non-linear system (1.2); see [81]. The roots λ_i , $i = 1, 2, \dots, n$, of the characteristic polynomial $\det(\mathbf{f}_{\mathbf{u}}^0 - \lambda \mathbf{I}) = 0$ are the *eigenvalues* of the system’s Jacobian at \mathbf{u}_0 . In \mathbb{R}^2 , there are three basic types of equilibria. Their types are indicated by the values of the eigenvalues; see the Table (2.1) and Figure (2.1). For further references see [81] and [108].

Equilibrium type	Characteristics of eigenvalues
<i>Node</i>	Two real eigenvalues of the same sign
<i>Focus</i>	A pair of conjugate complex eigenvalues
<i>Saddle</i>	Real eigenvalues of different sign

Table 2.1: Classification of equilibrium points

2.2 Continuation of solutions

2.2.1 Regular solutions

Consider the equation

$$\mathbf{G}(\mathbf{u}, \lambda) = \mathbf{0}, \quad \mathbf{u}, \mathbf{G}(\cdot, \cdot) \in \mathbb{R}^n, \quad \lambda \in \mathbb{R}. \quad (2.4)$$

With $\mathbf{x} \equiv (\mathbf{u}, \lambda)$, Equation (2.4) can be re-written as

$$\mathbf{G}(\mathbf{x}) = \mathbf{0}, \quad \mathbf{G} : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^n. \quad (2.5)$$

We have a solution \mathbf{x}_0 of $\mathbf{G}(\mathbf{x}) = 0$. \mathbf{x}_0 is *regular* if the n (rows) by $n + 1$ (columns) matrix $\mathbf{G}_x^0 \equiv \mathbf{G}_x(\mathbf{x}_0)$ has maximal rank. Near a regular solution \mathbf{x}_0 , there exists a unique one-dimensional continuum of solution $\mathbf{x}(s)$, called *a solution family* or *a solution branch*; see [30].

2.2.2 Parameter continuation

The discussion in the previous section hints at a way of getting a family of stationary solutions of a dynamical system, which is usually the first step in analyzing the system. Suppose we have a solution $(\mathbf{u}_0, \lambda_0)$ of $\mathbf{G}(\mathbf{u}, \lambda) = \mathbf{0}$ and the direction vector $\dot{\mathbf{u}}_0 = d\mathbf{u}/d\lambda$. The computation of a solution \mathbf{u}_1 at $\lambda_1 = \lambda_0 + \Delta\lambda$ can be done by solving the system (2.4) for \mathbf{u} using Newton's method. We get the following system

$$\begin{cases} \mathbf{G}_u(\mathbf{u}_1^{(\nu)}, \lambda_1) \Delta \mathbf{u}_1^{(\nu)} = -\mathbf{G}(\mathbf{u}_1^{(\nu)}, \lambda_1), \\ \mathbf{u}_1^{(\nu+1)} = \mathbf{u}_1^{(\nu)} + \Delta \mathbf{u}_1^{(\nu)}, \quad \nu = 0, 1, 2, \dots \end{cases} \quad (2.6)$$

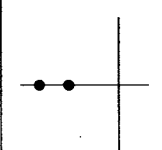
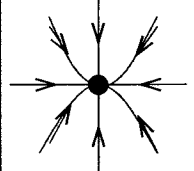
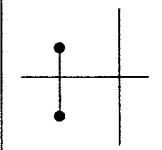

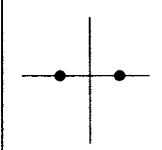
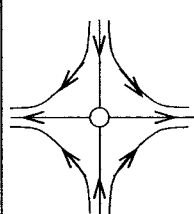
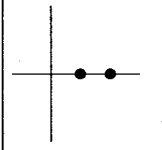
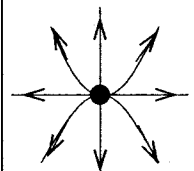
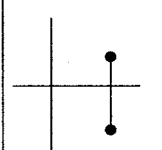
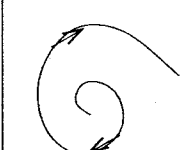
(n_+, n_-)	Eigenvalues	Phase portrait	Stability
(0, 2)		 Node	stable
		 focus	
(1, 1)		 saddle	unstable
(2, 0)		 node	unstable
		 focus	

Figure 2.1: Basic equilibria types in R^2 [108].

At the first step of the iteration, we set $\mathbf{u}_1^{(0)} = \mathbf{u}_0 + \Delta\lambda\dot{\mathbf{u}}_0$. If $\mathbf{G}_u(\mathbf{u}_1, \lambda_1)$ is nonsingular and $\Delta\lambda$ is sufficiently small, this iteration will converge. After \mathbf{u}_1 is obtained, the new direction vector $\dot{\mathbf{u}}_1$ can be computed by solving system $\mathbf{G}_u(\mathbf{u}_1, \lambda_1)\dot{\mathbf{u}}_1 = -\mathbf{G}_\lambda(\mathbf{u}_1, \lambda_1)$.

Parameter continuation will typically fail if the solution family has a fold. For details

see Section (3.1).

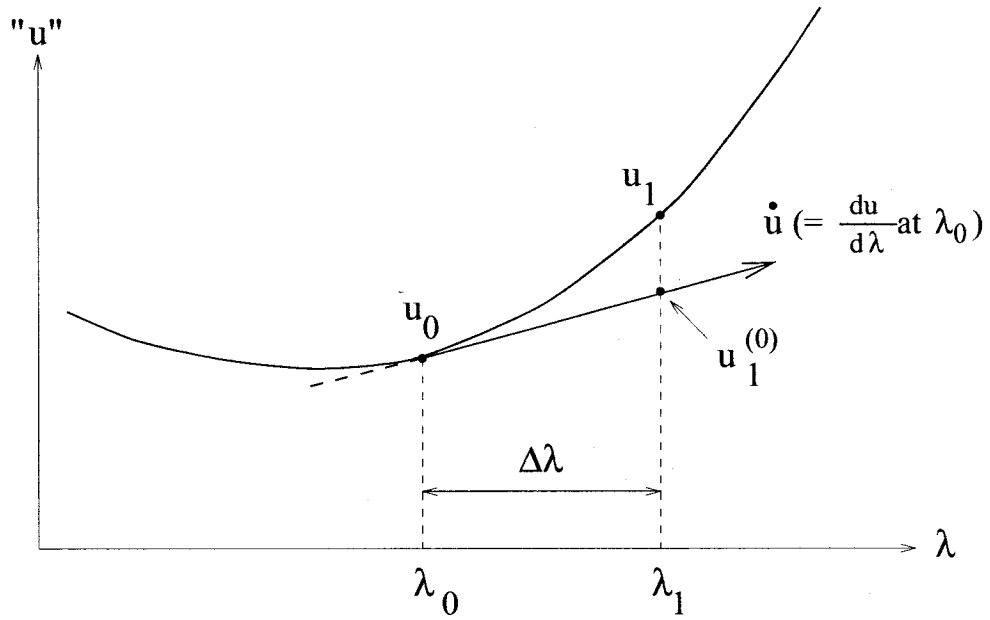


Figure 2.2: Graphical interpretation of parameter continuation [30].

2.2.3 Keller's pseudo-arclength continuation

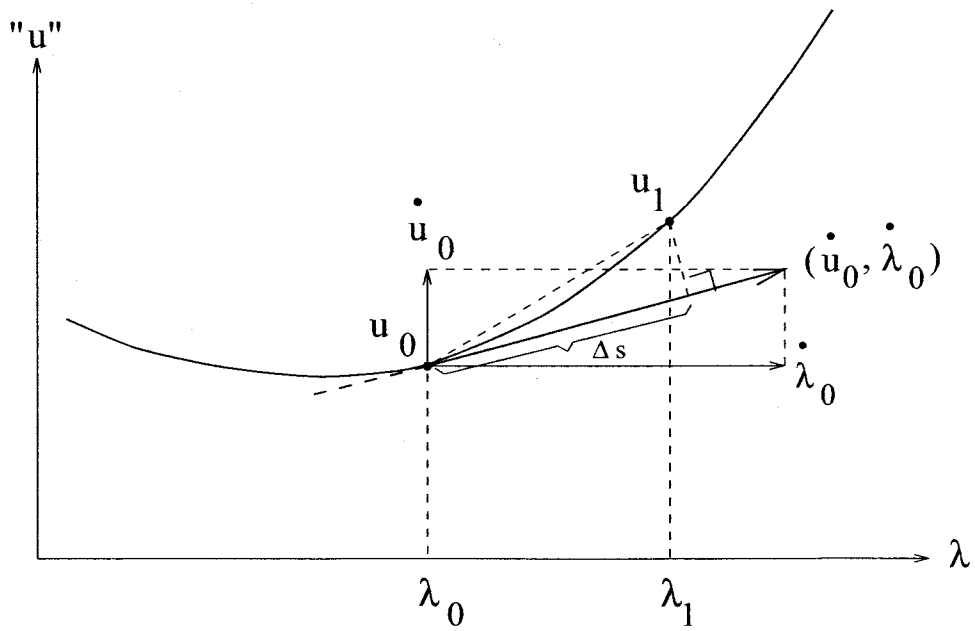


Figure 2.3: Graphical interpretation of pseudo-arclength continuation [30].

AUTO uses Keller's pseudo-arclength continuation which, in particular, allows the continuation to pass a fold. Given a solution $(\mathbf{u}_0, \lambda_0)$ of $\mathbf{G}(\mathbf{u}, \lambda) = \mathbf{0}$ and the direction vector $(\dot{\mathbf{u}}_0, \dot{\lambda}_0)$, the system to solve for $(\mathbf{u}_1, \lambda_1)$ is

$$\begin{cases} \mathbf{G}(\mathbf{u}_1, \lambda_1) & = \mathbf{0} \quad , \\ \langle \mathbf{u}_1 - \mathbf{u}_0, \dot{\mathbf{u}}_0 \rangle + (\lambda_1 - \lambda_0)\dot{\lambda}_0 - \Delta s & = 0 \quad ; \end{cases} \quad (2.7)$$

here $\langle \cdot, \cdot \rangle$ denotes the dot product of two vectors [30]. The Newton iteration will be

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

with the new direction obtained from

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

The Jacobian of the pseudo-arclength system is nonsingular at a regular solution point; see [30].

2.3 Boundary value problems

Suppose we have a first-order system of ordinary differential equations

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

where

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

It is subject to boundary conditions

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

and integral constraints

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

AUTO obtains solutions by solving such a boundary value problem (BVP) for $\mathbf{u}(\cdot)$ and $\boldsymbol{\mu}$. We require that $n_\mu = n_b + n_q - n \geq 0$, for this problem to be formally well-posed. Above, λ is the continuation parameter in which the solution $(\mathbf{u}, \boldsymbol{\mu})$ may be continued. A simple case is that where $n_q = 0$, $n_b = n$, and $n_\mu = 0$.

2.3.1 Orthogonal collocation

AUTO uses *orthogonal collocation with piecewise polynomials* to solve boundary value problems [14, 97]. It gives very accurate results and allows adaptive mesh-selection. We briefly describe the set-up.

First, there is a mesh

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

with

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

The space of (vector-valued) piecewise polynomials \mathbb{P}_h^m is defined as

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

where \mathbb{P}^m is the space of (vector-valued) polynomials of degree $\leq m$. The orthogonal collocation method with piecewise polynomials [99] is to find $\mathbf{p}_h \in \mathbb{P}_h^m$ and $\boldsymbol{\mu} \in \mathbb{R}^{n_\mu}$, so the following *collocation equations* are satisfied:

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

and \mathbf{p}_h also satisfies the boundary and integral conditions. The *collocation points* $z_{j,i}$ in each subinterval $[t_{j-1}, t_j]$ are the (scaled) roots of the m th-degree orthogonal polynomial (*Gauss points*); for a graphical interpretation see Figure (2.4). Since each local polynomial

is determined by $(m + 1)n$ coefficients, the total number of degrees of freedom (considering λ as fixed) is $(m + 1)nN + n_\mu$. This is matched by the total number of equations:

$$\text{collocation} : mnN,$$

$$\text{continuity} : (N - 1)n,$$

$$\text{constraints} : n_b + n_q \quad (= n + n_\mu).$$

If the solution $\mathbf{u}(t)$ of the BVP is sufficiently smooth then the order of accuracy of the orthogonal collocation method is m , *i.e.*,

$$\| \mathbf{p}_h - \mathbf{u} \|_\infty = O(h^m).$$

At the main mesh-points t_j we have *superconvergence*:

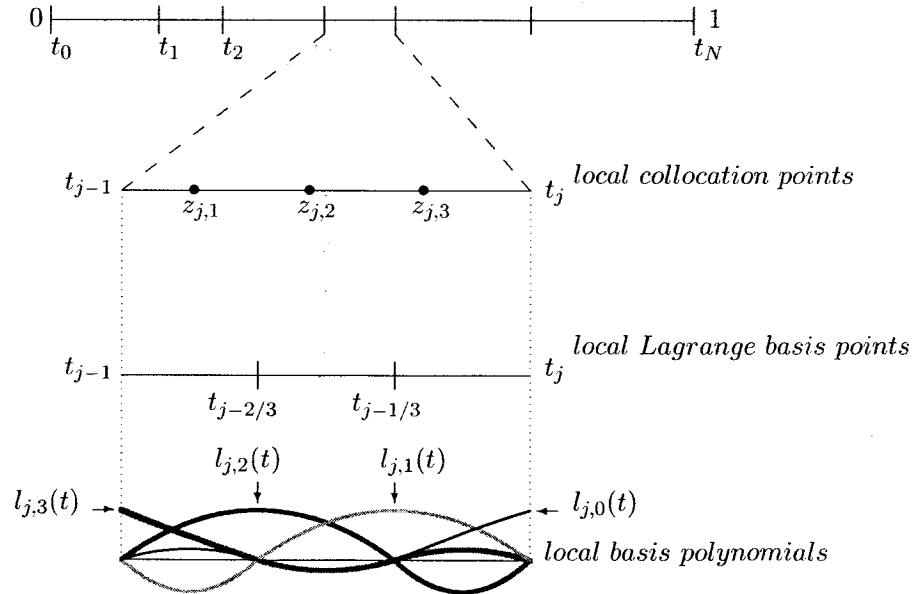


Figure 2.4: The mesh $\{ 0 = t_0 < t_1 < \dots < t_N = 1 \}$. Collocation points are shown for the case $m = 3$.

$$\max_j |\mathbf{p}_h(t_j) - \mathbf{u}(t_j)| = O(h^{2m}).$$

The scalar variables $\boldsymbol{\mu}$ are also superconvergent [14]. Combined with good mesh adaption strategies, this method gives very accurate solutions. This orthogonal collocation method with piecewise polynomial has been implemented very efficiently in AUTO.

2.4 Implementation of the orthogonal collocation method

In AUTO, the discretization of orbits uses the method of orthogonal collocation with piecewise polynomials. The number of collocation points per mesh interval is between two and seven, see [14, 97, 98]. The Lagrange basis polynomials for each subinterval $[t_{j-1}, t_j]$ are

$$\{\ell_{j,i}(t)\}, \quad j = 1, 2, \dots, N, \quad i = 0, 1, 2, \dots, m,$$

defined as

$$\ell_{j,i}(t) = \prod_{k=0, k \neq i}^m \frac{t - t_{j-\frac{k}{m}}}{t_{j-\frac{i}{m}} - t_{j-\frac{k}{m}}}, \quad (2.8)$$

where

$$t_{j-\frac{i}{m}} \equiv t_j - \frac{i}{m} h_j. \quad (2.9)$$

See Figure (2.4) for an illustration. The local polynomial is then of the form

$$\mathbf{p}_j(t) = \sum_{i=0}^m \ell_{j,i}(t) \mathbf{u}_{j-\frac{i}{m}}. \quad (2.10)$$

With this choice of basis, \mathbf{u}_j will approximate $\mathbf{u}(t_j)$, and $\mathbf{u}_{j-\frac{i}{m}}$ will approximate $\mathbf{u}(t_{j-\frac{i}{m}})$, where $\mathbf{u}(t)$ is the solution of the continuous problem.

As described before, the collocation equations are

$$\mathbf{p}'_j(z_{j,i}) = \mathbf{f}(\mathbf{p}_j(z_{j,i}), \boldsymbol{\mu}, \lambda), \quad i = 1, 2, \dots, m, \quad j = 1, 2, \dots, N. \quad (2.11)$$

The discrete boundary conditions are

$$b_i(\mathbf{u}_0, \mathbf{u}_N, \boldsymbol{\mu}, \lambda) = 0, \quad i = 1, \dots, n_b. \quad (2.12)$$

The integrals can be discretized as

$$\sum_{j=1}^N \sum_{i=0}^m \omega_{j,i} q_k(\mathbf{u}_{j-\frac{i}{m}}, \boldsymbol{\mu}, \lambda) = 0, \quad k = 1, \dots, n_q, \quad (2.13)$$

where the $\omega_{j,i}$ are the Lagrange quadrature coefficients.

The pseudo-arclength continuation equation is

$$\int_0^1 \langle \mathbf{u}(t) - \mathbf{u}_0(t), \dot{\mathbf{u}}_0(t) \rangle dt + \langle \boldsymbol{\mu} - \boldsymbol{\mu}_0, \dot{\boldsymbol{\mu}}_0 \rangle + (\lambda - \lambda_0) \dot{\lambda}_0 - \Delta s = 0,$$

where $(\mathbf{u}_0, \boldsymbol{\mu}_0, \lambda_0)$ is the previous computed solution of the solution family, and $(\dot{\mathbf{u}}_0, \dot{\boldsymbol{\mu}}_0, \dot{\lambda}_0)$ is the normalized direction of the solution family at the previous solution. The pseudo-arclength equation can be discretized as

$$\sum_{j=1}^N \sum_{i=0}^m \omega_{j,i} \langle \mathbf{u}_{j-\frac{i}{m}} - (\mathbf{u}_0)_{j-\frac{i}{m}}, (\dot{\mathbf{u}}_0)_{j-\frac{i}{m}} \rangle + \langle \boldsymbol{\mu} - \boldsymbol{\mu}_0, \dot{\boldsymbol{\mu}}_0 \rangle + (\lambda - \lambda_0) \dot{\lambda}_0 - \Delta s = 0,$$

where $\omega_{j,i}$ denotes quadrature weights.

2.5 Orbit continuation

AUTO can also use continuation to compute solution families to initial value problem. It has a great advantage over integration of a large number of initial conditions, because the manifold described by the orbits is well covered. Even when the system we are solving has very sensitive dependence on the initial conditions, orbit continuation still gives reliable results.

Suppose a dynamical system in the general form

$$\mathbf{u}' = \mathbf{f}(\mathbf{u})$$

has a saddle equilibrium \mathbf{u}_0 with a two-dimensional unstable manifold. For the case of a stable manifold the integration time T will be in the negative direction. Suppose we know that the Jacobian $\mathbf{f}_{\mathbf{u}}(\mathbf{u}_0)$ has exactly two positive real eigenvalues μ_1 and μ_2 with $\mu_1 \geq \mu_2$. Let \mathbf{v}_1 and \mathbf{v}_2 be the respective eigenvectors. We want to use continuation to compute the corresponding unstable manifold. We write the system in a scaled time format (a detailed derivation is in Chapter 4)

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

The initial condition is

$$\mathbf{u}(0) = \mathbf{u}_0 + \delta(\cos(\theta)\mathbf{v}_1 - \sin(\theta)\mathbf{v}_2), \quad (0 \leq \theta < 2\pi),$$

i.e., $\mathbf{u}(0)$ lies on a small closed curve around \mathbf{u}_0 . The choice of $\theta \in [0, 2\pi)$, guarantees that the manifold spanned by the two eigenvectors will be fully covered, at least in some neighborhood of \mathbf{u}_0 .

At first, we set a fixed θ . Normally we choose $\theta = 0$, so that $\mathbf{u}(0) = \mathbf{u}_0 + \delta\mathbf{v}_1$, where \mathbf{v}_1 is the eigenvector corresponding to the larger eigenvalue μ_1 . In this case, the orbit can usually be computed for sufficiently long time. Once the starting orbit is obtained, for example, a desired length is obtained, this orbit is subsequently continued numerically, where the initial point on the small closed curve around \mathbf{u}_0 , *i.e.*, as controlled by θ , is now one of the continuation variables. In this step, the system can be formalized as $\mathbf{F}(\mathbf{X}) \equiv \mathbf{0}$, where

$$\mathbf{F}(\mathbf{X}) \equiv \begin{cases} \mathbf{u}'(t) - T\mathbf{f}(\mathbf{u}(t)) \\ \mathbf{u}(0) - \delta(\cos(\theta)\mathbf{v}_1 - \sin(\theta)\mathbf{v}_2) \\ T \int_0^1 \|\mathbf{f}(\mathbf{u}(s))\| ds - L. \end{cases} \quad (2.14)$$

Suppose we have the solution \mathbf{X}_i . Then obtain \mathbf{X}_{i+1} by solving

$$\begin{cases} \mathbf{F}(\mathbf{X}_{i+1}) = \mathbf{0} , \\ \langle \mathbf{X}_{i+1} - \mathbf{X}_i, \dot{\mathbf{X}}_i \rangle - \Delta s = 0. \end{cases} \quad (2.15)$$

Here, $\|\dot{\mathbf{X}}_i\| = 1$ and $\mathbf{X} \equiv (\mathbf{u}(\cdot), \theta, T)$, while L and δ are fixed. We note that we do not just change the initial point (*i.e.*, the value of θ), and that the continuation step size Δs measures the change in \mathbf{X} [30].

2.5.1 Example 1, AUTO demo um2

Consider the system

$$\mathbf{u}' = \mathbf{f}(\mathbf{u}),$$

where

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \quad \text{and} \quad \mathbf{f}(\mathbf{u}) = \begin{pmatrix} \epsilon u_1 - u_2^3 \\ u_2 + u_1^3 \end{pmatrix}. \quad (2.16)$$

We see that point $\mathbf{u} = \mathbf{0}$ is an equilibrium. The Jacobian is

$$\mathbf{f}_{\mathbf{u}}(\mathbf{u}) = \begin{pmatrix} \epsilon & -3u_2^2 \\ 3u_1^2 & 1 \end{pmatrix},$$

and

$$\mathbf{f}_{\mathbf{u}}(\mathbf{0}) = \begin{pmatrix} \epsilon & 0 \\ 0 & 1 \end{pmatrix}.$$

From $\det(\mathbf{f}_{\mathbf{u}} - \mu \mathbf{I}) = 0$, we get the two eigenvalues $\mu_1 = \epsilon$, $\mu_2 = 1$. If $\epsilon > 0$, we have a 2D unstable manifold at the equilibrium point $(0, 0)$. This system is a good example showing the sensitive character of the initial condition when ϵ is small and positive. We use the AUTO demo um2 to compute families of orbits for different values of ϵ . In order to show the performance of AUTO in dealing with such an initial condition sensitive problem, we

also use an explicit ODE integrator. The choice is the fourth-order Runge-Kutta method,

$$\begin{aligned}
\mathbf{k}_1 &= \Delta t \mathbf{f}(t_n, \mathbf{u}_n) \quad , \\
\mathbf{k}_2 &= \Delta t \mathbf{f}\left(t_n + \frac{\Delta t}{2}, \mathbf{u}_n + \frac{1}{2} \mathbf{k}_1\right) \quad , \\
\mathbf{k}_3 &= \Delta t \mathbf{f}\left(t_n + \frac{\Delta t}{2}, \mathbf{u}_n + \frac{1}{2} \mathbf{k}_2\right) \quad , \\
\mathbf{k}_4 &= \Delta t \mathbf{f}(t_n + \Delta t, \mathbf{u}_n + \mathbf{k}_3) \quad , \\
\mathbf{u}_{n+1} &= \mathbf{u}_n + \frac{1}{6} \mathbf{k}_1 + \frac{1}{3} \mathbf{k}_2 + \frac{1}{3} \mathbf{k}_3 + \frac{1}{6} \mathbf{k}_4 \quad .
\end{aligned} \tag{2.17}$$

We use a C++ program to extract the initial points from AUTO results, and then we integrate the system from these initial values. The solution of AUTO and the solution of the fourth-order Runge-Kutta method are put in the same figure with QTPlaut. For the best visual effects, we plot the results as Figure (2.5) to (2.9) using MATPlaut. In these figures, the results of AUTO are colored blue and the results of the fourth-order Runge-Kutta method are colored red. In both methods, the computation starts from a small circle centered at the equilibrium point $(0, 0)$ with a radius 0.1 and ends when the end point is at distance 0.6 from the origin. In the Runge-Kutta method, Δt is 0.001. Since the method is $O(\Delta t^4)$, it has rather high accuracy. The accuracy setting of AUTO can be defined in the constants files. We observe that when $\epsilon = 1.0$ and $\epsilon = 0.5$, the results of both methods coincide, and the 2D-manifolds differ little graphically. In these cases, since it is an explicit method, the fourth-order Runge-Kutta method is faster. However, as ϵ decreases to 0.05, the Runge-Kutta method is incapable to cover the same manifold that AUTO generates. Not only do the two manifolds fail to coincide, but also the orbits computed by the Runge-Kutta method jump and fail to form a smooth manifold. However, Equation (2.16) shows that the solutions of the um2 demo form a smooth 2D-manifold. When $\epsilon = 0.005$, the Runge-Kutta method fails to cover the manifold shown in Figure (2.8). We further change Δt to 0.0001, but in the graphical presentation, see Figure (2.9), there is no significant improvement. We can decrease Δt even further, but the integration will take noticeably longer time and the results don't improve, whereas AUTO is able to cover the manifold even when ϵ is as small as 10^{-10} .

2.5.2 Example 2, AUTO demo um3

The system discussed in the above section can be extended to a three-dimensional system as follows:

$$\mathbf{u}' = \mathbf{f}(\mathbf{u}),$$

where

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \quad \text{and} \quad \mathbf{f}(\mathbf{u}) = \begin{pmatrix} \epsilon u_1 - u_2^3 + u_3^3 \\ u_2 + u_1^3 \\ -u_3 + u_1^2 \end{pmatrix}. \quad (2.18)$$

Here

$$\mathbf{f}_{\mathbf{u}}(\mathbf{u}) = \begin{pmatrix} \epsilon & -3u_2^2 & 3u_3^2 \\ 3u_1^2 & 1 & 0 \\ 2u_1 & 0 & -1 \end{pmatrix},$$

where the point $(0, 0, 0)$ is an equilibrium. The eigenvalues at $(0, 0, 0)$ are $\mu_1 = \epsilon$, $\mu_2 = 1$, $\mu_3 = -1$. Thus we have a 2D unstable manifold and 1D stable manifold at the point $(0, 0, 0)$. Again we compare the Runge-Kutta and the continuation methods discussed previously, as illustrated in Figures (2.10) to (2.12). When $\epsilon = 1$, in Figure (2.10), the manifolds spanned by AUTO and the 4th-order Runge-Kutta method agree. When $\epsilon = 0.5$, in Figure (2.11), there are subtle differences in the manifolds. When $\epsilon = 0.05$, the Runge-Kutta method again fails to span the manifold as AUTO does.

In both examples, the 4th-order Runge-Kutta is inaccurate not only because its graphical representation does not coincide with the results of AUTO, but also because the orbits jump irregularly, so that adjacent orbits will not form a part of a smooth manifold. This contradicts the smooth variation of the mathematical solutions of the system.

2.5.3 Discussion

Initial condition sensitive equations may have sharp turns in the trajectories governed by the evolution laws. An explicit integrator fails at dealing with such equations. Like walking in the dark with a dimmed flash light, a good solver of dynamical systems like AUTO judges its step according to the known situation. At sharp, bumpy turns, it will take careful steps.

When the road is straight and even, it will stride with confidence and speed. When things go wrong, it can provide analysis, step back or suspend for user interaction. AUTO utilizes Keller's pseudo-arclength algorithm which enhances convergence at each step. It also utilizes dynamic mesh adaption and it uses collocation with super-convergent accuracy. Further, AUTO uses orbit continuation on entire trajectories to span the manifold. For these reasons the computational results are more likely to have the smoothness that can be derived from their mathematical formalization.

For further references of the algorithms in AUTO see [30].

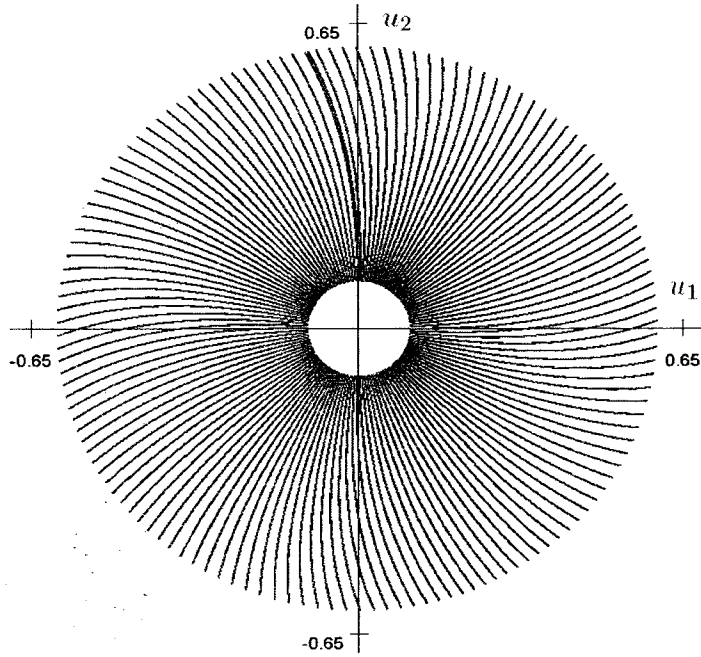


Figure 2.5: AUTO demo um2 with $\epsilon = 1.0$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.001$ are colored red.

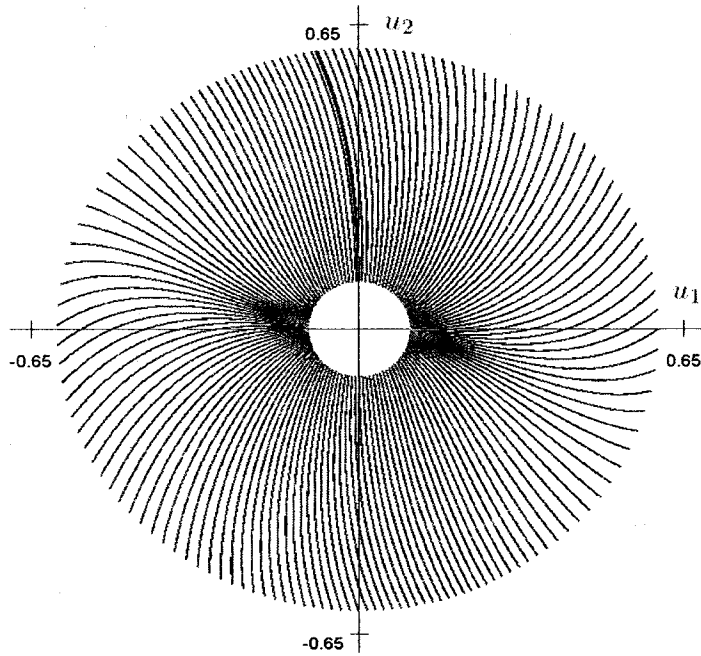


Figure 2.6: AUTO demo um2 with $\epsilon = 0.5$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.001$ are colored red.

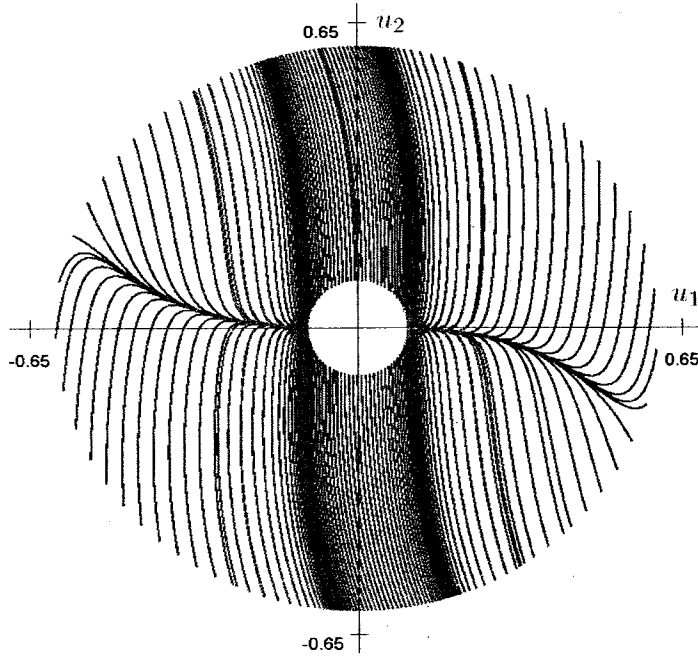


Figure 2.7: AUTO demo um2 with $\epsilon = 0.05$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.001$ are colored red.

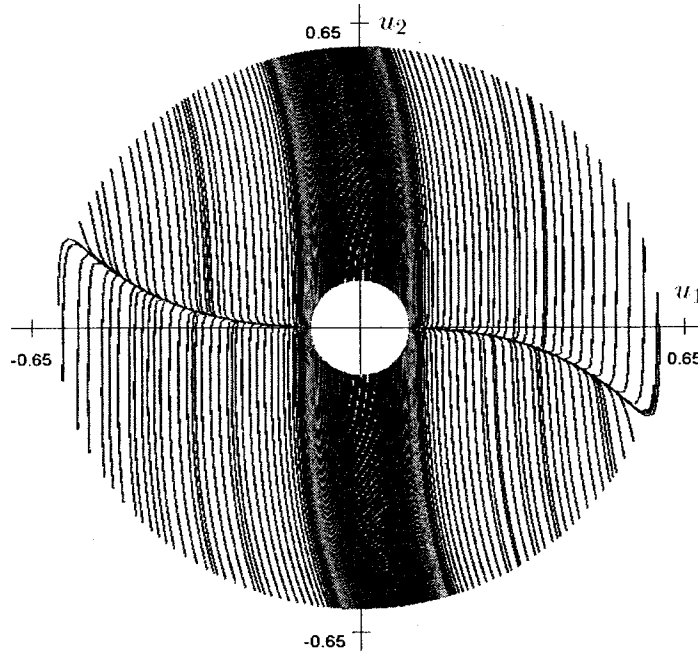


Figure 2.8: AUTO demo um2 with $\epsilon = 0.005$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.001$ are colored red.

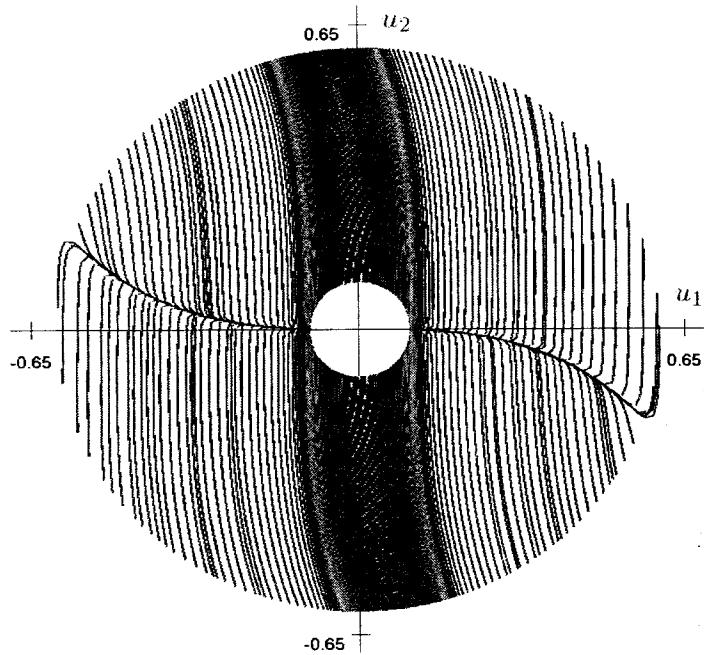


Figure 2.9: AUTO demo um2 with $\epsilon = 0.005$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.0001$ are colored red.

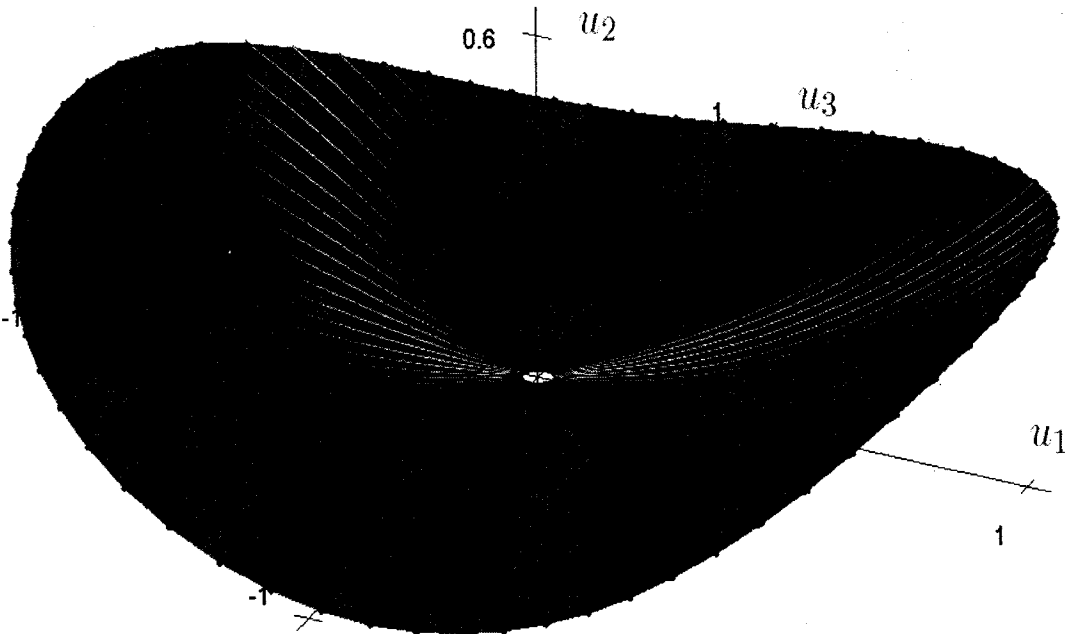


Figure 2.10: AUTO demo um3 with $\epsilon = 1.0$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.001$ are colored red.

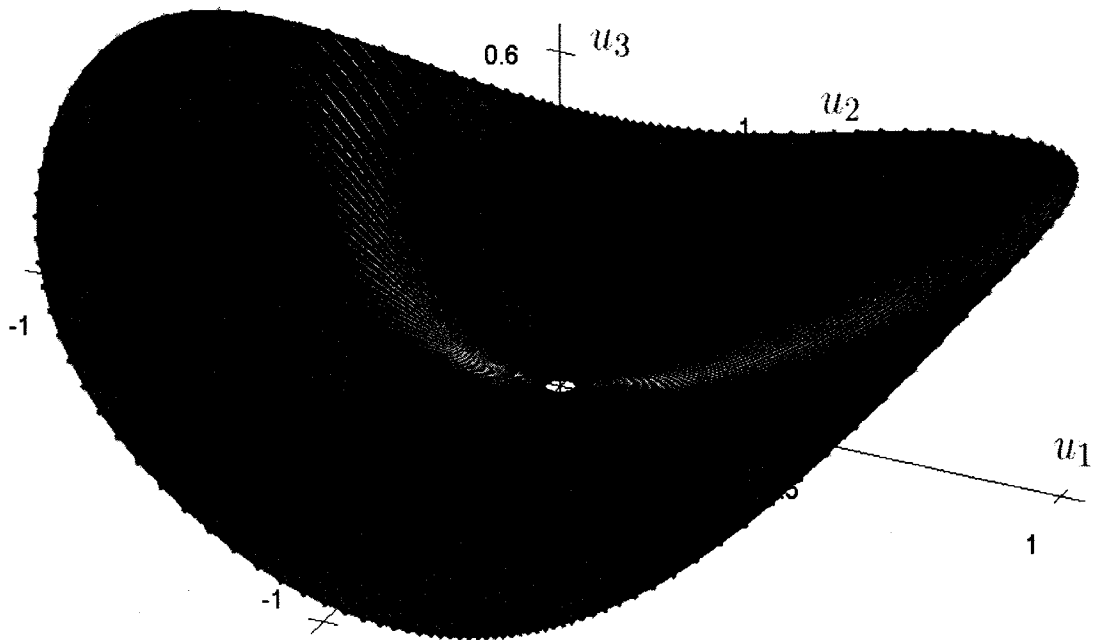


Figure 2.11: AUTO demo um3 with $\epsilon = 0.5$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.001$ are colored red.

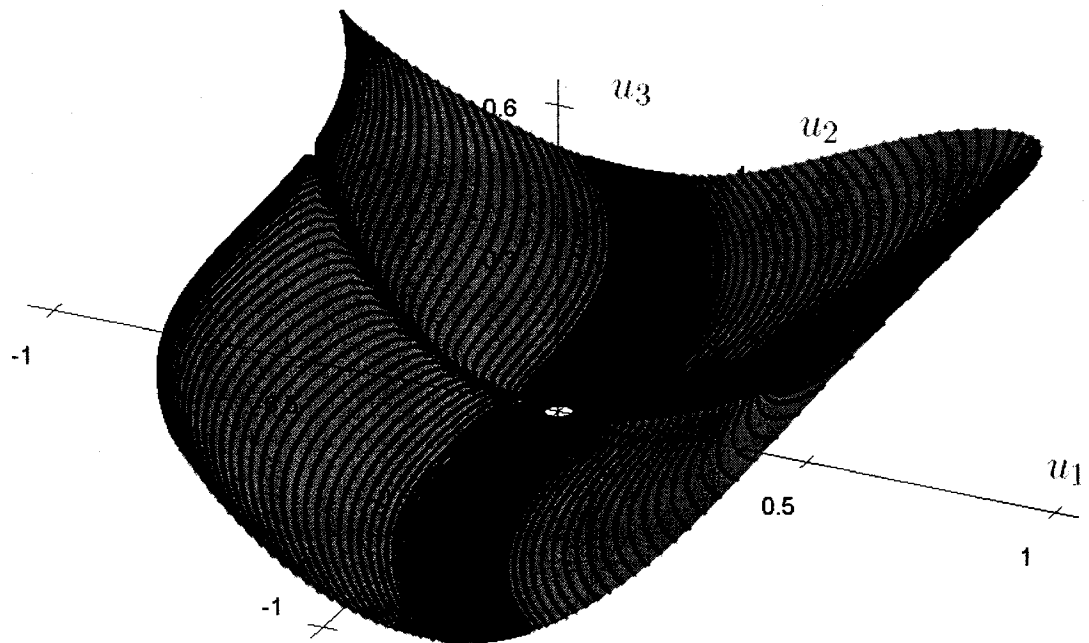


Figure 2.12: AUTO demo um3 with $\epsilon = 0.05$. The results of AUTO are colored blue; the results of the 4th-order Runge-Kutta method with $\Delta t = 0.001$ are colored red.

Chapter 3

Elementary Bifurcation Concepts

In this chapter, we introduce basic but crucial concepts in bifurcation theory for this thesis. In Chapter 2, we gave two examples for which the manifolds can be computed without bifurcation analysis. However, the computations in Chapter 4 and Chapter 6 can not be done without such analysis. A discussion of the stability of periodic solutions is given in Chapter 5. For a further reference for this Chapter, we refer to [81].

3.1 Bifurcation and bifurcation diagrams

Consider a finite dimensional system

$$\mathbf{f}(\mathbf{u}, \lambda) = \mathbf{0}, \quad \mathbf{u} \in \mathbb{R}^n, \mathbf{f}(\cdot, \cdot) \in \mathbb{R}^n \text{ and } \lambda \in \mathbb{R}. \quad (3.1)$$

We first consider the case $n = 1$. We apply the chain rule to $f(u(\lambda), \lambda) = 0$ to help us to identify the relation between the value of λ and the solution. We get

$$\frac{\partial f(u, \lambda)}{\partial u} \frac{du}{d\lambda} + \frac{\partial f(u, \lambda)}{\partial \lambda} = 0. \quad (3.2)$$

Thus the relation between u and λ

$$\frac{du}{d\lambda} = -\frac{\partial f / \partial \lambda}{\partial f / \partial u}. \quad (3.3)$$

There are three basic cases in deciding the nature of the solution family of $u(\lambda)$.

- $f_u(u_0, \lambda_0) \neq 0$: The curve can be uniquely continued in a λ -neighborhood, that is, the curve is smoothly continuable.
- $f_u(u_0, \lambda_0) = 0$ and $f_\lambda(u_0, \lambda_0) \neq 0$: The tangent of the curve $u(\lambda)$ is not defined.
- $f_u(u_0, \lambda_0) = 0$ and $f_\lambda(u_0, \lambda_0) = 0$: The tangent of curve $u(\lambda)$ is not unique. We need special methods to find the tangent(s).

In the case $\mathbf{u} \in \mathbb{R}^n$, we require a measurement of the vector \mathbf{u} . We shall use the notation $[\mathbf{u}]$ for such a scalar measure of \mathbf{u} . For example we can choose

$$[\mathbf{u}] = u_k, \quad \text{for some } k, 1 \leq k \leq n;$$

$$[\mathbf{u}] = \|\mathbf{u}\|_2 := (u_1^2 + u_2^2 + \dots + u_n^2)^{1/2};$$

$$[\mathbf{u}] = \|\mathbf{u}\|_\infty := \max\{|u_1|, |u_2|, \dots, |u_n|\},$$

where $\|\cdot\|$ denotes the norm of a vector. Equipped with a scalar measure $[\mathbf{u}]$ of the vector \mathbf{u} , we are able to depict the solutions of Equation (3.1) in a diagram like Figure (3.1).

Definition 3.1. A diagram depicting $[\mathbf{u}]$ versus λ , where (\mathbf{u}, λ) solves Equation (3.1), is called a *bifurcation diagram*.

Solutions may form continua, reflected by continuous curves in bifurcation diagrams. The continua of solutions are called *branches* or *solution families*.

Definition 3.2. A *branch point* or *bifurcation point* (with respect to λ) is a solution (\mathbf{u}_0, λ) of Equation (3.1), where the number of solutions changes when λ passes λ_0 .

While varying the parameter λ , a point where solutions begin to exist can be a *fold*. Other names used for a fold are *limit point*, *turning point*, or *saddle node bifurcation*. We call a solution family or a part of a solution family *stable* (*unstable*) if all its solutions are *stable* (*unstable*). A family is *periodic*, *symmetric*, or *stationary* if its solutions are *periodic*, *symmetric*, or *stationary*. We see that locally there are no solutions on one side of a fold and

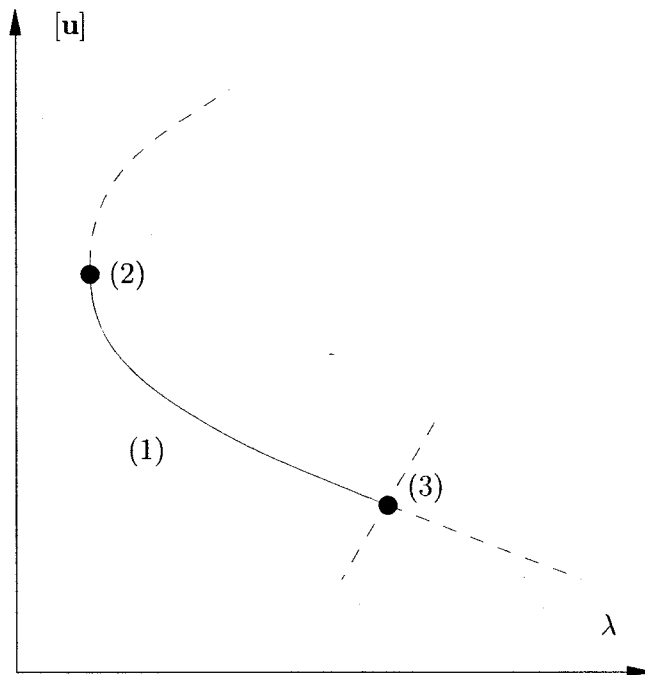


Figure 3.1: An example bifurcation diagram [81].

two solutions on the other side. At a fold, two solutions are born or two solutions annihilate each other. We can also characterize a bifurcation point geometrically: two branches with distinct tangents intersect at a bifurcation point .

Several key concepts from linear algebra are crucial for bifurcation analysis. “The *column space* of a matrix is the set of all possible linear combinations of its column vectors. The column space of an $m \times n$ matrix is a subspace of m -dimensional Euclidean space. The dimension of the column space is called the *rank* of the matrix. The column space of a matrix is the *range* of the corresponding matrix transformation” [15, 24, 41, 83, 84, 106].

Definition 3.3. Let \mathbf{u} be an n -vector, $(\mathbf{u}_0, \lambda_0)$ is a *simple stationary bifurcation point* if the following four conditions are true:

- $\mathbf{f}(\mathbf{u}_0, \lambda_0) = \mathbf{0}$;
- rank of $\mathbf{f}_{\mathbf{u}}(\mathbf{u}_0, \lambda_0) = n - 1$;
- $\mathbf{f}_{\lambda}(\mathbf{u}_0, \lambda_0) \in \text{range } \mathbf{f}_{\mathbf{u}}(\mathbf{u}_0, \lambda_0)$; and
- exactly two branches of stationary solutions intersect with two distinct tangents.

Definition 3.4. $(\mathbf{u}_0, \lambda_0)$ is a (simple quadratic) *fold* (also referred as a turning point, a fold bifurcation, a limit point bifurcation, or a saddle-node bifurcation) of stationary solutions if the following four conditions are met:

- $\mathbf{f}(\mathbf{u}_0, \lambda_0) = \mathbf{0}$;
- rank of $\mathbf{f}_{\mathbf{u}}(\mathbf{u}_0, \lambda_0) = n - 1$;
- $\mathbf{f}_{\lambda}(\mathbf{u}_0, \lambda_0) \notin \text{range } \mathbf{f}_{\mathbf{u}}(\mathbf{u}_0, \lambda_0)$, that is, rank $(\mathbf{f}_{\mathbf{u}}(\mathbf{u}_0, \lambda_0) | \mathbf{f}_{\lambda}(\mathbf{u}_0, \lambda_0)) = n$; and
- there is a parameterization $\mathbf{u}(\sigma), \lambda(\sigma)$ with $\mathbf{u}(\sigma_0) = \mathbf{u}_0, \lambda(\sigma_0) = \lambda_0$, and $d^2\lambda(\sigma_0)/d\sigma^2 \neq 0$.

For further details, see [81].

3.2 Hopf bifurcation

The bifurcation that connects equilibria with periodic motion is the Hopf bifurcation.

Definition 3.5. A bifurcation from a branch of equilibria to a branch of periodic oscillations is called *Hopf bifurcation*.

Poincaré proved basic results about the Hopf bifurcation; in 1929, Andronov studied the planar case [1]. These early results refer to the bifurcation from equilibria to limit cycles as the Poincaré-Andronov-Hopf bifurcation. The name “Hopf bifurcation” is nowadays more commonly used, in recognition of the work of Hopf, who considered the n -dimensional case:

Theorem 3.6. (Hopf 1942 [28]) Assume for $\mathbf{f} \in C^2$

- $\mathbf{f}(\mathbf{u}_0, \lambda_0) = \mathbf{0}$;
- $\mathbf{f}_{\mathbf{u}}(\mathbf{u}_0, \lambda_0)$ has a simple pair of purely imaginary eigenvalues $\mu(\lambda_0) = \pm i\beta$ and no other eigenvalue with zero real part; and
- $\frac{d(\operatorname{Re} \mu(\lambda_0))}{d\lambda} \neq 0$.

Then there is a birth of limit cycles at (\mathbf{u}_0, λ) . The “initial period” (*i.e.*, in the limit toward the zero-amplitude oscillation) is

$$T_0 = \frac{2\pi}{\beta}.$$

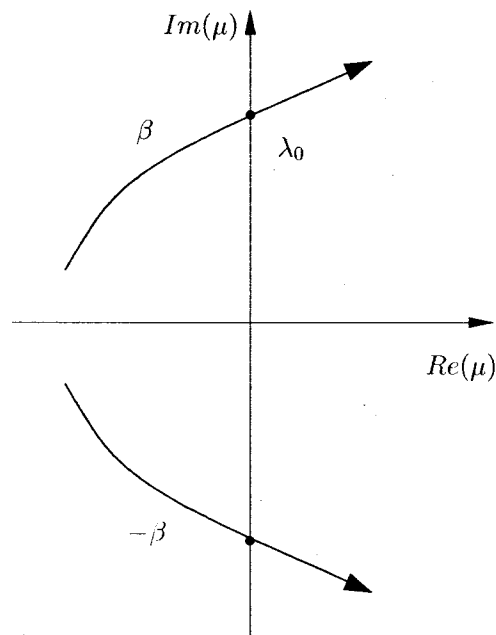


Figure 3.2: The path of eigenvalues of $\mathbf{f}_{\mathbf{u}}(\mathbf{u}(\lambda), \lambda)$ related to the Hopf bifurcation, as λ passes through λ_0 [81].

The Hopf bifurcation theorem is a local result. It guarantees the emergence of a small amplitude limit cycle branching from the fixed point. Figure (3.2) shows a possible path of the eigenvalues corresponding to Theorem (3.6-(2)). For further reference, see [81].

3.3 Homoclinic and heteroclinic orbits

A *heteroclinic orbit* (sometimes called a *heteroclinic connection*) is a trajectory in phase space which joins two different equilibrium points. If the equilibrium points at the start and the end of the orbit are the same, the orbit is a *homoclinic orbit* [52, 106]. Figure (3.3-a) shows the connection between two distinct saddle points; this is a heteroclinic orbit. Figure (3.3-b) shows a homoclinic orbit connecting a saddle point to itself. Figure (3.3-c) gives an example of a *Silnikov connection*, a homoclinic connection of a saddle-focus equilibrium in three-dimensional space. A sequence of heteroclinic orbits may form a closed path, called a *heteroclinic cycle*; see Figure (3.3-d).

We assume that $\mathbf{u}' = \mathbf{f}(\mathbf{u}, \lambda)$ has a homoclinic orbit at $\lambda = \lambda_0$.

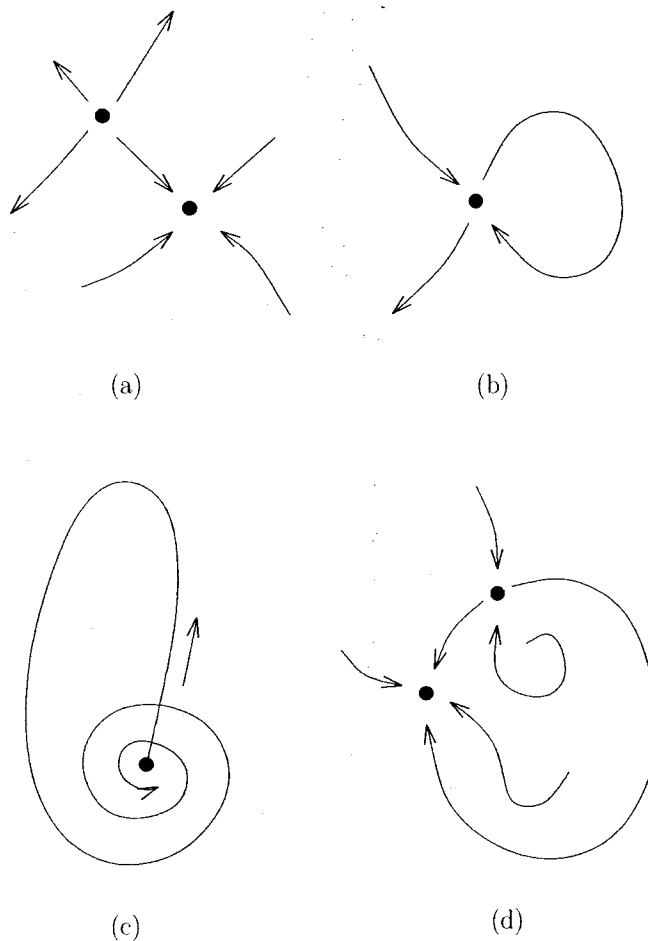


Figure 3.3: Heteroclinic and homoclinic orbits [81].

Definition 3.7. A branch of periodic orbits undergoes a *homoclinic bifurcation* at λ_0 if the orbits approach a homoclinic orbit for $\lambda \rightarrow \lambda_0$.

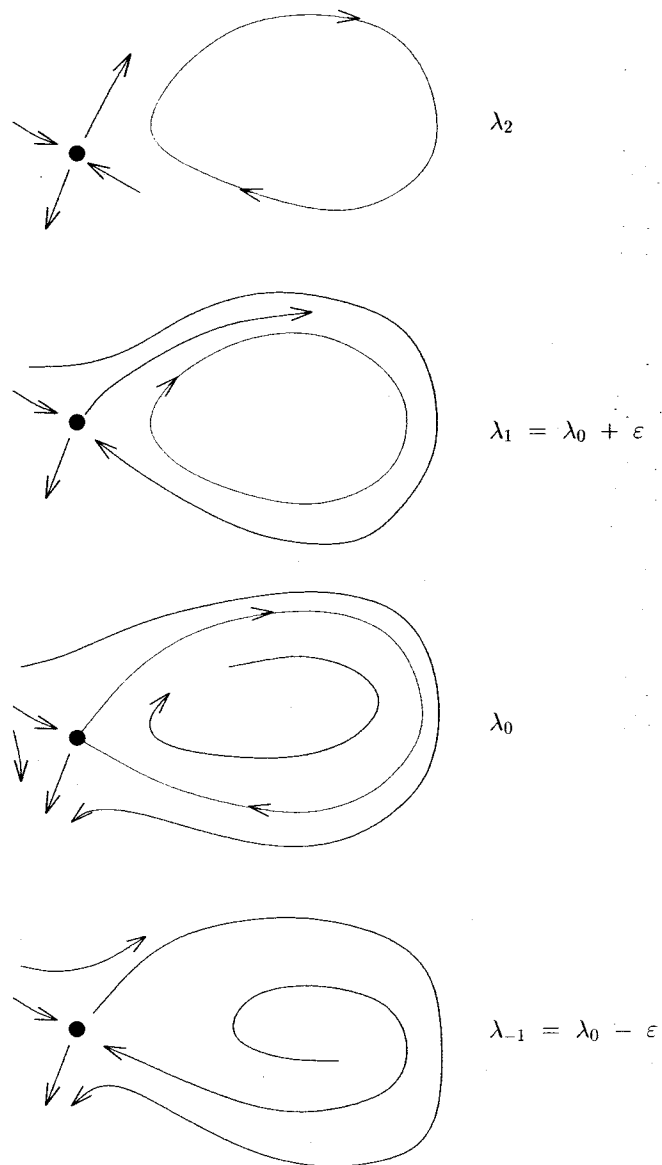


Figure 3.4: A homoclinic bifurcation at λ_0 [81].

In Figure (3.4), we illustrate how a saddle equilibrium and a periodic orbit get close to each other as $\lambda \rightarrow \lambda_0$.

Chapter 4

Computing the Lorenz Manifold

4.1 Introduction

4.1.1 Formalization of the system

In the early 1960s, the meteorologist Edward Lorenz started modeling the Earth's atmosphere. His work models the convective motion of a volume of air, which is warmed from below and cooled from above. His final model involves only three elementary equations:

$$\begin{cases} u_1' &= \sigma(u_2 - u_1) \quad , \\ u_2' &= \rho u_1 - u_2 - u_1 u_3 \quad , \\ u_3' &= u_1 u_2 - \beta u_3 \quad . \end{cases} \quad (4.1)$$

Here, we have $\mathbf{u} = (u_1, u_2, u_3)^T \in \mathbb{R}^3$; σ , ρ and $\beta \in \mathbb{R}$ are physical constants; u_1 is proportional to the intensity of the convective motion, and u_2 is proportional to the temperature difference between the ascending and descending currents. The third component u_3 is proportional to the distortion of vertical temperature profile from linearity. The Lorenz system (4.1) has the symmetry $(u_1, u_2, u_3) \rightarrow (-u_1, -u_2, u_3)$ of rotation by π radians about the u_3 -axis, *i.e.*, if $(u_1(t), u_2(t), u_3(t))$ is a solution, then $(-u_1(t), -u_2(t), u_3(t))$ is also a solution.

4.1.2 Elementary analysis

In the Lorenz system there are three system parameters ρ , σ and β , where ρ is perhaps the most important parameter. It is also called the *Rayleigh* number and it is proportional to the temperature difference across the layer, which is responsible for driving the fluid motion at a rate given by the variable u_1 , in the context of convection in a fluid or the atmosphere. The parameter σ is called the *Prandtl number*. The parameter β is positive. Usually σ and β are set to 10 and $8/3$ respectively. The “standard” value of ρ is 28. For these customary values, if we plot $u_3(t)$ against $u_1(t)$ as t varies, the famous “Lorenz butterfly” appears. It is interesting to observe that different starting points lead to quite different behavior. However the system evolves into a visually identifiable butterfly structure, which is called an *attractor*. The dynamics on the attractor [51] will be similar in outline, but much different in details depending on the initial conditions, that is, the dynamics is chaotic [44]. The attractor is a *fractal*, a geometric pattern that iterates infinitely often with recurring self-similarity. It is also called a *strange attractor*, in which the system exhibits chaotic behavior. We can see orbits go back and forth irregularly between the two “wings” of the “butterfly”. An important property of chaos is the sensitive dependence on initial conditions. The Lorenz system is a classic example of a vector field with a chaotic attractor. In this thesis, the study of chaos is not our main concern.

The computation of the Lorenz manifold, the two-dimensional stable manifold, is of interest in exhibiting the interesting behavior of the system. Different algorithms have been developed. In the discussion of Chapter 1, we mentioned different approaches to compute 2D stable and unstable manifolds, which apply to the Lorenz system. In fact, the motive of these methods was, in part, to study the Lorenz system better. We will consider the Lorenz system in more details in the following.

Obviously, the origin $\mathbf{0} = (0, 0, 0)^T$ is a stationary solution of Equation (4.1). The Jacobian matrix of \mathbf{f} is

$$\mathbf{f}_{\mathbf{u}}(\mathbf{u}) = \begin{bmatrix} -\sigma & \sigma & 0 \\ -u_3 + \rho & -1 & -u_1 \\ u_2 & u_1 & -\beta \end{bmatrix}.$$

Evaluating the Jacobian matrix at $\mathbf{0}$ we have

$$\mathbf{f}_{\mathbf{u}}(\mathbf{0}) = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho & -1 & 0 \\ 0 & 0 & -\beta \end{bmatrix},$$

so the characteristic polynomial of \mathbf{f} at the origin is

$$\lambda^3 + (1 + \sigma + \beta)\lambda^2 + (\beta\sigma + \beta + \sigma - \rho\sigma)\lambda + \beta\sigma(1 - \rho) = 0,$$

from which

$$(\lambda + \beta)[\lambda^2 + (\sigma + 1)\lambda + \sigma(1 - \rho)] = 0.$$

The eigenvalues are

$$\begin{aligned} \lambda_1 &= -\beta, \\ \lambda_{2,3} &= \frac{-(\sigma + 1) \pm \sqrt{(\sigma + 1)^2 - 4\sigma(1 - \rho)}}{2}. \end{aligned}$$

If $\rho < 1$, all three eigenvalues are negative and real. If $\rho > 1$, λ_2 becomes positive, so the stability of the origin changes from attracting to repelling. For instance, if we fix the parameters at the standard values $\rho = 28$, $\sigma = 10$ and $\beta = 8/3$, we will obtain one positive eigenvalue $\lambda_2^u \approx 11.828$ and two negative eigenvalues $\lambda_1^s \approx -2.667$ and $\lambda_3^s \approx -22.828$. The eigenvectors that correspond to the two stable eigenvalues span the stable eigenspace $E^s(\mathbf{0})$, and the Lorenz manifold $W^s(\mathbf{0})$ is tangent at $\mathbf{0}$ to the eigenspace $E^s(\mathbf{0})$. There are two more stationary solutions of Equation (4.1) when $\rho > 1$, namely,

$$\mathbf{p}^\pm = (\pm\sqrt{\beta(\rho - 1)}, \pm\sqrt{\beta(\rho - 1)}, \rho - 1),$$

approximately at $(\pm 8.485, \pm 8.485, 27)$ when $\rho = 28$, and symmetric to each other. Consider the Jacobian matrix at \mathbf{p}^+

$$\mathbf{f}_u(\mathbf{p}^+) = \begin{bmatrix} -\sigma & \sigma & 0 \\ 1 & -1 & -\sqrt{\beta(\rho-1)} \\ \sqrt{\beta(\rho-1)} & \sqrt{\beta(\rho-1)} & -\beta \end{bmatrix} ;$$

its characteristic polynomial is

$$\lambda^3 + (1 + \sigma + \beta)\lambda^2 + (\beta\sigma + \beta\rho)\lambda + 2\beta\sigma(\rho - 1) = 0 .$$

We can see, for the customary parameters values each of these two secondary equilibria has one negative eigenvalue and a pair of complex conjugate eigenvalues with positive real part. In fact there exists a critical value of ρ , where the stability of these equilibria changes from stable to unstable for ρ beyond that value [9, 92].

The origin $\mathbf{0}$ is stable for $\rho < 1$ and becomes a saddle in a pitchfork bifurcation at

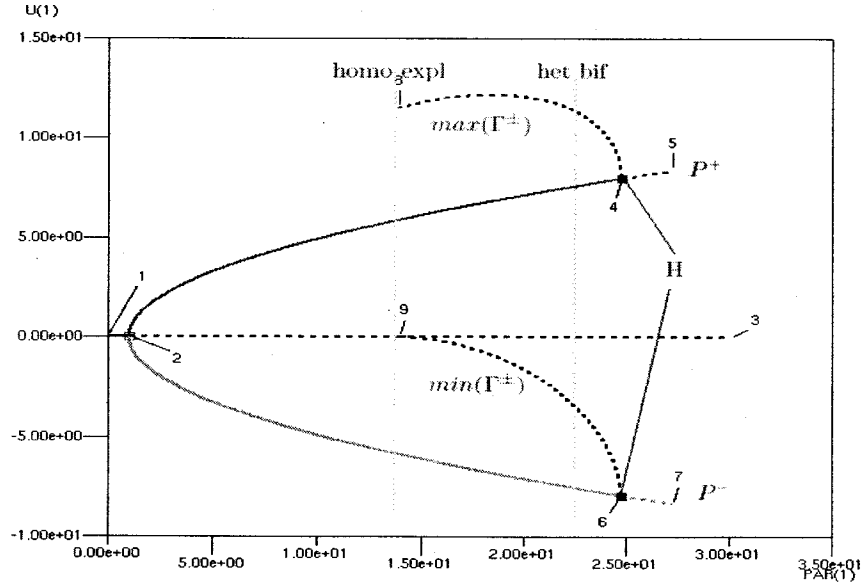


Figure 4.1: Bifurcation diagram of the Lorenz system: $\max u_1$ vs. ρ .

$\rho = 1$. For $\rho > 1$, $\mathbf{0}$ has a one-dimensional unstable manifold $W^u(\mathbf{0})$ and a two-dimensional stable manifold $W^s(\mathbf{0})$, which we refer to as the *Lorenz manifold*. For $\rho > 1$ the two

other secondary equilibria \mathbf{p}^\pm are each other's image under the symmetry. The secondary equilibria are attractors with a pair of complex conjugate eigenvalues until they lose stability in a subcritical Hopf bifurcation at

$$\rho_H = \sigma(\beta + \sigma + 3)(\sigma - \beta - 1)^{-1} = \frac{470}{19} \approx 24.7368.$$

Past the Hopf bifurcation for $\rho > \rho_H$ the equilibria \mathbf{p}^\pm are saddle points and have two-dimensional unstable manifolds $W^u(\mathbf{p}^\pm)$, associated with the complex conjugate eigenvalues with positive real parts and one-dimensional stable manifold $W^s(\mathbf{p}^\pm)$.

There exist two saddle periodic orbits Γ^\pm that bifurcate at the Hopf bifurcation in the interval $\rho_r < \rho < \rho_H$. They bifurcate at $\rho_r \approx 13.9265$ in a symmetrically related pair of homoclinic orbits, that are homoclinic to the origin; see Figure (4.1). This codimension-one homoclinic bifurcation is also known as a homoclinic explosion point. It is the source of all complicated dynamics in the Lorenz system: for $\rho > \rho_r \approx 13.9265$ we can find in a tubular neighborhood of the two homoclinic loops not only the two periodic orbits Γ^\pm but also infinitely many other saddle periodic orbits (of known symbolic description) [21, 74].

We note that the homoclinic explosion point at ρ_r does not produce a chaotic attractor. Rather one initially finds 'pre-turbulence' in the form of a chaotic transient before the system settles down to one of the attracting equilibria \mathbf{p}^\pm [21, 57]. At $\rho_{het} \approx 24.0579$ there are two codimension-one heteroclinic orbits that create a chaotic attractor. More precisely, at ρ_{het} there are two symmetrically related heteroclinic orbits between the origin $\mathbf{0}$ and the saddle periodic orbits Γ^\pm . For $\rho < \rho_{het}$, the unstable manifold $W^u(\mathbf{0})$ lies in the basin of attraction of the attractors \mathbf{p}^\pm . At the heteroclinic bifurcation at ρ_{het} the one-dimensional manifold $W^u(\mathbf{0})$ connects $\mathbf{0}$ to the saddle periodic orbits Γ^\pm , that is, $W^u(\mathbf{0})$ is entirely contained in $W^s(\Gamma^\pm)$. As a result, for $\rho > \rho_{het}$, $W^u(\mathbf{0})$ no longer lies in the basin of attraction of \mathbf{p}^\pm , but instead its closure is a chaotic attractor. This chaotic attractor coexists with the attracting equilibria \mathbf{p}^\pm until they disappear in the Hopf bifurcation at ρ_H . For $\rho > \rho_H$, the chaotic attractor is the only attractor. For further references see [9], [32] and [77].

4.2 Previous work

To study the phase portraits of a dynamical system, the stable and unstable manifolds associated with equilibrium points are important objects. It would appear that the computation of such manifolds can be carried out easily by numerical integration. This is often true for one-dimensional manifolds. However, we find that higher dimensional manifolds are difficult to compute using numerical integration. For example, a 2D stable manifold like the Lorenz system can exhibit the characteristics of complex geometric objects. For the standard system parameters, the origin has a two-dimensional stable manifold and each of the two secondary equilibria has a two-dimensional unstable manifold. The intersections of these two manifolds in the three-dimensional phase space form *heteroclinic connections* from the non-zero equilibrium to the origin. There are two difficulties in computing the Lorenz manifold [53]. For example, the case $\rho = 28$:

- “The stable eigenvalues at the origin of this system are approximately -2.667 and -22.828 , with a ratio that is approximately 8.56 . Trajectories in the manifold tend to follow the weakly stable direction, which makes it difficult to ‘cover’ the full manifold, even relatively near the origin” [77]. We observe a similar behavior from the two examples discussed in Chapter 2.
- “Part of the global manifold spirals around the z -axis while other parts of it curl around the stable manifolds of the equilibria located at approximately $(\pm 8.485, \pm 8.485, 27)$ ” [77].

The manifold will approach itself arbitrarily closely, in fact infinitely often, in certain areas of phase space. It is practically impossible to reliably compute a large portion of the manifold with techniques that advance its computational boundary based on local information near this boundary. In fact, such methods are prone to “sheet jumping” as we illustrated in Chapter 2. Many papers address the geometry of the Lorenz manifold because of its astonishing properties. Perelló [20] computed the stable manifold of the origin as a function of ρ , and also gave a sketch for ρ close to the critical value $\rho_c \approx 24.74$. He also considered the unstable manifold of the non-zero equilibria. It was obtained by following a line

segment, which is quite close to the approach of computing the unstable manifold of the non-zero equilibria in [77]. Thompson and Stewart [60] computed trajectories that illustrate the local stable manifold. Based on this, a more advanced visualization was done by Stewart [42]; the dynamics and global bifurcations of the Lorenz system can be observed there in the three-dimensional phase space. The first hand-drawn image of the Lorenz manifold computed under the standard parameter values appeared in the book of Abraham and Shaw [79] in 1985, while Guckenheimer and Worfolk's article [49] is the first paper that has computer-generated images of the Lorenz manifold. For more details, see [9].

4.3 Computing the stable manifold of the origin

In Chapter 2, we introduced the concept of orbit continuation and the algorithm to span a 2D manifold starting from an equilibrium point. Here we discuss the spanning procedure further, as we use orbit continuation to compute the Lorenz manifold, which is far more complex than the examples given in Chapter 2.

4.3.1 The method for computing the stable manifold of the origin

We use numerical continuation as the method for computing the stable manifold of the origin. Consider the ODEs representing the Lorenz equations, with the standard parameter values, written as

$$\mathbf{u}' = \mathbf{f}(\mathbf{u}(t)) , \quad t \in [0, T] . \quad (4.2)$$

Since the integration time T will be variable, we introduce a new, scaled time variable \hat{t} to let the integration always take place over the interval $[0, 1]$. The transformation is as follows

$$\hat{t} \equiv \frac{t}{T}, \quad \hat{t} \in [0, 1] .$$

Then

$$\mathbf{u}'(t) = \frac{d\mathbf{u}}{dt} = \frac{d\mathbf{u}}{d\hat{t}} \frac{d\hat{t}}{dt} = \frac{d\mathbf{u}}{d\hat{t}} \frac{1}{T} .$$

Let

$$\hat{\mathbf{u}}(\hat{t}) = \mathbf{u}(t(\hat{t})) ,$$

so that

$$\frac{d\hat{\mathbf{u}}}{d\hat{t}} = \mathbf{f}(\hat{\mathbf{u}}(\hat{t})),$$

or

$$\frac{d\hat{\mathbf{u}}}{d\hat{t}} = T\mathbf{f}(\hat{\mathbf{u}}(\hat{t})).$$

Dropping the “ $\hat{\cdot}$ ” the transformed equation is

$$\mathbf{u}'(t) = T\mathbf{f}(\mathbf{u}(t)), \quad t \in [0, 1]. \quad (4.3)$$

Note that the original integration time T is now an explicit variable in the equations. As mentioned before, the origin $\mathbf{0} = (0, 0, 0)^T$ is a saddle point, for $\rho = 28$. The eigenvalues are $\mu_1 \approx -2.66$, $\mu_2 \approx -22.8$, $\mu_3 \approx 11.82$, and corresponding normalized eigenvectors \mathbf{v}_1 , \mathbf{v}_2 and \mathbf{v}_3 . In the AUTO implementation, the parameter ρ can be changed in the equations file ‘man.f’ file; see Appendix (A). The computational procedure is as follows. First, suppose that an initial orbit $\mathbf{u}_0(t)$ has already been computed, for t from 0 to T_0 (where $T_0 < 0$, since we deal with a *stable* manifold here), with $\mathbf{u}_0(0)$ close to the origin $\mathbf{0}$, and $\mathbf{u}_0(0)$ in the stable eigenspace spanned by \mathbf{v}_1 and \mathbf{v}_2 , or, more precisely,

$$\mathbf{u}_0(0) = \mathbf{0} + \epsilon \left(\frac{\cos(\theta)}{|\mu_1|} \mathbf{v}_1 - \frac{\sin(\theta)}{|\mu_2|} \mathbf{v}_2 \right),$$

where ϵ is a small radius in the stable eigenspace centered at the origin. Starting data include a value of θ between 0 and 2π . If we take $\theta = 0$, then the initial orbit satisfies Equation (4.3) and

$$\mathbf{u}_0(0) = \frac{\epsilon}{|\mu_1|} \mathbf{v}_1.$$

The starting orbit $\mathbf{u}(t)$ has length

$$L_0 = T_0 \int_0^1 \|\mathbf{f}(\mathbf{u}_0(s))\| ds.$$

In other words, for given $\epsilon = \epsilon_0$ and $L = L_0$, the initial orbit is a solution $\mathbf{X}_0 = (\mathbf{u}_0(\cdot), \theta_0, T_0)$, with $\theta_0 = 0$, of the equation $\mathbf{F}(\mathbf{X}) = \mathbf{0}$, where $\mathbf{X} = (\mathbf{u}(\cdot), \theta, T)$. Here $\mathbf{F}(\mathbf{X})$ can be written

as

$$\mathbf{F}(\mathbf{X}) \equiv \begin{cases} \mathbf{u}'(t) - T\mathbf{f}(\mathbf{u}(t)) \\ \mathbf{u}(0) - \epsilon\left(\frac{\cos(\theta)}{|\mu_1|}\mathbf{v}_1 - \frac{\sin(\theta)}{|\mu_2|}\mathbf{v}_2\right) \\ T \int_0^1 \|\mathbf{f}(\mathbf{u}(s))\| ds - L \end{cases} \quad (4.4)$$

Given \mathbf{X}_0 and $\dot{\mathbf{X}}_0$, with $\|\dot{\mathbf{X}}_0\| = 1$, pseudo-arclength continuation can now be used to compute a next solution \mathbf{X}_1 , *etc.* . In this way, step by step, the whole manifold can be covered. Given \mathbf{X}_i , the next solution \mathbf{X}_{i+1} is obtained by solving

$$\begin{cases} \mathbf{F}(\mathbf{X}_{i+1}) = \mathbf{0}, \\ \langle \mathbf{X}_{i+1} - \mathbf{X}_i, \dot{\mathbf{X}}_i \rangle - \Delta s = 0, \end{cases}$$

where $i \geq 0$. This is the so-called *orbit continuation* referred to in Chapter 2. There are possible variations on the basic numerical scheme. For example, instead of fixing L and allowing T to vary, one can fix T , and allow L to vary. In this case, $\mathbf{F}(\mathbf{X})$ still takes the form of Equation (4.4), but with $\mathbf{X} = (\mathbf{u}(\cdot), \theta, L)$, for given ϵ and T . Alternatively, one can fix one coordinate or a function of the coordinates at a particular value to constrain the end point, and again free both T and θ . This is done by including an appropriate function g [9]; thus $\mathbf{F}(\mathbf{X})$ becomes

$$\mathbf{F}(\mathbf{X}) \equiv \begin{cases} \mathbf{u}'(t) - T\mathbf{f}(\mathbf{u}(t)) \\ \mathbf{u}(0) - \epsilon\left(\frac{\cos(\theta)}{|\mu_1|}\mathbf{v}_1 - \frac{\sin(\theta)}{|\mu_2|}\mathbf{v}_2\right) \\ g(\mathbf{u}(1), T) - \alpha \end{cases} \quad (4.5)$$

In the computation, we sometimes want the orbits be computed with a fixed T value or a fixed L value. We call such values the *nominal time* or the *nominal arclength*, denoted by T_{nom} or L_{nom} . However, there are cases where we cannot use the exact nominal values. For example, when during the continuation the orbit is approaching an equilibrium, the value of T will approach ∞ . If we compute the orbit with fixed arclength L_{nom} , the continuation will not end. On the other hand, if we set a fixed time T , some orbits will be too long and some too short. Thus we set a constant $C_{nom} \equiv (L_{nom} - L_0) \times |T_0 - T_{nom}|$. An alternate

boundary condition is then

$$C_{nom} - (L_{nom} - L) \times |T - T_{nom}| = 0.$$

Thus the complete $\mathbf{F}(\mathbf{X})$ becomes

$$\mathbf{F}(\mathbf{X}) \equiv \begin{cases} \mathbf{u}'(t) - T\mathbf{f}(\mathbf{u}(t)) \\ \mathbf{u}(0) - \epsilon \left(\frac{\cos(\theta)}{|\mu_1|} \mathbf{v}_1 - \frac{\sin(\theta)}{|\mu_2|} \mathbf{v}_2 \right) \\ T \int_0^1 \|\mathbf{f}(\mathbf{u}(s))\| ds - L \\ C_{nom} - (L_{nom} - L) \times |T - T_{nom}|, \end{cases} \quad (4.6)$$

with $\mathbf{X} \equiv (\mathbf{u}(\cdot), \theta, T, L, (L_{nom} - L) \times |T - T_{nom}|)$;

4.3.2 Numerical results

In Figure (4.2) we show the Lorenz manifold $W^s(\mathbf{0})$ for four different ρ values. The manifolds are colored according to the geodesic distance to the origin, and the nominal arclength is set to be 200. The stable manifold $W^s(\mathbf{0})$ is the smooth image of a disc that simultaneously ‘rolls’ into the right and left ‘wing’ of the Lorenz attractor. It has in a primary helix along the positive z -axis [11, 44, 45]. We can observe that the torsion of the helix around the positive z -axis increases as ρ decreases [32]. Although not shown in this figure, we also observe that the larger the value ρ the more initial-condition sensitive character the system becomes. In Figure (4.3) we show two manifolds, one with $\rho = 15$ and another one with $\rho = 60$, together in one figure. It should give a better sense of how a planar disc ‘rolls up’ under the evolution laws. At the same time, we also see more clearly the role that ρ plays.

4.4 Locating heteroclinic connections in the Lorenz manifold

We have discussed the method of covering the 2D stable manifold of the Lorenz system of the equilibrium $\mathbf{0}$. Now we consider how to locate heteroclinic connections in the manifold we computed.

4.4.1 The existence of homoclinic orbits and heteroclinic orbits

The existence of a homoclinic orbit in the Lorenz system was proved in the early 1990s. Hastings and Troy [86, 87] show that for each (σ, β) in some neighborhood of $(10, 1)$, there is a ρ in the range $(1, 1000)$ for which the Lorenz system has a homoclinic orbit. Hassard and Zhang [5] gave a rigorous proof. Their work is based on Sparrow's [21] numerical shooting argument that a homoclinic orbit exists. Chen [107] gave a necessary and sufficient condition on the parameters (σ, β) for the Lorenz system to have a homoclinic orbit for some ρ . To be more explicit, he proved that there is a homoclinic orbit to the origin for some $\rho > 0$ if and only if $\sigma > \frac{2\beta+1}{3}$. It also follows from this proof that the homoclinic orbit converges to a singularly degenerate heteroclinic cycle as $\rho \rightarrow \infty$ and $\sigma \rightarrow (2\beta + 1)/3$. Hiroshi and Robert [43] give a system that possesses a singularly degenerate heteroclinic cycle, adding to the mathematical understanding of its structure. They also show that their system can be considered as a small perturbation of the original Lorenz system.

4.4.2 The continuation procedure

The Lorenz system has two secondary equilibria \mathbf{p}^+ and \mathbf{p}^- . We want to compute the connection from the zero equilibrium \mathbf{p}^0 to \mathbf{p}^+ or \mathbf{p}^- . Let \mathbf{p}^x denote either choice of the two secondary equilibria. Let \mathbf{E}_{sp}^u be the unstable eigenspace associated with \mathbf{p}^x . Let \mathbf{w}_x denote the eigenvector associated with the stable eigenvalue of the transpose Jacobian at \mathbf{p}^x . At each continuation, we compute an orbit starting at $\mathbf{u}(0)$ ending at $\mathbf{u}(1)$. We define

$$\tau_x = \langle \mathbf{u}(1) - \mathbf{p}^x, \mathbf{w}_x \rangle.$$

See Figure (4.4). If τ_x is equal to zero, an orbit's end is in the plane \mathbf{E}_{sp}^u . We let d_0 denote the (very small) distance of the end point of the orbit to \mathbf{p}^x ; see the notation in Figure (4.4). We clarify the "numerical" detection of a heteroclinic connection as follows.

The conditions for the detection of a heteroclinic orbit are

- (1) $|\mathbf{u}(1) - \mathbf{p}^x| = d_0$, where d_0 is small;
- (2) $\tau_x = 0$.

We reformulate $\mathbf{F}(\mathbf{X})$ as follows

$$\mathbf{F}(\mathbf{X}) \equiv \begin{cases} \mathbf{u}'(t) - T\mathbf{f}(\mathbf{u}(t)) \\ \mathbf{u}(0) - \epsilon\left(\frac{\cos(\theta)}{|\mu_1|}\mathbf{v}_1 - \frac{\sin(\theta)}{|\mu_2|}\mathbf{v}_2\right) \\ T \int_0^1 \|\mathbf{f}(\mathbf{u}(s))\| ds - L \\ \|\mathbf{u}(1) - \mathbf{p}^x\| - d_0 \\ \langle \mathbf{u}(1) - \mathbf{p}^x, \mathbf{w}_x \rangle - \tau_x \\ C_{nom} - (L_{nom} - L) \times |T - T_{nom}|. \end{cases} \quad (4.7)$$

Both conditions (1) and (2) are necessary for the orbit's end to approximate \mathbf{p}^x . Condition (1) is sufficient for a heteroclinic connection. However, when $\mathbf{u}(1)$ approaches an equilibrium, the continuation can take infinitely long time. Thus condition (1) with very small d_0 is not a good boundary condition. We can use condition (2) to compute many more *candidate orbits*. These candidates orbits' ends lie in \mathbf{E}_{sp}^u . They may stay in \mathbf{E}_{sp}^u and evolve toward the equilibrium or they may leave \mathbf{E}_{sp}^u . In the AUTO implementation, we compute many candidate orbits and then select those orbits that stay in \mathbf{E}_{sp}^u and approach the respective equilibrium.

Thus the AUTO continuation consists of two runs. Before these two runs, we will compute an initial orbit, where $(L_{nom} - L) \times |T - T_{nom}|$ is computed to C_{nom} . In the first run, we use condition (2) as the boundary condition. In the second run, we fix the continuation parameter such that $\tau_x = 0$, and try to increase time $|T|$. Since we are computing the stable manifold, T is less than zero. The continuation ends when condition (1) is met. Such a setting ensures that the orbit stays in \mathbf{E}_{sp}^u .

Figure (4.5) show the paths that the orbits' ends sweep in the \mathbf{E}_{sp}^u of \mathbf{p}^+ and \mathbf{p}^- . The path in the plane of the eigenspace of \mathbf{p}^+ is indicated by small red dots. The path on the plane of the eigenspace of \mathbf{p}^- is indicated by small blue dots. The two green dots represent the two secondary equilibria.

In Figure (4.6), we show 1,024 computed heteroclinic connections. The length of the connections' symbol sequence is ten. The concrete implementation can be seen in the FORTRAN file and Python scripts in Appendix (A).

4.4.3 The symbol sequence of the heteroclinic connections

In Figure (4.6), all heteroclinic connections are computed to the rim of a disk with radius 2.0 centered at either secondary equilibrium. The holes in the manifold could be filled in fact, but this would take more computer time. The orbits can circle an arbitrary number of times around an equilibrium. Each such revolution relates to a unique intersection point of the heteroclinic orbit with the section Σ_ρ , which is located at the height of the secondary equilibria. The flow of the orbit is required to be pointing upwards, that is, in the direction of increasing z . The precise notion of the revolution around \mathbf{p}^+ or \mathbf{p}^- enables us to introduce a symbolic coding. In [21] and [25] a similar coding has been used. In this text, we follow generally the coding used in [32]. However, in [32], heteroclinic orbits are computed from one of the secondary equilibria to the equilibrium at $\mathbf{0}$, which is just on the contrary direction of our computation. The first symbol is either an R or an L . R denotes the orbit passing section Σ_ρ of \mathbf{p}^+ infinitely many times. L denotes the orbit passing section Σ_ρ of \mathbf{p}^- infinitely many times. Then we add one or more l or r to the left of the first symbol. Here, l denotes the orbit passing section Σ_ρ of \mathbf{p}^- once, while r denotes the orbit passing section Σ_ρ of \mathbf{p}^+ once. In a symbol sequence, any l adjacent to L will of course be absorbed by L . Similarly any r will be absorbed by R . Thus there is no adjacent l to the left of L , no adjacent r to the left of R . We formalize the definition as the following.

Definition 4.1. For a given integer $N \geq 1$ (the length of the symbol sequence), we write the symbol sequence as

$$s_1 s_2 \dots s_N, \text{ where } \begin{cases} s_j = l \text{ if the } j\text{th passing of } \Sigma_\rho \text{ is near } \mathbf{p}^- , \\ s_j = r \text{ if the } j\text{th passing of } \Sigma_\rho \text{ is near } \mathbf{p}^+ . \end{cases}$$

Here, s_N is either R or L , which represents the infinitely many passes through Σ_ρ near \mathbf{p}^+ or \mathbf{p}^- respectively. We call such a sequence in the *verbose* form. Further, we use a *compact* form as

$$s_1 s_2 \dots s_{M-1} s_M, \text{ where } \begin{cases} \text{if } s_M = R, s_{M-1} \text{ is } l, \\ \text{if } s_M = L, s_{M-1} \text{ is } r. \end{cases}$$

Here, $M \leq N$, $s_M = s_N$ and if $M - 1 \leq 0$ then $s_1 \dots s_{M-1}$ is empty.

For example, if the length of the symbol sequence is ten, then $lllrrrrrR$ is in the verbose form. $lllR$ is the compact form of $lllrrrrrR$, which is convenient for data handling. However, it is difficult to tell the symbol length, for example, $lllR$ may represent $lllrrR$ of a sequence of length seven. It may also represent a length ten sequence $lllrrrrrR$. So it is necessary to specify the sequence length when giving a symbol sequence.

In [32], 512 heteroclinic connections from \mathbf{p}^+ to $\mathbf{0}$ are computed. These heteroclinic orbits occur in the *recursive ordering of a full binary tree*. By contrast, in this thesis, we start the computation from $\mathbf{0}$. We fully span the stable manifold of $\mathbf{0}$ and extract 1,024 heteroclinic connections with symbol sequence length ten. As in [32], our results can be presented in Figure (4.8).

In our computation, shown in Figure (4.8), the beginning level is the empty set $\mathbf{0}$. The first heteroclinic connection is always L , no matter the length of symbol sequence. The last heteroclinic connection is R . Then for any sequence, if the sequence goes one level down, it will first branch to an l , then branch to an r . At each deeper level, half of connections will have the same compact form as their parent sequences. The total number of connections at this level is twice that of the higher level. Therefore, there are a total of 2^n connections if the length of the symbol sequence is n . The symbol sequence is helpful for us to understand the folding features of the Lorenz manifold.

We plot the computation time length, L_2 norm, θ , the arclength, and the nominal constant C_{nom} of the orbits as a function of the scaled index in Figure (4.9) to Figure (4.13), respectively. In each figure, a black dot represents a heteroclinic connection. The scaled index is calculated as follows. Suppose we set the length of the symbol sequence as n , so we have a total of 2^n orbits. The scaled index of orbit m ($1 \leq m \leq 2^n$) is $(m-1)/(2^n-1)$. We see that at a heteroclinic connection the L_2 norm and the arclength have local minima; while the time T and the nominal constant $(L_{nom} - L) \times |T - T_{nom}|$ have local maxima. In addition, we also observe that θ , which is a very sensitive parameter in the system, has local maxima and local minima in two distinct regions in Figure (4.13), respectively.

The computation of the full series of heteroclinic orbits of a given a symbol sequence length is important in the study of the combinatorics of limiting homoclinic orbits that are encountered when following the heteroclinic connections for varying parameter ρ . In this

thesis, we only compute orbits with the typical setting of the Lorenz system ($\rho = 28$, $\sigma = 10$ and $\beta = 8/3$). Similar computations would make it possible to study the dependence of the Lorenz manifold, $W^s(\mathbf{0})$, and its relation to the unstable manifolds $W^u(\mathbf{p}^\pm)$ and $W^u(\Gamma^\pm)$, on the other parameters β and σ .

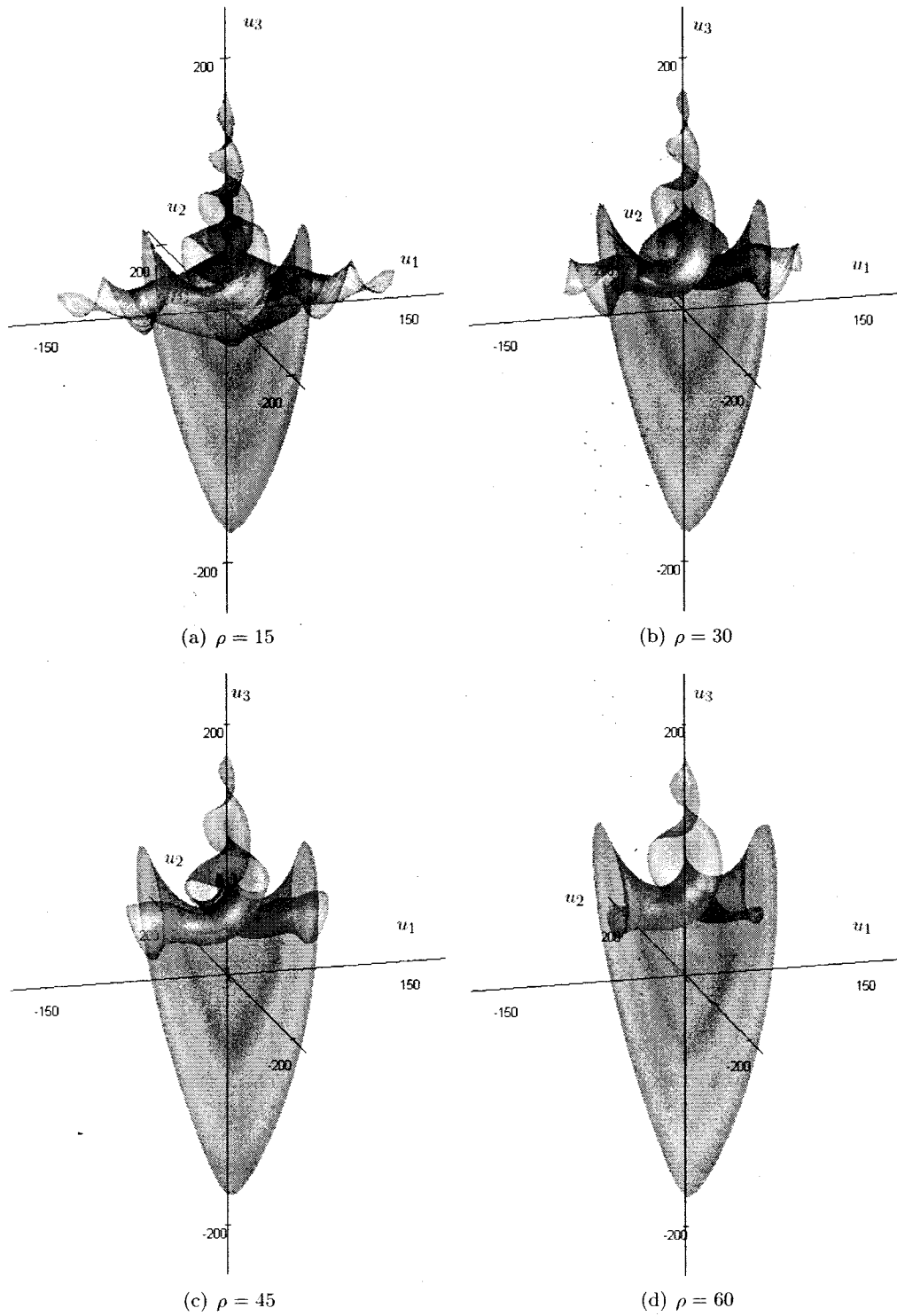


Figure 4.2: The Lorenz manifold with different ρ , computed using $T_{nom} = -7$, $L_{nom} = 200$

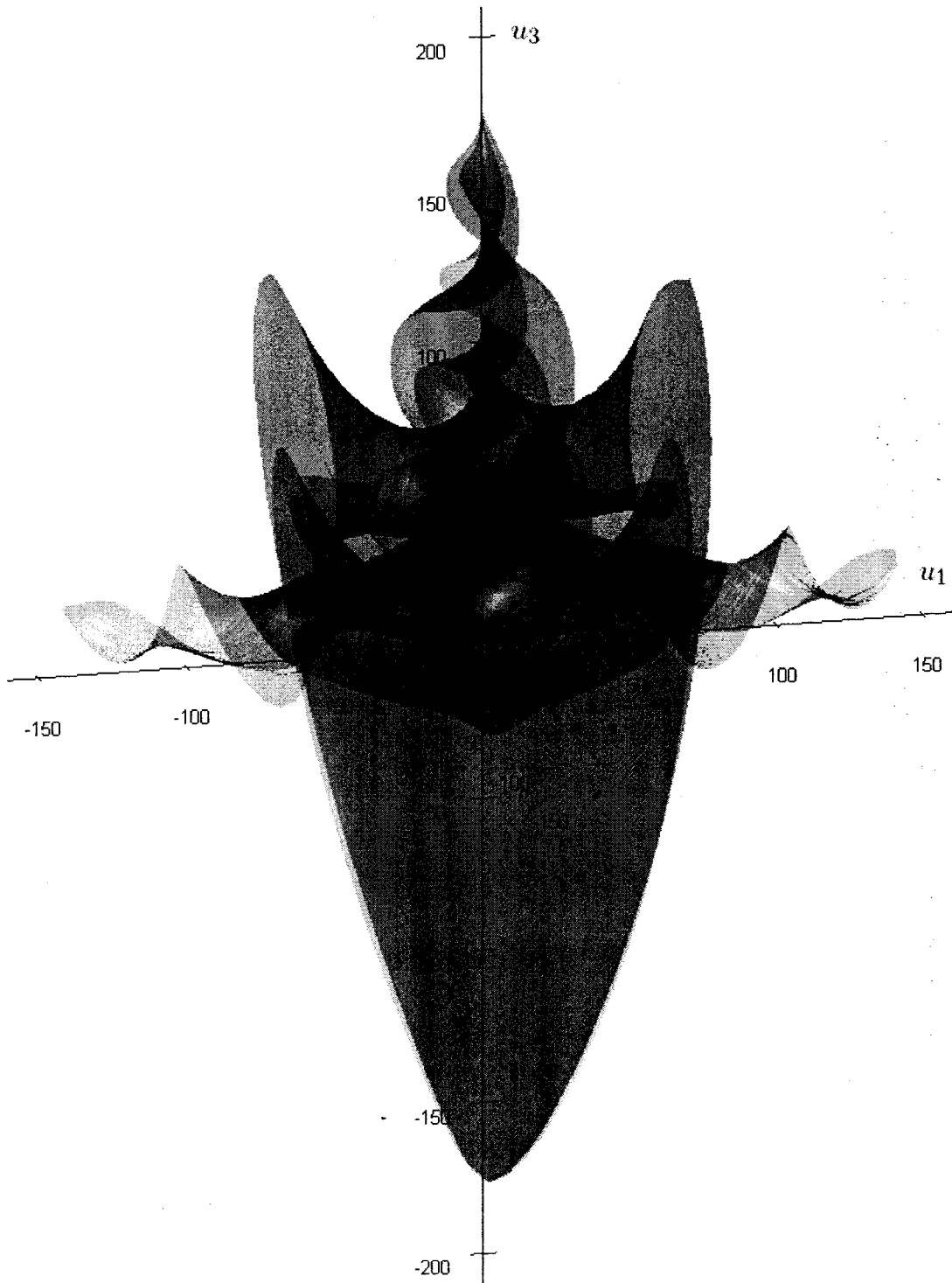


Figure 4.3: Comparing the Lorenz manifolds for $\rho = 15$ and $\rho = 60$.

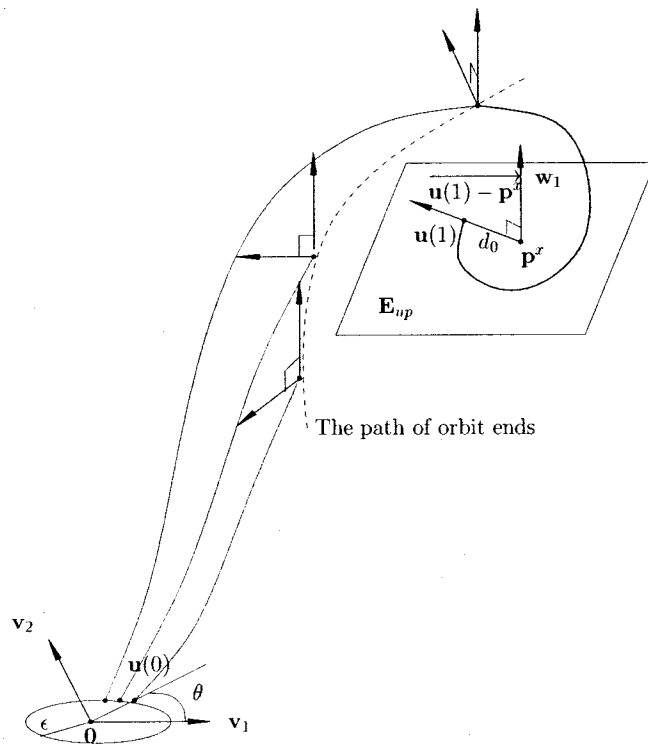


Figure 4.4: Detecting a heteroclinic orbit.

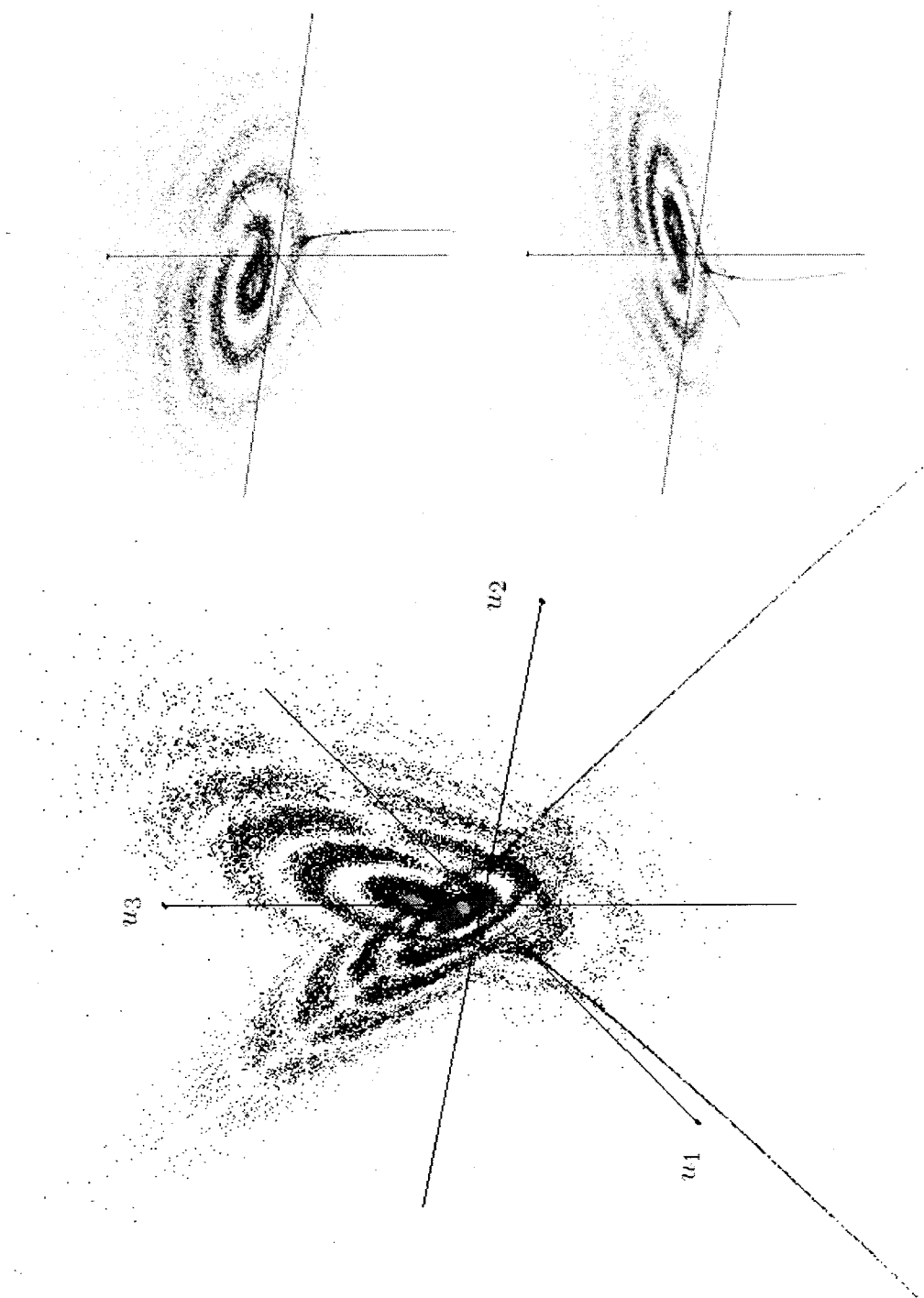


Figure 4.5: The path of candidate orbits' end points.

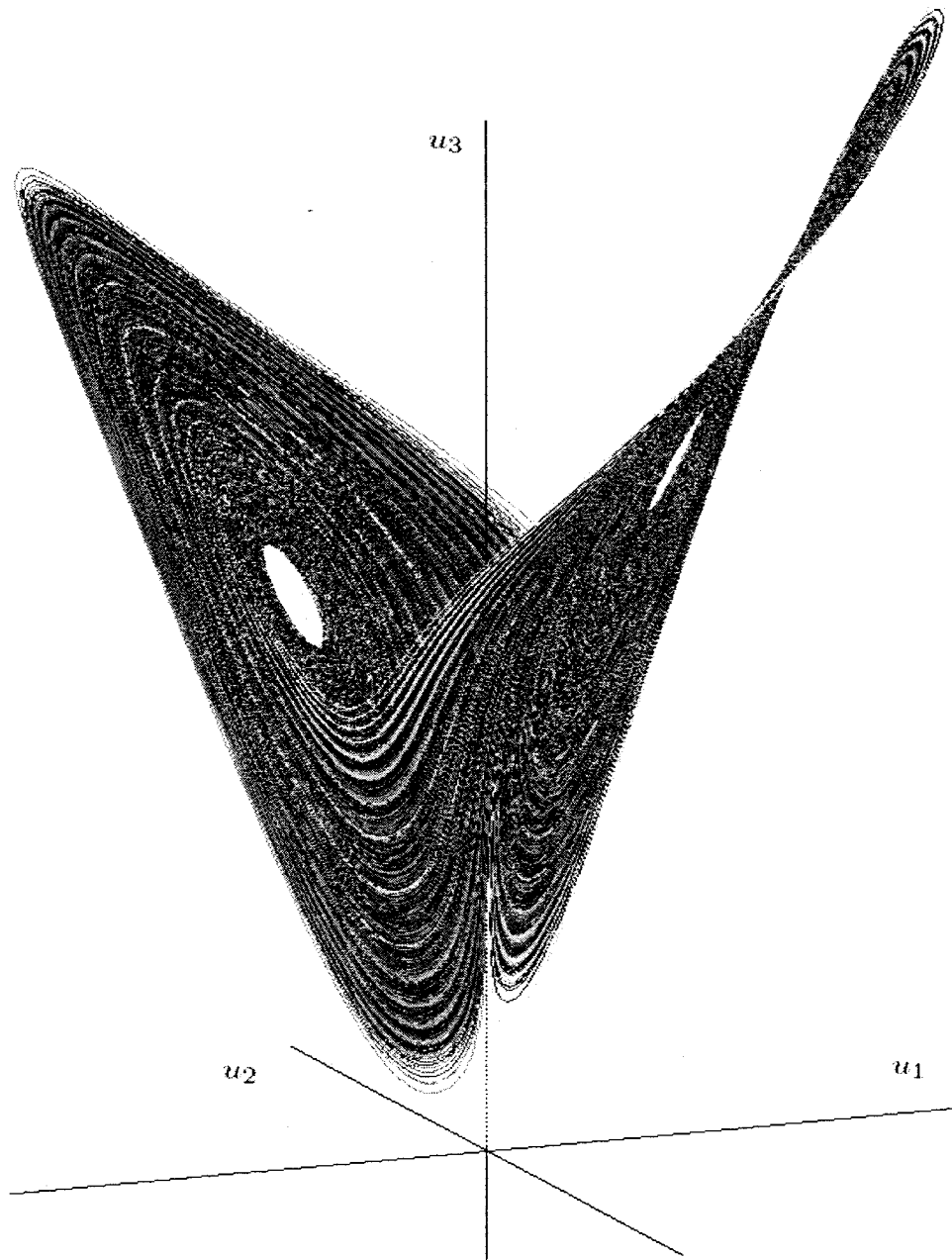


Figure 4.6: 1024 heteroclinic orbits.

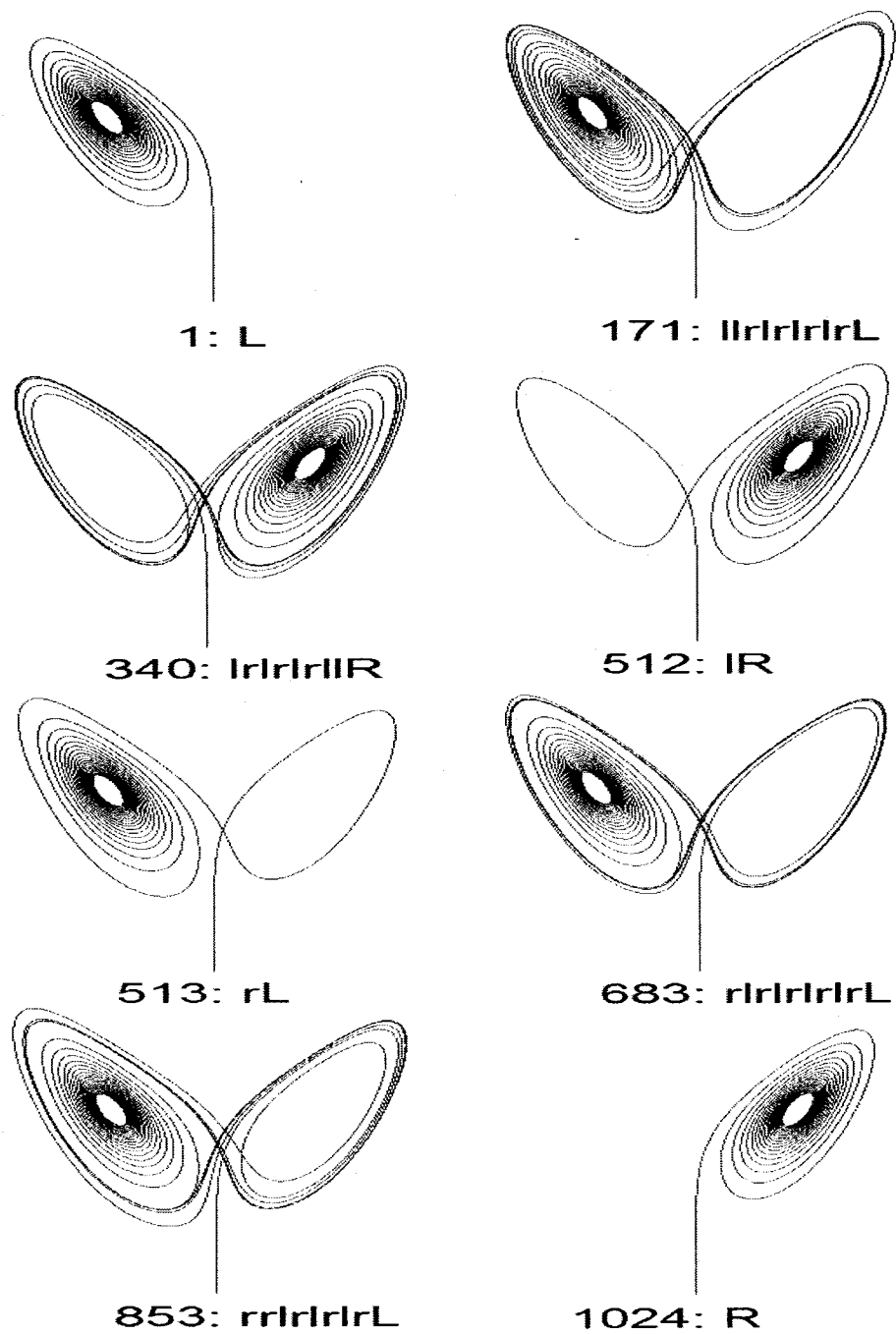


Figure 4.7: Selected heteroclinic orbits (their symbols are in the compact form).

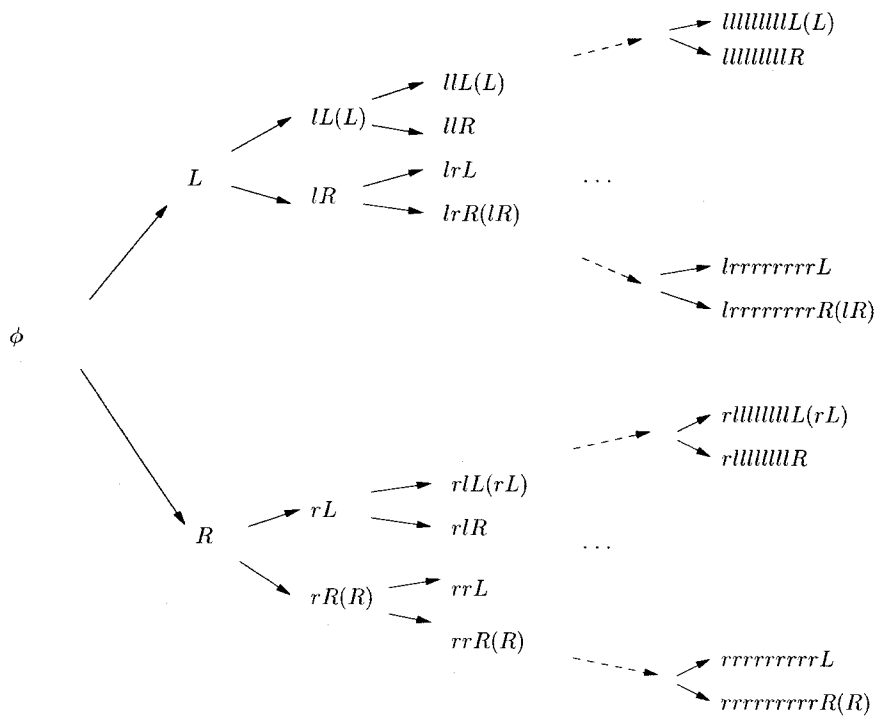


Figure 4.8: The relation between symbol sequences and their lengths.

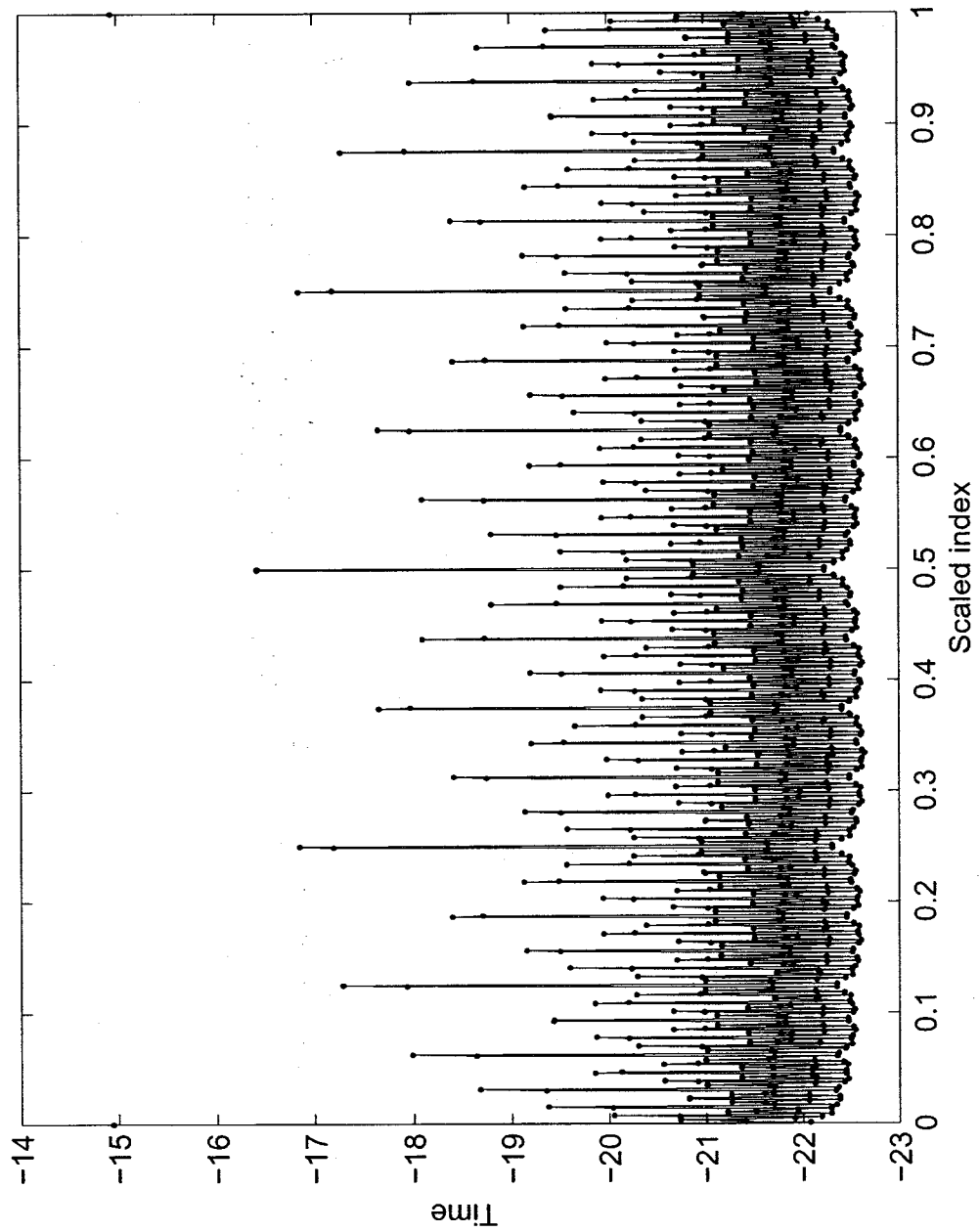


Figure 4.9: Scaled orbit indices *vs.* time.

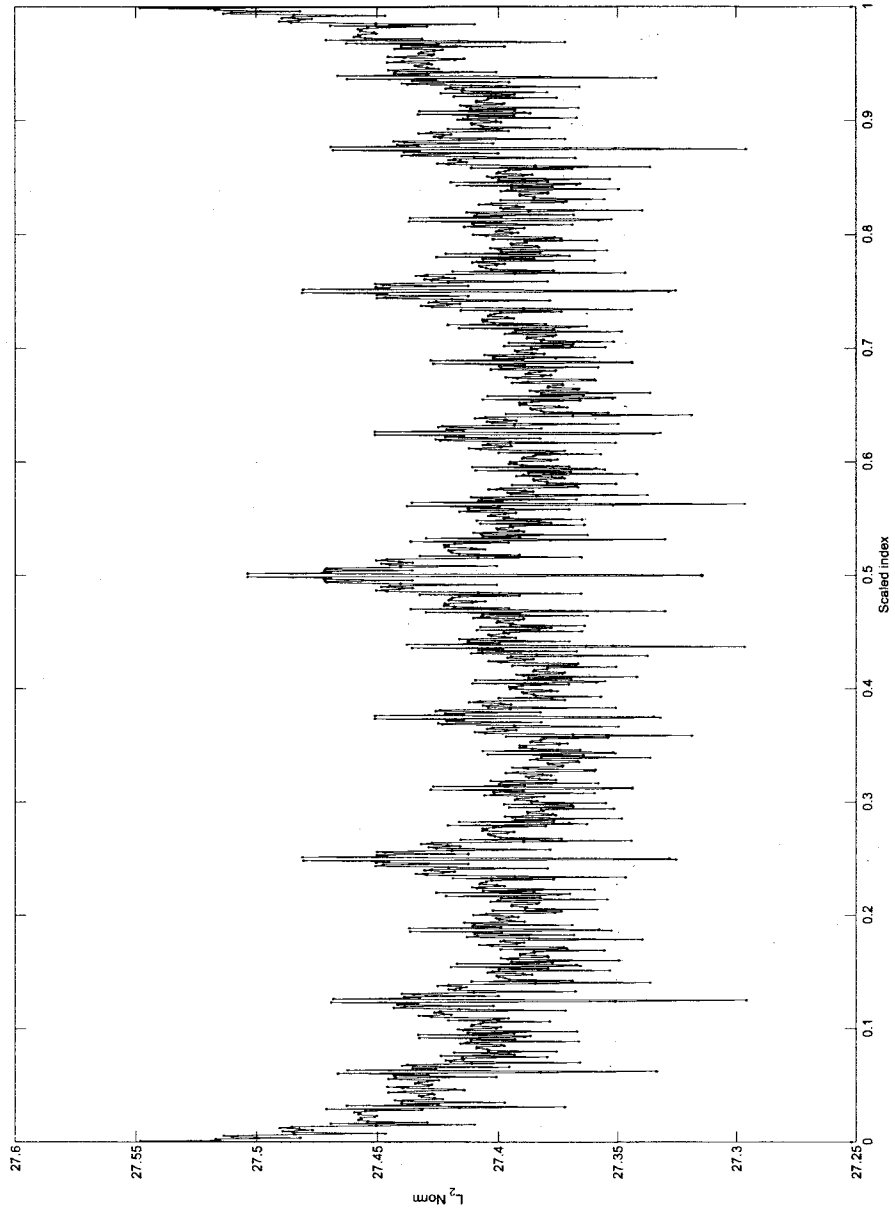


Figure 4.10: Scaled orbit indices *vs.* L_2 norm.

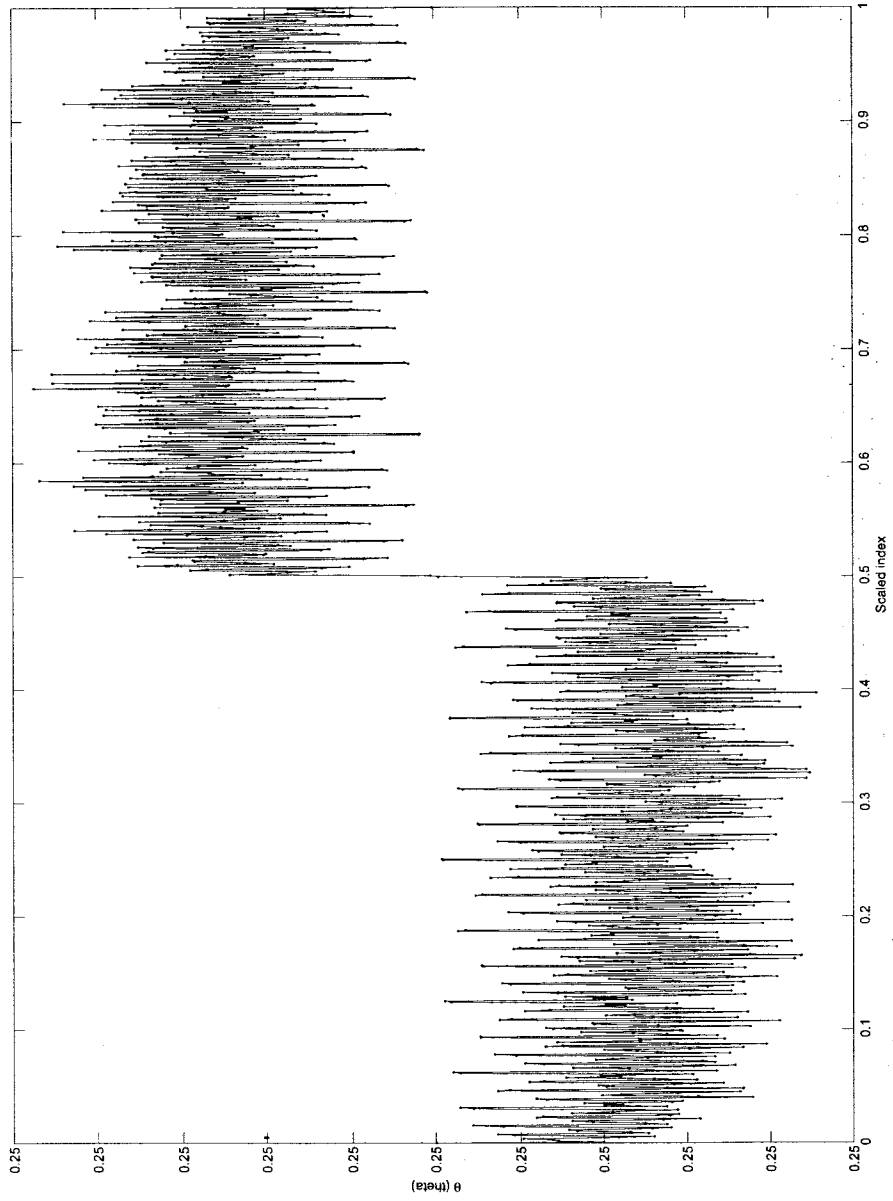


Figure 4.11: Scaled orbit indices *vs.* θ .

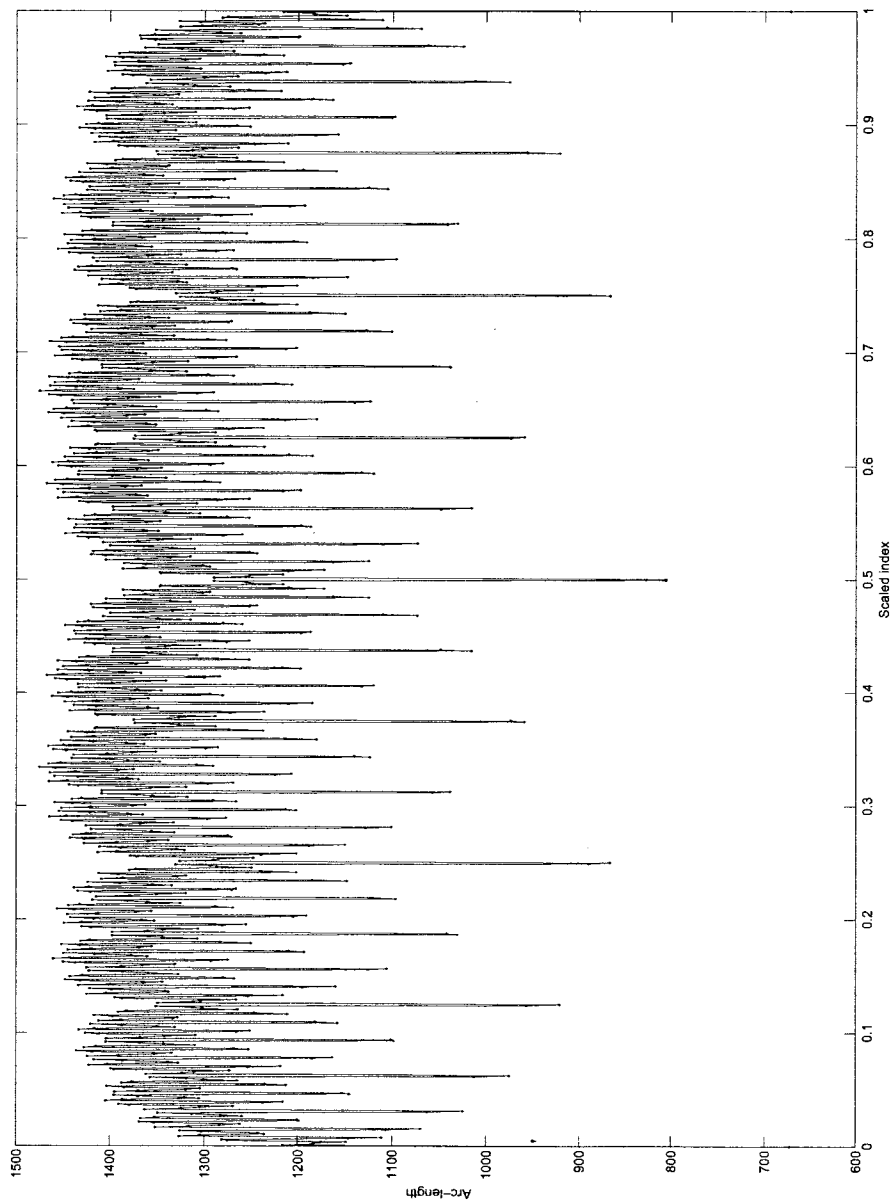


Figure 4.12: Scaled orbit indices *vs.* arclength.

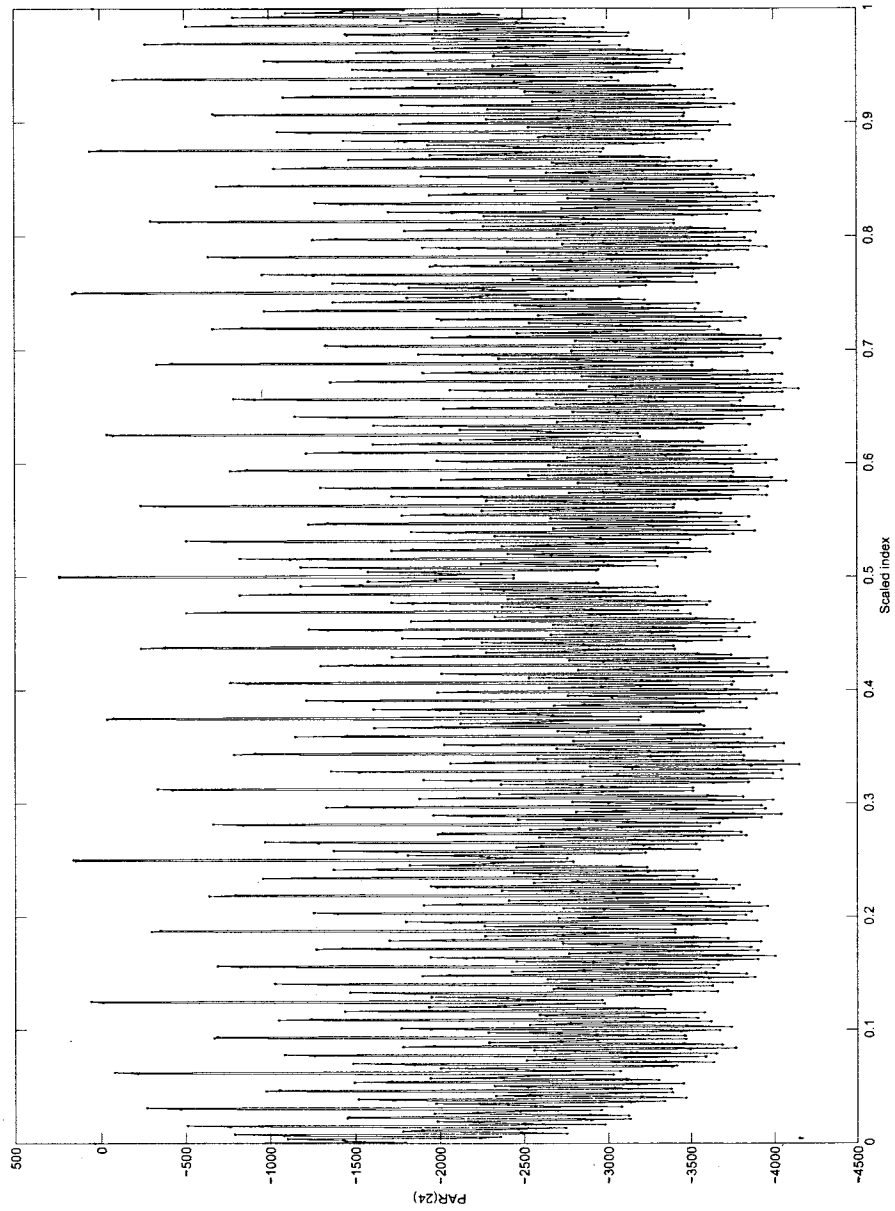


Figure 4.13: Scaled orbit indices *vs.* $(L_{nom} - L) \times |T - T_{nom}|$.

Chapter 5

The Stability of Periodic Solutions

In this chapter, we discuss the stability analysis of periodic solutions in dynamical systems. The analysis is immediately applied in the computations of Chapter 6. For further reference of this chapter see Chapter 7 of [81].

5.1 Periodic solutions of autonomous systems

Periodic solutions are time-dependent or space-dependent orbits that “cycle” (or “oscillate”). Our main concern is time-dependent periodicity of solutions of autonomous system of ODEs

$$\mathbf{u}'(t) = \mathbf{f}(\mathbf{u}, \lambda), \quad \mathbf{f} : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n. \quad (5.1)$$

For time-periodic solutions, the minimum time interval $T \in \mathbb{R}^+$ is called the *period* for the system to return to the original state . We have $\mathbf{u}(t+T) = \mathbf{u}(t)$ for all t . Equation (5.1) is autonomous, thus whenever $\mathbf{u}(t)$ is a solution, $\mathbf{u}(t+\zeta)$ is also a solution for all constants ζ . Therefore if we want to measure the period T , which is usually not known beforehand, we can start at any point denoted by $\mathbf{u}(t_0)$. We assume an “initial” moment $t_0 = 0$. T must be calculated together with \mathbf{u} . Since T is the minimum period and $T > 0$, we let $\mathbf{u}(0) = \mathbf{u}(T)$.

In Chapter 2 we introduced stability of equilibrium points. The stability of periodic orbits can also be studied in a similar way. To analyze the stability of periodic solutions, we need the concepts of monodromy matrix and Poincaré map.

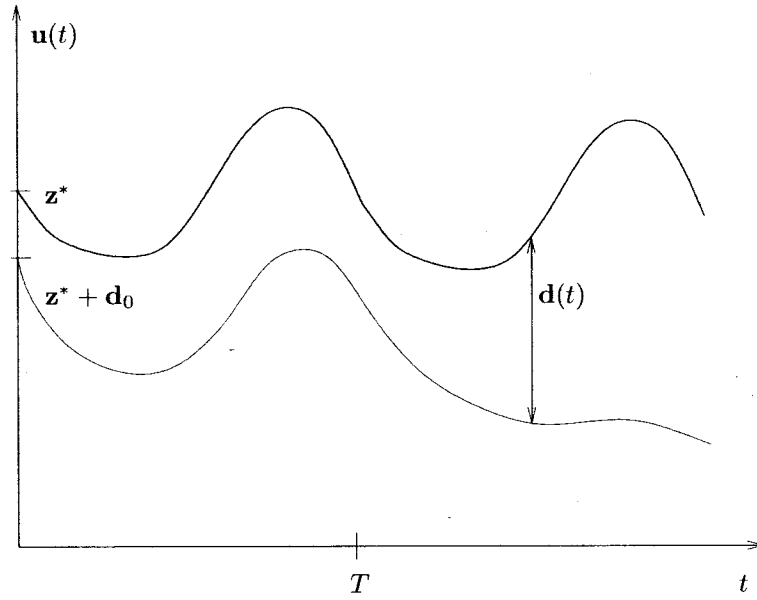


Figure 5.1: A periodic trajectory \mathbf{u}^* and a neighboring trajectory [81].

5.2 The monodromy matrix

We now investigate the stability of a particular periodic solution $\mathbf{u}^*(t)$. Its period is T and the system's dependence on λ is omitted.

The stability of \mathbf{u}^* depends on the behavior of its neighboring trajectories. We illustrate the key notations for our discussion in Figure(5.1). Let φ be a periodic trajectory such

$$\varphi(t, \mathbf{z}) \text{ solves Equation (5.1) with } \mathbf{z} = \mathbf{u}(0). \quad (5.2)$$

We define a distance vector with an initial distance vector \mathbf{d}_0 such

$$\mathbf{d}(t) = \varphi(t; \mathbf{z}^* + \mathbf{d}_0) - \varphi(t; \mathbf{z}^*).$$

Here $\mathbf{z}^* = \mathbf{u}^*(0)$. After one period, the distance is

$$\mathbf{d}(T) = \varphi(T; \mathbf{z}^* + \mathbf{d}_0) - \varphi(T, \mathbf{z}^*).$$

Taylor expansion gives

$$\mathbf{d}(T) = \frac{\partial \varphi(T; \mathbf{z}^*)}{\partial \mathbf{z}} \mathbf{d}_0 + \text{terms of higher order.}$$

Obviously, the matrix

$$\frac{\partial \varphi(T; \mathbf{z}^*)}{\partial \mathbf{z}} \tag{5.3}$$

decides whether the \mathbf{d}_0 decays or grows. We call the matrix (5.3) the *monodromy matrix*.

From Equation (5.1) and Equation (5.2), we also get

$$\frac{d\varphi(t; \mathbf{z})}{dt} = \mathbf{f}(\varphi(t; \mathbf{z}), \lambda), \quad \text{for all } t.$$

Differentiating this equation with respect to \mathbf{z} yields

$$\frac{d}{dt} \frac{\partial \varphi(t; \mathbf{z})}{\partial \mathbf{z}} = \frac{\partial \mathbf{f}(\varphi, \lambda)}{\partial \varphi} \frac{\partial \varphi(t; \mathbf{z})}{\partial \mathbf{z}}.$$

Since $\varphi(0, \mathbf{z}) = \mathbf{z}$, we get

$$\frac{\partial \varphi(0; \mathbf{z})}{\partial \mathbf{z}} = \mathbf{I}.$$

Thus the matrix (5.3) can be re-written as $\Phi(T)$, such $\Phi(t)$ solves the matrix initial-value problem

$$\Phi' = \mathbf{f}_u(\mathbf{u}^*, \lambda)\Phi, \quad \Phi(0) = \mathbf{I}. \tag{5.4}$$

Φ also depends on \mathbf{u}^* and $\Phi = \Phi(t; \mathbf{u}^*)$. We can consider $\mathbf{u}^*(t)$ only and we neglect the argument \mathbf{u}^* for convenience. We call the Φ the *fundamental solution matrix*. It also gives rise to the following definition:

Definition 5.1. The n by n *monodromy matrix* or *circuit matrix* \mathbf{M} of the periodic solution $\mathbf{u}^*(t)$ with period T and initial vector \mathbf{z}^* is defined by

$$\mathbf{M} := \Phi(T) = \frac{\partial \varphi(T; \mathbf{z}^*)}{\partial \mathbf{z}}.$$

φ and Φ are defined in Equation (5.2) and (5.4).

We can check the local stability of \mathbf{u}^* by Floquet theory, based on the monodromy matrix [27, 36, 56, 75, 105]. We also know the monodromy matrix \mathbf{M} has the following two properties.

- Lemma 5.2.** (a) $\Phi(jT) = \mathbf{M}^j$, and
 (b) \mathbf{M} has +1 as eigenvalue with eigenvector $\mathbf{f}(\mathbf{u}(0), \lambda)$ [81].

5.3 The Poincaré map

The Poincaré map is used for describing dynamics of different types of periodic oscillation. The arguments based on the Poincaré map are very helpful for us to find the stability results for periodic orbits.

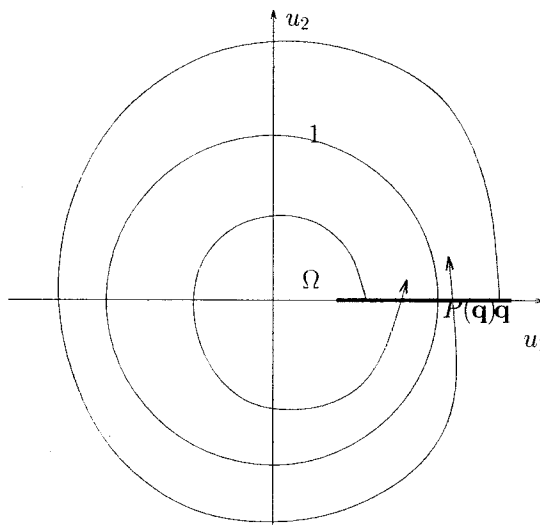


Figure 5.2: The phase plane of Example (5.3), an example Poincaré map [81].

Example 5.3. Consider the ODE system

$$\begin{aligned} u_1' &= u_1 - u_2 - u_1(u_1^2 + u_2^2), \\ u_2' &= u_1 + u_2 - u_2(u_1^2 + u_2^2). \end{aligned}$$

We rewrite the system in polar coordinate as

$$\begin{aligned}\rho' &= \rho(1 - \rho^2), \\ \vartheta' &= 1.\end{aligned}$$

We can see a stable limit cycle when $\rho = 1$, $\vartheta = t$ in Figure (5.2). The neighboring trajectories approach the unit limit cycle. We choose the positive u_1 -axis to measure how the neighboring trajectories vary, namely

$$\Omega = \{(\rho, \vartheta) | \rho > 0, \vartheta = 0\}.$$

An analytical solution is

$$\rho = [1 + (\rho_0^{-2} - 1)e^{-2t}]^{-\frac{1}{2}}, \quad \vartheta = t + \vartheta_0,$$

for initial values ρ_0, ϑ_0 . As a result, a trajectory that starts from $q \in \Omega$ requires time 2π to complete one orbit and passes Ω at the radius

$$P(q) = [1 + (q^{-2} - 1)e^{-4\pi}]^{-\frac{1}{2}}.$$

Function $P(q)$ is an example of a *Poincaré map*. We have a *fixed point* of the Poincaré map where the periodic orbit passes Ω ($P(1) = 1$). Numerically we also get a trajectory at $q = 0.0001$ that gives the following intermediate results

$$\begin{aligned}P(q) &= 0.05347\dots, \\ P^2(q) &= P(P(q)) = 0.99939\dots\end{aligned}$$

We can generalize the concept of a Poincaré map by an n -component differential equation like Equation (5.1). Ω is an $(n - 1)$ -dimensional hypersurface, see Figure (5.3). In addition, all trajectories cross Ω in a neighborhood of $\mathbf{q}^* \in \Omega$ meet two requirements:

- The trajectories intersect Ω transversally, and
- the trajectories cross Ω in the same direction.

Ω is thus specified as a local set by these requirements. If we choose a different \mathbf{q}^* , then there will be another Ω . Ω is also called the *Poincaré section*. We let \mathbf{q}^* be the position where the specific periodic orbit intersects Ω such

$$\mathbf{q}^* = \varphi(T; \mathbf{q}^*).$$

Let $T_\Omega(\mathbf{q})$ be the time for $\varphi(t; \mathbf{q})$ to first return to Ω with $\mathbf{q} \in \Omega$. We have

$$\varphi(T_\Omega(\mathbf{q}); \mathbf{q}) \in \Omega.$$

The *Poincaré map* or *return map* $\mathbf{P}(\mathbf{q})$ is defined by

$$\mathbf{P}(\mathbf{q}) := \mathbf{P}_\Omega(\mathbf{q}) = \varphi(T_\Omega(\mathbf{q}); \mathbf{q}). \quad (5.5)$$

We illustrate $\mathbf{P}(\mathbf{q})$ in Figure (5.3). \mathbf{P} is an $(n-1)$ -dimensional map. Both \mathbf{q} and $\mathbf{P}(\mathbf{q})$ have $(n-1)$ components. It satisfies

$$\mathbf{P}(\mathbf{q}^*) = \mathbf{q}^*.$$

For a fixed point \mathbf{q}^* of \mathbf{P} ,

$$T_\Omega(\mathbf{q}) \rightarrow T \text{ as } \mathbf{q} \rightarrow \mathbf{q}^*.$$

Here, we can study the stability of $\mathbf{u}^*(t)$ by studying the Poincaré map near the fixed point \mathbf{q}^* to see if it is repelling or attracting. Here \mathbf{q}^* can be considered as an $(n-1)$ -dimensional part of $\mathbf{z}^* = \mathbf{u}^*(0)$. Let μ_1, \dots, μ_{n-1} be the eigenvalues of the linearization of \mathbf{P}

$$\frac{\partial \mathbf{P}(\mathbf{q}^*)}{\partial \mathbf{q}}, \quad \text{for } j = 1, \dots, n-1,$$

near the fixed point \mathbf{q}^* . The following rule applies for the stability of fixed points: If the moduli (the absolute value) of all eigenvalues are smaller than 1, then \mathbf{q}^* is stable (attracting); if the modulus of at least one of the eigenvalues is larger than 1, then \mathbf{q}^* is

unstable(repelling) [81].

The dynamic behavior of the sequence $\mathbf{P}^j(\mathbf{q})$ on Ω relies on the eigenvalues and the corresponding eigenvectors in an analogous manner [81]. We observe the possible paths of the sequence

$$\mathbf{q} \rightarrow \mathbf{P}(\mathbf{q}) \rightarrow \mathbf{P}^2(\mathbf{q}) \rightarrow \dots$$

as in Figure (5.4). The sequences of points $\mathbf{P}^j(\mathbf{q})$ of one trajectory are not continuous curves but discrete return points. The curves shown in Figure (5.4) that look continuous are the union of a large number of possible intersection points of distinct orbits; compare Figure (5.3). Depending on the dynamic behavior on Ω , the periodic orbit related to \mathbf{q}^* is called a *saddle cycle* as in Figure (5.4-a), a *spiral cycle* as in Figure (5.4-b), or a *nodal cycle* as in Figure (5.4-c). If we reverse the arrows in (b) and (c), we can obtain the illustrations of the repelling behaviors.

We now apply the stability result for the fixed-point equation $\mathbf{P}(\mathbf{q}) = \mathbf{q}$ to periodic

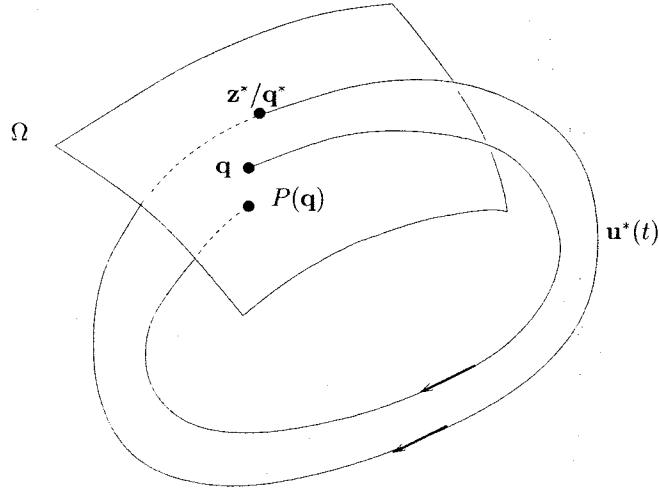


Figure 5.3: Two trajectories intersecting a Poincaré section Ω [81].

solutions of Equation (5.1). To get the related eigenvalues, we calculate the linearization of the Poincaré map of Equation (5.5) around the fixed point \mathbf{q}^* as

$$\frac{\partial \mathbf{P}(\mathbf{q}^*)}{\partial \mathbf{q}} = \frac{\partial \varphi(T; \mathbf{q}^*)}{\partial \mathbf{q}}.$$

Here, we restrict φ to the $(n - 1)$ -dimensional Ω . The n by n monodromy matrix

$$\mathbf{M} = \frac{\partial \varphi(T; \mathbf{z}^*)}{\partial \mathbf{z}}$$

has $+1$ as an eigenvalue. The related eigenvector $(\mathbf{u}^*)'(0)$ is tangent to the intersecting curve $\mathbf{u}^*(t)$. Because $\mathbf{u}^*(t)$ is transversal to Ω , this eigenvector is not in Ω . If we select a basis for the n -dimensional space, we find that the remaining $n - 1$ eigenvalues are those of

$$\frac{\partial \mathbf{P}(\mathbf{q}^*)}{\partial \mathbf{q}}.$$

These eigenvalues are independent of the choice of Ω [71].

We can draw a conclusion about the relation between the eigenvalues of the linearization of the Poincaré map and the eigenvalues of the monodromy matrix \mathbf{M} : \mathbf{M} always has $+1$ as an eigenvalue corresponding to a perturbation along $\mathbf{u}^*(t)$ leading out Ω ; the remaining ones of \mathbf{M} are just the eigenvalues of the linearization of the Poincaré map. These ones determine what happens to small perturbations within Ω . The eigenvalues of \mathbf{M} are called *Floquet multipliers* or *characteristic multipliers*.

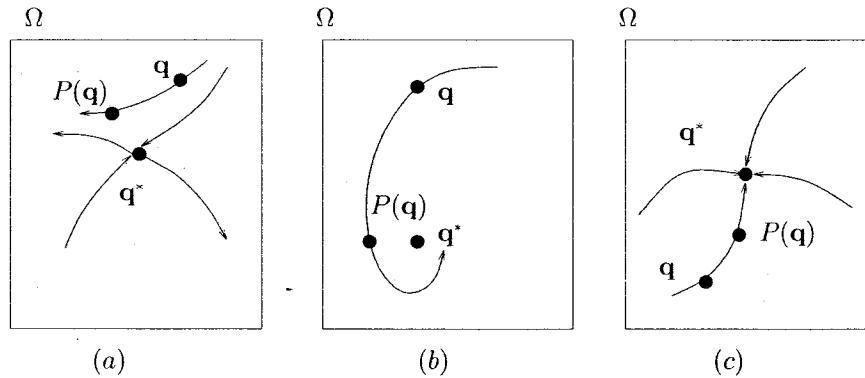


Figure 5.4: Three Poincaré sections with discrete return points filled to curves. \mathbf{q}^* is the return point of a periodic cycle. (a) Saddle cycle, (b) spiral cycle, and (c) nodal cycle [81].

5.4 Mechanism of losing stability for periodic orbits

We have discussed how the local stability of a particular periodic solution is related to the Floquet multipliers. The multipliers and the stability may vary with λ . We give a summary of stability of periodic orbits below. After that, we will discuss mechanisms of losing stability. For further references see [50], [81] and [103].

Summary 5.4. Let $\mathbf{u}(t)$ be a periodic solution to $\mathbf{u}' = \mathbf{f}(\mathbf{u}, \lambda)$ with period T . The monodromy matrix is defined by $\mathbf{M}(\lambda) = \Phi(T)$, where $\Phi(t)$ solves the matrix initial-value problem

$$\dot{\Phi} = \mathbf{f}_{\mathbf{u}}(\mathbf{u}, \lambda)\Phi, \quad \Phi(0) = \mathbf{I}.$$

The matrix $\mathbf{M}(\lambda)$ has n eigenvalues $\mu_1(\lambda), \dots, \mu_n(\lambda)$. Let μ_n be the one equal to $+1$. The other $n - 1$ eigenvalues determine the local stability by the following rule :

$\mathbf{u}(t)$ is stable if $|\mu_j| < 1$ for all $j = 1, \dots, n - 1$;

$\mathbf{u}(t)$ is unstable if $|\mu_j| > 1$ for some j .

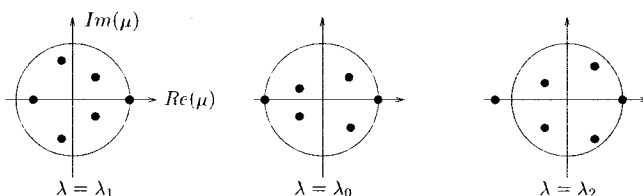


Figure 5.5: Multipliers (eigenvalues of the monodromy matrix) for three values of λ [81].

In Figure (5.5), we show three values of λ and the related multipliers of fictive periodic solutions. The circle is the unit circle. For all λ , there is an eigenvalue $+1$. For $\lambda = \lambda_1$, the figure illustrates a stable solution, because all eigenvalues μ_j lie inside the unit circle for $j = 1, \dots, n - 1$. For $\lambda = \lambda_2$, there is one multiplier outside the unit circle. Thus the periodic orbit is unstable. For some value λ_0 between λ_1 and λ_2 , we see that one multiplier crosses the unit circle. The stability is lost or gained, depending on the approaching direc-

tion to λ_0 . In Figure (5.5), we assume that the critical multiplier crosses the unit circle at -1 . In the following discussion, we see how different types of branching depend on where the critical multiplier or a pair of complex conjugate multipliers cross the unit circle [103].

Generally, there are three ways of crossing the unit circle, thus three associated types of branching. In Figure (5.6), we show the path of the critical multiplier, the value of the eigenvalue with $|\mu(\lambda_0)| = 1$. In Figure (5.6-a), one eigenvalue reaches in addition to $+1$, $\mu(\lambda_0) = 1$. In Figure (5.6-b) the multiplier crosses the unit circle along the negative real axis, $\mu(\lambda_0) = -1$. In Figure (5.6-c) the crossing is with nonzero imaginary part, where a pair of complex conjugate eigenvalues cross the unit circle. All three cases correspond to a loss of stability when λ passes λ_0 . If the arrows are changed to the opposite direction, the figures will illustrate the gaining of stability. Below we list three kinds of mechanisms of losing stability in the remaining part of this section.

- (a) $\mu(\lambda_0) = 1$,
- (b) $\mu(\lambda_0) = -1$,
- (c) $Im(\mu(\lambda_0)) \neq 0$.

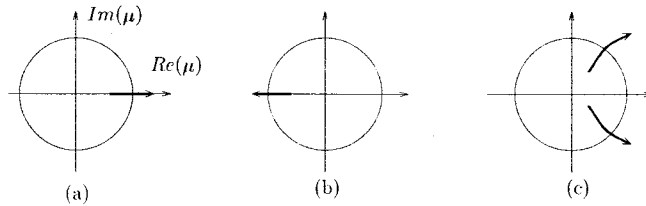


Figure 5.6: Three ways multipliers leave the unit circle [81].

5.4.1 Branch points of periodic solutions

Let \mathbf{P} be the Poincaré map associated with Equation (5.1). We study the fixed points of the Poincaré map,

$$\mathbf{P}(\mathbf{q}, \lambda) = \mathbf{q},$$

for $\mathbf{q} \in \Omega$. Assume situation (a) is met, where $\frac{\partial \mathbf{P}}{\partial \mathbf{q}}$ has unity as an eigenvalue for $\lambda = \lambda_0$, the fixed point equation is

$$\tilde{\mathbf{f}}(\mathbf{q}, \lambda) := \mathbf{P}(\mathbf{q}, \lambda) - \mathbf{q} = \mathbf{0}, \quad (5.6)$$

which is a system of $n - 1$ scalar equations. We can set the Jacobian of $\tilde{\mathbf{f}}$ as

$$\frac{\partial \tilde{\mathbf{f}}}{\partial \mathbf{q}} = \frac{\partial \mathbf{P}}{\partial \mathbf{q}} - \mathbf{I}.$$

Here, $\mu(\lambda_0) = 1$ implies that at λ_0 we get a zero eigenvalue of the Jacobian of $\tilde{\mathbf{f}}$. This is the stationary bifurcation scenario. If the eigenvalue is real, that is, we have simple bifurcation points and folds of the Poincaré map. In Figure (5.7), we illustrate the multiplicity of fixed point of \mathbf{P} on Ω . For λ_1 , we have one fixed point only. For λ_2 , we have three fixed points. At λ_0 , there is a bifurcation of the periodic orbit; see Figure (5.8).

The situation of a fold is illustrated in Figure (5.9). If we vary λ approaching λ_0 from λ_2 , the two fixed points eventually collapse. At $\lambda = \lambda_0$, the resulting periodic orbit is *semi-stable*. If λ passes λ_0 , this periodic orbit will vanish. This scenario is shown in Figure (5.10). It shows the three two-dimensional phase planes for λ_1 , λ_0 and λ_2 . At λ_2 , the periodic orbit disappears. If we vary λ from λ_2 to λ_1 , it gives birth of a limit cycle at $\lambda = \lambda_0$. Further, it will split into two periodic orbits.

For $\mu(\lambda_0) = 1$, we give a summary: typically, folds occur. These folds are accompanied

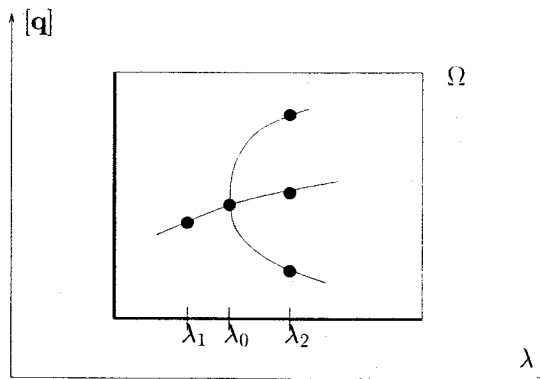


Figure 5.7: Illustration of a pitchfork bifurcation in a Poincaré section [81].

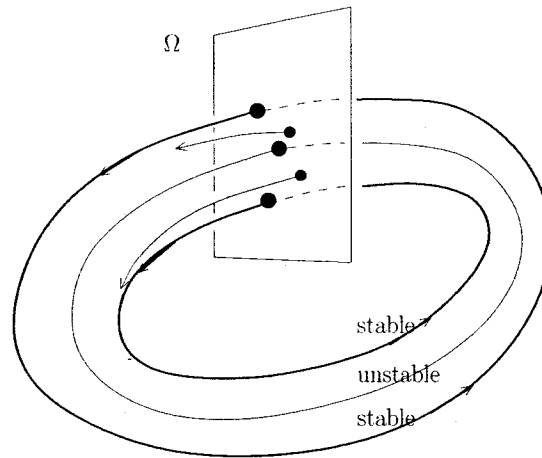


Figure 5.8: Close to a pitchfork bifurcation; corresponding to λ_2 in Figure (5.7) [81].

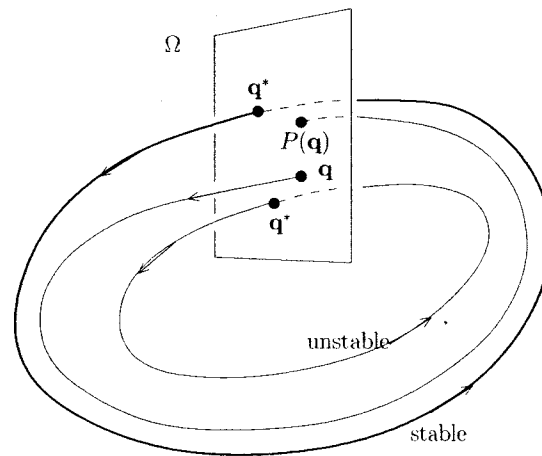


Figure 5.9: Close to a fold [81].

by the birth or death of limit cycles. Sometimes, if f satisfies symmetry or other regularity properties, pitchfork or transcritical bifurcation may happen.

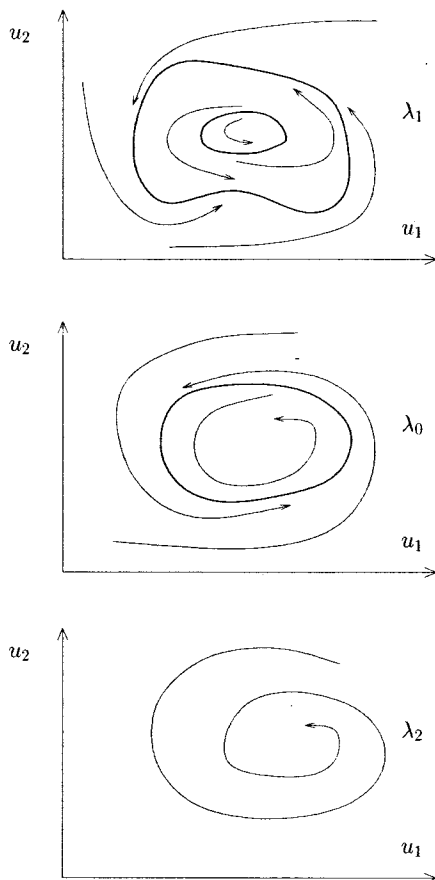


Figure 5.10: Phase planes illustrating how a stable orbit collapses at a fold for λ_0 [81].

5.4.2 Period doubling

We now discuss the second case, the loss of stability with $\mu(\lambda_0) = -1$. From Equation (5.6) we expect that the Jacobian of

$$\tilde{\mathbf{f}}(\mathbf{q}, \lambda_0) = \mathbf{P}(\mathbf{q}, \lambda_0) - \mathbf{q}$$

is nonsingular. As a result, at λ_0 there is no branch point as that happens under condition (a). Furthermore, what we see is a smooth branch $\mathbf{q}(\lambda)$ passes through $\mathbf{q}(\lambda_0)$ without bifurcating itself. The logistic map is a typical example [81]. There is a swap of stability of period-one fixed points to period-two fixed points at λ_0 . We call this phenomenon *period doubling*.

We give a short discussion of periodic oscillations, when for $\lambda = \lambda_0$ the monodromy

matrix has an eigenvalue $\mu = -1$. We apply chain rule to the Poincaré map and get

$$\frac{\partial}{\partial \mathbf{q}} [\mathbf{P}(\mathbf{P}(\mathbf{q}))] = \left(\frac{\partial \mathbf{P}(\mathbf{q})}{\partial \mathbf{q}} \right)^2.$$

We can see the fixed point \mathbf{q} of $\mathbf{P}(\mathbf{q}) = \mathbf{q}$ leads to the eigenvalue $\mu = +1$ for $\mathbf{P}^2(\mathbf{q}) =$

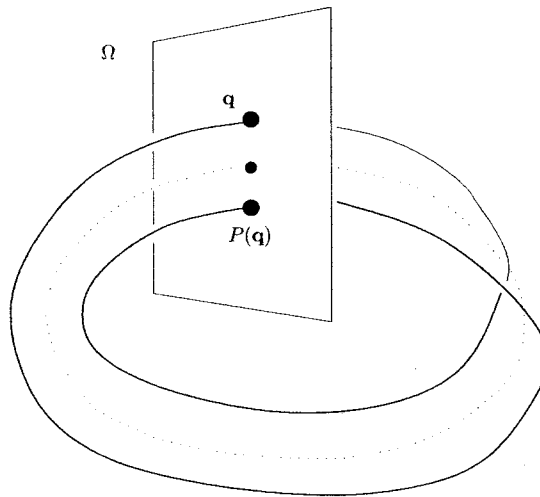


Figure 5.11: Close to period doubling [81].

$\mathbf{P}(\mathbf{P}(\mathbf{q}))$. We illustrate this bifurcation of the double period in Figure (5.11). The dashed curve represents an unstable simple period oscillation; the heavy curve is a stable double period orbit. We call a band which is spanned by the trajectories a Möbius strip; see Figure (5.11). For λ approaching λ_0 , the double-loop orbit reduces to the single-loop orbit. For λ leaving λ_0 , we observe that at λ_0 the stable single-period oscillation splits into stable double-period oscillations. At least a three-dimensional phase space of the autonomous system (5.1) is required for this splitting, namely $n \geq 3$. The term “period” has different meanings for the Poincaré map \mathbf{P} and the periodic oscillation $\mathbf{u}(t)$. For a map, it is the number of iterations for \mathbf{P} to reproduce the fixed point. When λ is varied, this integer remains constant given no further period doubling occurs. However, that of the periodic oscillation may vary with λ . The single-period oscillations and the double-period oscillations have different responses to this variation. It is known that the periods do not differ exactly by a factor of 2 [22].

Period doubling is also called *flip bifurcation* or *subharmonic bifurcation*. We observe

this phenomenon in many applications, for example, in chemical reactions [13, 18, 26], nerve models [64], and Navier-Stokes equations [101]. It also occurs in the Brusselator [70] and the Lorenz equations [21].

Although we do not focus on the computing of period doubling solutions, such dynamical behavior is crucial to understand systems like the Lorenz equations, the CR3BP, *etc.*

5.4.3 Bifurcation to a torus

For condition (c), for which a pair of complex-conjugate multipliers crossing the unit circle cause the loss of stability, where

$$\mu(\lambda_0) = e^{\pm i\vartheta}, \text{ for } \vartheta \neq 0 \text{ and } \vartheta \neq \pi.$$

i is the imaginary unit and the angle ϑ is an irrational multiple of 2π . Then we see the Poincaré map has an *invariant curve* C in Ω ; see Figure (5.12). If we mark an arbitrary point \mathbf{q} on C , then all iterates of the Poincaré map stay on that “drift ring.” Consequently, trajectories spiral around a torus-like object. There is a cross section between the hypersurface Ω and the torus as the invariant curve C . The central axis of the torus is the unstable periodic orbit. At λ_0 , we see a bifurcation that a periodic orbit bifurcates into a torus. A bifurcation into a torus can take place only for $n \geq 3$, since such bifurcation requires complex multipliers in addition to the eigenvalue that is unity. Tori can also be attractive or repelling

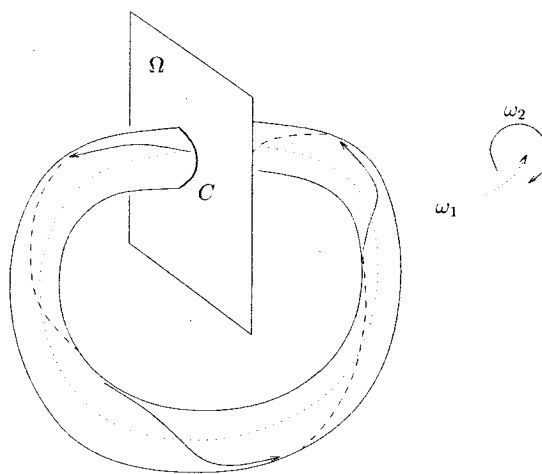


Figure 5.12: A “Torus” trajectory encircles an unstable periodic orbit [81].

like periodic orbits. Trajectories approach an attractive torus from both the inside domain and the outside domain, for $n = 3$. The projection of trajectories to the hypersurface Ω resemble the dynamics close to a Hopf bifurcation, where a stable limit cycle encircles an unstable equilibrium. We also call the bifurcation into a torus a *Hopf bifurcation of periodic orbits*, a *secondary Hopf bifurcation*, or a *generalized Hopf bifurcation*. Due to a theoretical result of Naimark and Sacker [62, 80], bifurcation into tori is also called *Naimark-Sacker bifurcation*. For a repelling torus, we can visualize it as a tube surrounding a stable periodic orbit ($n = 3$). The domain of attraction of the limit cycle vanishes and the torus dies as λ tending to λ_0 . We call this case a subcritical torus bifurcation. If ϑ is an irrational multiple of 2π , a trajectory that starts on the invariant curve C will never return to the original position. There are two frequencies $\omega_1(\lambda)$, $\omega_2(\lambda)$ after the bifurcation from periodic orbit with frequency $\omega(\lambda)$ to a torus. One frequency shows longitudinal motion along the axis within the torus, the other frequency shows latitudinal motion along the cross section; see the inset in Figure (5.12)). If the ratio ω_1/ω_2 is irrational, the flow is called quasi-periodic. ω_1 and ω_2 are *incommensurate* if the equation

$$p\omega_1 + q\omega_2 = 0$$

has no nonzero integer solutions for p and q . If for some λ , ω_1/ω_2 is rational, we call it a *locked state*. The trajectory on the torus is closed and becomes periodic. In [60] this phenomenon is explained as a global bifurcation caused by saddle node entering the drift ring. For quasi-periodic flow on tori, refer to [56, 61]. For higher dimensional tori, “m-torus” for m-dimensional torus, see [81]. In chapter 6, we show some tori computed by AUTO for the CR3BP.

Chapter 6

The Circular Restricted Three-Body Problem

We will study the CR3BP in this chapter. We first describe the history of the circular restricted Three-Body problem. For a detailed bifurcation and stability analysis see [29, 102]. Our focus is the computation of the 2D unstable manifold of periodic orbits and the detection of heteroclinic and homoclinic connections. As in Chapter 4, we also try to visualize the computational results with the latest visualization tools. The computational work in this chapter is an extension of the AUTO demo ‘r3b’.

6.1 Introduction

In this section we give a brief introduction to the history of the studies leading to the CR3BP. For more details see, for example, [19]. To study the CR3BP, we must start from the famous N-Body Problem. The CR3BP is a special case of the N-Body Problem. The classical *N-Body Problem* assumes that there are n objects that attract one another by their mutual gravitational forces only. If the initial positions and velocities of the objects are known, we want to determine their positions and velocities at any time in the future. For centuries, the N-Body Problem has been actively studied. Newton first formulated this problem precisely. Other famous scientists, such as Copernicus, Kepler, Lagrange and Poincaré also contributed. However, there remain many unresolved issues. The simplest

case of the N-Body Problem is the *One-Body Problem*. In the One-Body Problem, there is only one body with initial position and velocity. When an external force is applied to it, the motion of the body changes. Next, consider two isolated objects that interact with each other through their gravitational forces, for example the Sun and the Earth. This problem can be studied analytically. However, it requires advanced knowledge of calculus and analytic geometry. Newton solved this problem by means of *Newton's law of gravitation*. If we add a third body such as the Moon to the problem, we obtain a Three-Body system. The third body makes the motion of the bodies very complicated. It is quite difficult to predict the positions of the bodies later on, and many questions remain unanswered.

Facing the complexity of the Three-Body Problem, mathematicians began to look for some simplifications of the problem. People considered special cases, for example, the case where one particle is much less massive than the other two, or the case where two particles are much less massive than the third. Pierre-Simon de Laplace gave results for a special case. However, he failed at the general case. Henri-Poincaré also studied this problem, but his attempt failed too. Weierstrass tried a different approach. He believed that the problem has no closed-form solution. He proposed to approximate the solution by finding a convergent series. His approach led to many breakthroughs. Heinrich Bruns confirmed Weierstrass' idea in 1892. The Finnish mathematician Karl Sundham obtained the first convergent series for the Three-Body Problem in 1913, although for practical purposes the convergence is so slow for it to be useless in practice. In 1941, Siegel introduced a series solutions for a triple collision. In the 1960s, Stephen Smale of the University of California, Berkeley, proposed a new method for the Three-Body Problem. He believed that the dynamical system can be studied in terms of topological transformations. In 1991, Qiu Dong Wang also got a convergent series. In his method, Wang resolved the problem of collisions. He introduced a measure that makes time run faster as two or more bodies approach each other. Thus, if the system approaches a collision, time approaches infinity. However, the convergence is still too slow to be practical.

Chenciner and Montgomery [2] discovered a strange periodic solution of the Three-Body Problem. In this solution, the three bodies have equal masses and they move along the same planar figure-8 curve. The three bodies share the same period, but there are phase delays

of one-third the period. The system has a zero total angular momentum. Except for the well-known triangular circular orbits, the figure-8 orbit is the first periodic solution that has the remarkable triple overlap property.

6.2 The circular restricted Three-Body Problem

To formalize the CR3BP, we have to start from *Newton's universal force law* or *Newton's gravitational law*. It states that

$$F_g = G \frac{m_1 \cdot m_2}{r^2} \quad (6.1)$$

where,

F_g is the gravitational force in units of N (Newton),

G is the universal gravitational constant, $G = 6.672 \times 10^{-11} N \cdot m^2/kg^2$,

m_1, m_2 are the masses of the two bodies in units of kg ,

r is the distance between the two bodies, in units of m (meter).

If we take into account the direction of the force, then the above equation becomes

$$\mathbf{F}_g = G \frac{m_1 \cdot m_2}{\|\mathbf{r}\|^3} \mathbf{r}, \quad (6.2)$$

where \mathbf{r} denotes the vector from m_1 to m_2 and $\|\mathbf{r}\|$ represent the distance between the two bodies. In the general Three-Body Problem, each of the three bodies i has mass m_i , and its position in space is denoted by \mathbf{r}_i , for $i = 1, 2, 3$. In this system, the motion of a body obeys the classical *Newtonian law of inertia*, namely, *acceleration = force/mass*. According to this law, we obtain

$$\begin{cases} d^2\mathbf{r}_1/dt^2 = (\mathbf{F}_{12} + \mathbf{F}_{13})/m_1, \\ d^2\mathbf{r}_2/dt^2 = (\mathbf{F}_{21} + \mathbf{F}_{23})/m_2, \\ d^2\mathbf{r}_3/dt^2 = (\mathbf{F}_{31} + \mathbf{F}_{32})/m_3. \end{cases} \quad (6.3)$$

In the above equations,

m_i is the mass of the object i ,

\mathbf{r}_i is the position vector of the object m_i in space,

\mathbf{F}_{ij} is the gravitational force of attraction of object m_i toward object m_j ,

t is the time.

By geometric principles, we let \mathbf{r}_{ij} be $\mathbf{r}_j - \mathbf{r}_i$. By Newton's universal force law, if we do not consider the direction of the forces, we have,

$$\|\mathbf{F}_{ij}\| = G \frac{m_i \cdot m_j}{\|\mathbf{r}_{ij}\|^2}. \quad (6.4)$$

We know that the direction of \mathbf{F}_{ij} is the same as that of the unit vector $\mathbf{r}_{ij}/\|\mathbf{r}_{ij}\|$. Therefore, the vector form of Equation (6.4) is

$$\mathbf{F}_{ij} = G \frac{m_i \cdot m_j \cdot \mathbf{r}_{ij}}{\|\mathbf{r}_{ij}\|^3}. \quad (6.5)$$

Substituting Equation (6.5) into Equation (6.3), the classical form of motion of the Three-Body Problem is

$$\frac{d^2 \mathbf{r}_i}{dt^2} = G \left(\frac{m_j \cdot \mathbf{r}_{ij}}{\|\mathbf{r}_{ij}\|^3} + \frac{m_k \cdot \mathbf{r}_{ik}}{\|\mathbf{r}_{ik}\|^3} \right); \quad i, j, k = 1, 2, 3; \quad i \neq j, i \neq k, j \neq k; \quad (6.6)$$

for further details see [19, 48]. In the *restricted Three-Body Problem* (R3BP), the mass of the third body is supposed to be infinitesimally small. It will not influence the motion of the other two large bodies, which are called the *primaries*. Many possible forms of the R3BP are discussed in [104]. The most general one is the *circular restricted Three-Body Problem* (CR3BP). In the CR3BP, the two primaries move in planar circular orbits around their common center of mass (barycenter). The movement of the third body is determined by

$$\frac{d^2 \mathbf{r}_3}{dt^2} = G \left(\frac{m_1 \cdot \mathbf{r}_{31}}{\|\mathbf{r}_{31}\|^3} + \frac{m_2 \cdot \mathbf{r}_{32}}{\|\mathbf{r}_{32}\|^3} \right). \quad (6.7)$$

Since the two primaries are moving in circular planar orbits, we can put the system in a rotating coordinate frame. In this frame, the two primaries are fixed. The origin of the frame is at the barycenter of the two primaries. The x -axis points from the large primary to the small primary. The z -axis is orthogonal to the orbital plane, and the y -axis completes the right-handed orthogonal coordinate system. The x and y axes rotate with constant angular velocity, ω . The two primaries are fixed at $(x_1, 0, 0)$ and $(x_2, 0, 0)$, where x_1 is

negative. The third body, called the *infinitesimal*, is located at the position denoted by (x, y, z) .

If we let x' be the first order derivative of x and x'' the second order derivative of x , then from Equation (6.7), we get

$$\begin{aligned} x'' &= G \left(\frac{m_2(x_1 - x)}{\|\mathbf{r}_{31}\|^3} + \frac{m_1(x_2 - x)}{\|\mathbf{r}_{32}\|^3} \right) + 2\omega y' + \omega^2 x, \\ y'' &= -G y \left(\frac{m_1}{\|\mathbf{r}_{31}\|^3} + \frac{m_2}{\|\mathbf{r}_{32}\|^3} \right) + 2\omega x' + \omega^2 y, \\ z'' &= -G z \left(\frac{m_1}{\|\mathbf{r}_{31}\|^3} + \frac{m_2}{\|\mathbf{r}_{32}\|^3} \right). \end{aligned} \quad (6.8)$$

Here, ω is the rotation speed of the frame, determined by

$$\omega^2 \|\mathbf{r}_{12}\|^3 = G(m_1 + m_2), \quad (6.9)$$

and

$$\begin{aligned} \|\mathbf{r}_{12}\| &= x_2 - x_1, \\ x_1 &= -\frac{m_1 \cdot \|\mathbf{r}_{12}\|}{m_1 + m_2}, \\ x_2 &= \frac{m_2 \cdot \|\mathbf{r}_{12}\|}{m_1 + m_2}. \end{aligned} \quad (6.10)$$

A parameter μ is introduced to simplify the system further. It denotes the ratio of the small primary's mass to the total mass of the system. For example, for the Earth-Moon system, $\mu = 0.01215$; for the Sun-Jupiter system, $\mu = 9.53 \times 10^{-4}$; and for the Sun-Earth system, $\mu = 3.0 \times 10^{-6}$. Proper units are chosen so that the distance between the primaries, the sum of the masses of the primaries, the angular velocity ω of the primaries and the gravitational constant are all equal to one. The small primary is located at $(\mu, 0, 0)$, and the large primary at $(1 - \mu, 0, 0)$. The orbital period T is 2π . Equation (6.8) can then be

written in the following form [4, 58]

$$\begin{aligned}
x'' &= 2y' + x - \frac{(1-\mu)(x+\mu)}{r_1^3} - \frac{\mu(x-1+\mu)}{r_2^3}, \\
y'' &= -2x' + y - \frac{(1-\mu)y}{r_1^3} - \frac{\mu y}{r_2^3}, \\
z'' &= -\frac{(1-\mu)z}{r_1^3} - \frac{\mu z}{r_2^3},
\end{aligned} \tag{6.11}$$

where

$$\begin{aligned}
r_1 &= \|\mathbf{r}_{31}\| = \sqrt{(x+\mu)^2 + y^2 + z^2}, \\
r_2 &= \|\mathbf{r}_{32}\| = \sqrt{(x-1+\mu)^2 + y^2 + z^2}.
\end{aligned} \tag{6.12}$$

We also have the *Jacobi constant* C which is equal to $-2E$. For the above equations, there is an integral of motion, which is a function of the coordinates and the velocities, and which is constant along a trajectory in phase space. This constant is called the energy [4, 58],

$$E = \frac{x'^2 + y'^2 + z'^2}{2} - U(x, y, z) - \frac{\mu(1-\mu)}{2}, \tag{6.13}$$

where

$$U = \frac{1-\mu}{r_1} + \frac{\mu}{r_2} + \frac{x^2 + y^2}{2}. \tag{6.14}$$

The Jacobi integral is the only integral of the Restricted Circular Three-Body Problem [58]. See the thesis of C. Zhang [23] about the history and derivation of the CR3BP; see also thesis of V. Romanov [102] for work related to this section.

6.3 Computing periodic orbits with AUTO

We introduced pseudo-arclength continuation in Chapter 2. In AUTO, we can use it to compute a family of periodic solutions of a system. In Equation (5.1), there are n component functions for the nonlinear function \mathbf{f} , e.g., $f_1(\mathbf{u}, \lambda)$, ..., $f_n(\mathbf{u}, \lambda)$. We formulate the orbit

continuation as the following constrained periodic boundary value problem [23, 30]:

$$\begin{aligned}
(a_1) \quad & \mathbf{u}'_1(t) = T_1 \mathbf{f}(\mathbf{u}_1(t), \lambda_1), \\
(a_2) \quad & \mathbf{u}_1(0) = \mathbf{u}_1(1), \\
(a_3) \quad & \int_0^1 \langle \mathbf{u}_1(\tau), \mathbf{u}'_0(\tau) \rangle d\tau = 0, \\
(b) \quad & \int_0^1 \langle \mathbf{u}_1(\tau) - \mathbf{u}_0(\tau), \dot{\mathbf{u}}_0(\tau) \rangle d\tau + (T_1 - T_0)\dot{T}_0 + (\lambda_1 - \lambda_0)\dot{\lambda}_0 = \Delta s.
\end{aligned} \tag{6.15}$$

This equation is to be solved for $\mathbf{X}_1 = (\mathbf{u}_1(\cdot), T_1, \lambda_1)$, given $\mathbf{X}_0 = (\mathbf{u}_0(\cdot), T_0, \lambda_0)$ and the path tangent $\dot{\mathbf{X}}_0 = (\dot{\mathbf{u}}_0, \dot{T}_0, \dot{\lambda}_0)$. T_1 is the unknown period. Equation (6.15- a_2) imposes unit periodicity, after we rescale the independent variable t . Equation (6.15- a_3) is a phase condition. The condition fixes the phase of the new orbit $\mathbf{u}_1(\cdot)$ relative to the given orbit $\mathbf{u}_0(\cdot)$. We can also replace it by the classical Poincaré phase condition

$$(a'_3) \quad \langle \mathbf{u}_1(0) - \mathbf{u}_0(0), \mathbf{u}'_0(0) \rangle = 0.$$

However, the integral phase condition Equation (6.15- a_3) has the desirable property of minimizing phase drift relative to $\mathbf{u}_0(\cdot)$. This often allows much bigger continuation steps to be taken [31]. Equation (6.15- b) provides the functional form of the pseudo-arclength constraint of the orbit continuation; see Equation (2.15). Orbit continuation enables us to get $\mathbf{u}_1, \mathbf{u}_2 \dots$ after we compute the first orbit \mathbf{u}_0 . As we have discussed in the previous chapters, AUTO uses piecewise polynomial collocation with Gaussian-Legendre collocation points (“*orthogonal collocation*”) with adaptive mesh selection [14, 78, 100]. In this respect it is similar to COLSYS [97] and COLDAE [99]. Combining this method with continuation, we can get the numerical solution of “difficult” problems. AUTO also computes the *characteristic multipliers* (or *Floquet Multipliers*), that determine asymptotic stability and bifurcation properties, as a by-product of the decomposition of the Jacobian of the boundary value collocation system [31, 66, 90]. The stability analysis using Floquet multipliers is discussed in Chapter 5. For more details on the boundary value approach for computing periodic solutions see [31]; further references include [3, 35].

6.4 The CR3BP implementation in AUTO

As mentioned earlier, in the CR3BP system, there is one conserved quantity, namely, the Jacobi constant C ; see Equation (6.13). To use AUTO to continue periodic solutions of the CR3BP, we need to modify the original form of the equations. First, the three-dimensional second-order equations are re-written as a standard six-dimensional system in first order form with scaled time. Thus the period T appears explicitly in the equations. Periodic boundary conditions and the integral phase constraint are also added, as in Equation (6.15). Second, in order to use boundary value algorithms, we introduce an *unfolding term* with corresponding *unfolding parameter*. A suitable choice for the unfolding term is $\lambda \nabla E$, where λ is the unfolding parameter. However, this choice is not unique. For the CR3BP it is more convenient if we introduce a simpler unfolding term, namely, one that corresponds to “damping”. After we introduce the unfolding parameter, the first order system then becomes:

$$\begin{aligned}
 x' &= T v_x, \\
 y' &= T v_y, \\
 z' &= T v_z, \\
 v'_x &= T \left(2v_y + x - \frac{(1-\mu)(x+\mu)}{r_1^3} - \frac{\mu(x-1+\mu)}{r_2^3} \right) + \lambda v_x, \\
 v'_y &= T \left(-2v_x + y - \frac{(1-\mu)y}{r_1^3} - \frac{\mu y}{r_2^3} \right) + \lambda v_y, \\
 v'_z &= T \left(-\frac{(1-\mu)z}{r_1^3} - \frac{\mu z}{r_2^3} \right) + \lambda v_z.
 \end{aligned} \tag{6.16}$$

It is important to stress that, while λ is a scalar unknown that is solved for at each continuation step, its value will be zero (up to numerical precision) upon convergence of Newton’s method. There is a discussion for this simple yet crucial observation in [33]. It is also theoretically justified in [39].

6.5 Some facts about the CR3BP

There are five equilibrium points [19, 29, 58], also called the *libration points* or *Lagrange points*, L_1 , L_2 , L_3 , L_4 and L_5 . All five lie in the same plane as the primaries. The points L_1 , L_2 and L_3 are on the same line as the two primaries. The points L_4 and L_5 each form an equilateral triangle with the primaries.

The points L_1 , L_2 and L_3 can be computed by solving x from the following equation, where $z = 0$, $y = 0$, and x_1 and x_2 are acquired from Equation (6.10), choosing $\|\mathbf{r}_{12}\| = 1$:

$$0 = x - \frac{(1 - \mu)(x - x_1)}{|x - x_1|^3} - \frac{\mu(x - x_2)}{|x - x_2|^3}. \quad (6.17)$$

For the libration points L_4 and L_5 , the y-coordinate is non-zero. Their positions [72] are

$$(x, y, z) = \left(\frac{(1 - 2\mu)}{2}, \pm \frac{\sqrt{3}}{2}, 0 \right). \quad (6.18)$$

In Table (6.1), we list some notation used for the CR3BP. For further details see [29]. There are also symmetries in the CR3BP system: first, if (x, y, z) is a solution, then so is $(x, y, -z)$. Second, if (x, y, z) is a solution then so is $(x, -y, z)$.

Symbol	Definition
L_i	libration Point, ($i = 1, \dots, 5$)
\mathbf{A}_i	The Axial family from \mathbf{L}_i , ($i = 1, 2, 3$)
\mathbf{B}_i	The Backflip family from \mathbf{V}_i , ($i = 1, 2, 3$)
\mathbf{C}_i	The planar ‘‘Circular’’ family, ($i = 1, 2$)
\mathbf{H}_i	The Halo family that bifurcates from \mathbf{L}_i , ($i = 1, 2, 3$)
\mathbf{K}_i	The \mathbf{K} -families from \mathbf{L}_i , ($i = 1, \dots, 4$)
\mathbf{L}_i	The planar Lyapunov family from L_i , ($i = 1, 2, 3$)
\mathbf{L}_i	The Long-Period planar family from L_i , ($i = 4, 5$)
\mathbf{S}_i	The planar families that bifurcate from L_i , ($i = 1, 2, 3$)
\mathbf{V}_i	The Vertical family from L_i , ($i=1, \dots, 5$)

Table 6.1: Some abbreviations used in the CR3BP

6.6 Computing unstable manifolds and heteroclinic orbits of the CR3BP with AUTO

6.6.1 The general procedure

- (a) Compute the starting data. Normally we compute the equilibria at this stage of different μ values. Then from one equilibrium, we can compute a family of periodic orbits. Sometimes, another family of periodic orbits can be reached from a computed family. For example, the \mathbf{H}_1 family bifurcates from the \mathbf{L}_1 family.
- (b) AUTO computes the Floquet multipliers of the periodic orbits in step (a). We can use the AUTO command '@ff' to list their values. The orbits that we are interested in will have exactly one multiplier for which the absolute value is greater than 1, thus these orbits are unstable.
- (c) Convert the data for a selected solution from the family chosen in (b) and add a zero adjoint variable. Save the converted solution.
- (d) Compute the Floquet eigenfunction. At the end of this step, the norm of the eigenfunction should be nonzero. Save the result.
- (e) We extract data for the selected solution from the results of steps (c) and step (d). We save the extracted data containing the mass-ratio parameter μ , the energy E , the initial step size ϵ into the direction of the unstable manifold, the orbit coordinates at "time zero", and the Floquet eigenfunction coordinates at "time zero".
- (f) In step (e) we have the starting point in the unstable manifold. Now we do a time integration using continuation in the orbit segment time T . We save the trajectory as the starting orbit in the unstable manifold.

- (g) After we successfully compute a starting orbit in the unstable manifold, we also adapt the step size into the unstable manifold, namely, by varying the parameter ϵ . This variation resembles the different propelling efforts for a satellite's rocket. However, the parameter ϵ should not be too large, because then the manifold may no longer be accurate.
- (h) If we extend the computation in step (f) by continuing for higher value of T , it may be possible to locate a heteroclinic orbit or a homoclinic orbit.

For details see AUTO demo 'r3b'. In Appendix (B), we give a Python script that generalizes the computational set-up and provides more supporting functions.

6.6.2 Results

The CR3BP has a wealth of solutions. In our numerical results, we reveal some aspects of the nature of the solutions. Our results are given in graphical presentations, together with explanations.

For convenience of computation, μ is set equal to 0.063 (not the real value for the Earth-Moon system). Other μ values are left for future study.

In the following figures, the large primary is shown as the Earth and the small primary is shown as the Moon. The five grey cubes represent the equilibrium points L_1 to L_5 .

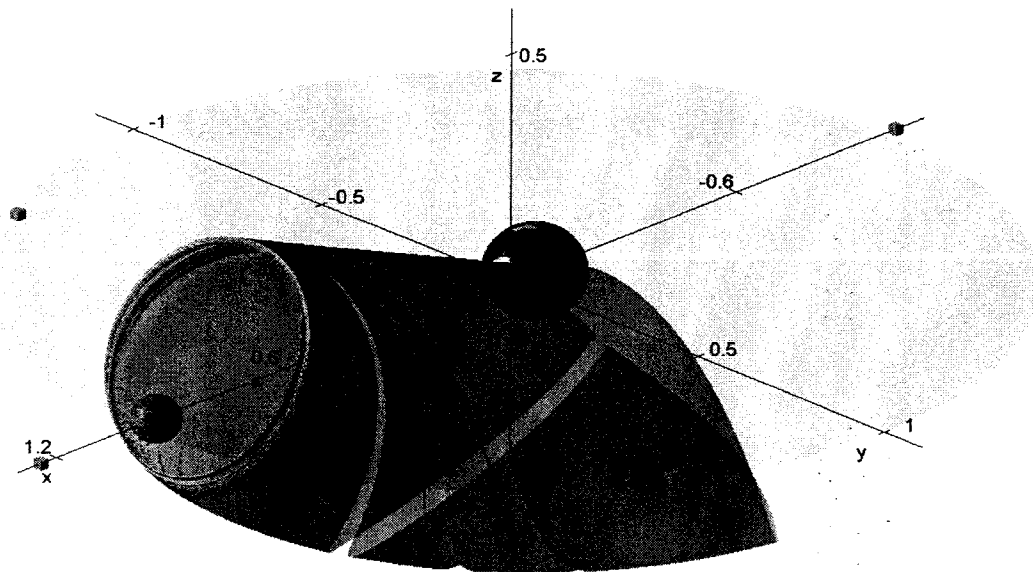


Figure 6.1: An unstable manifold of a periodic orbit in the family \mathbf{H}_1 at energy -1.465584 .

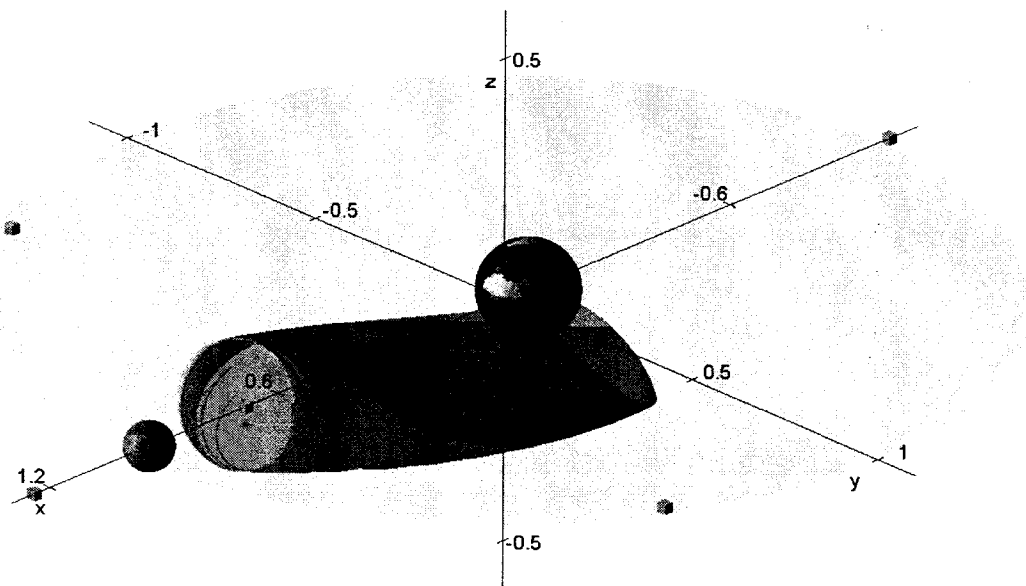


Figure 6.2: An unstable manifold of a periodic orbit in the family \mathbf{H}_1 at energy -1.673174 .

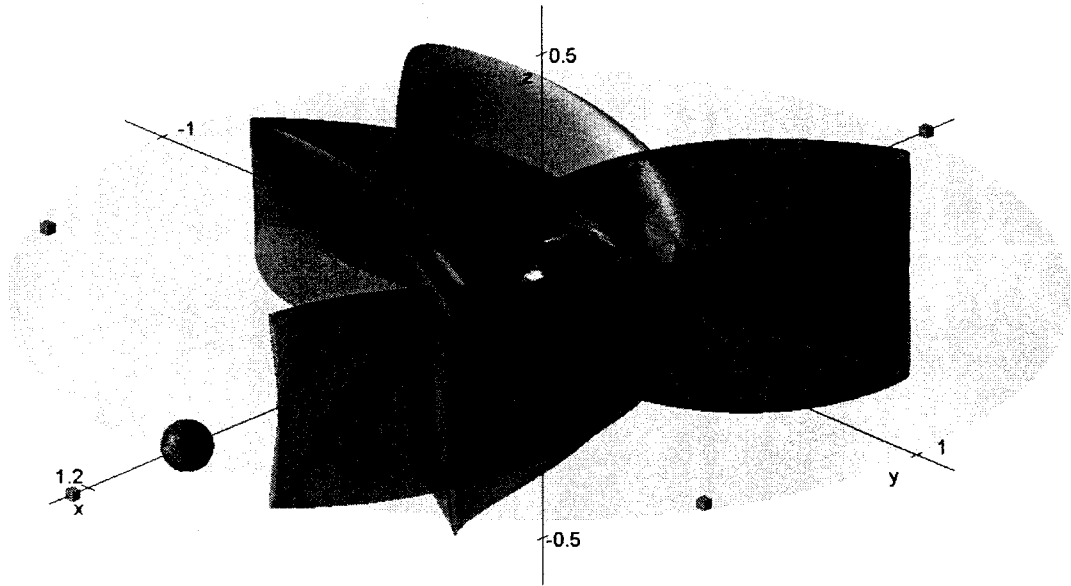


Figure 6.3: An unstable manifold of a periodic orbit in the family V_1 at energy -1.660129 .

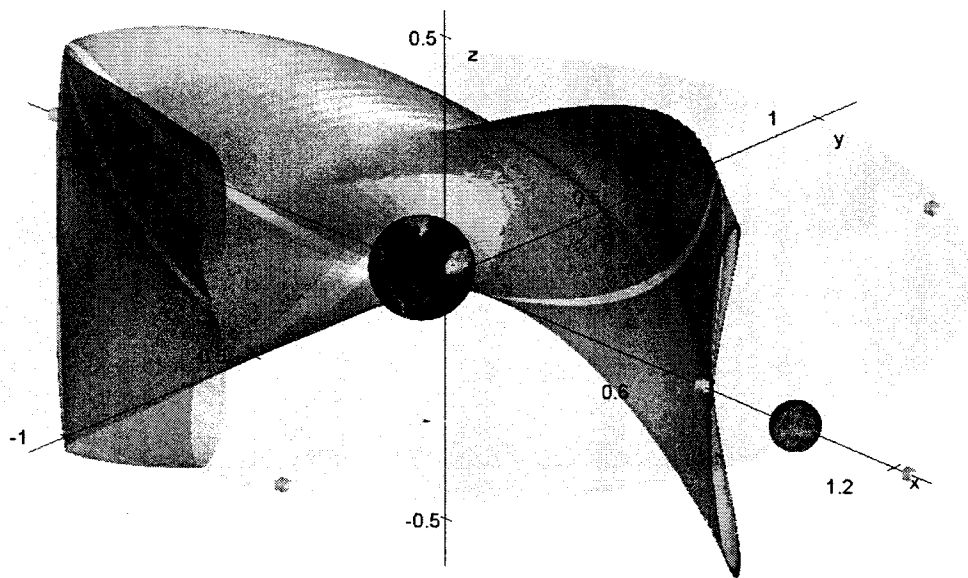


Figure 6.4: An unstable manifold of a periodic orbit in the family V_1 at energy -1.507751 .

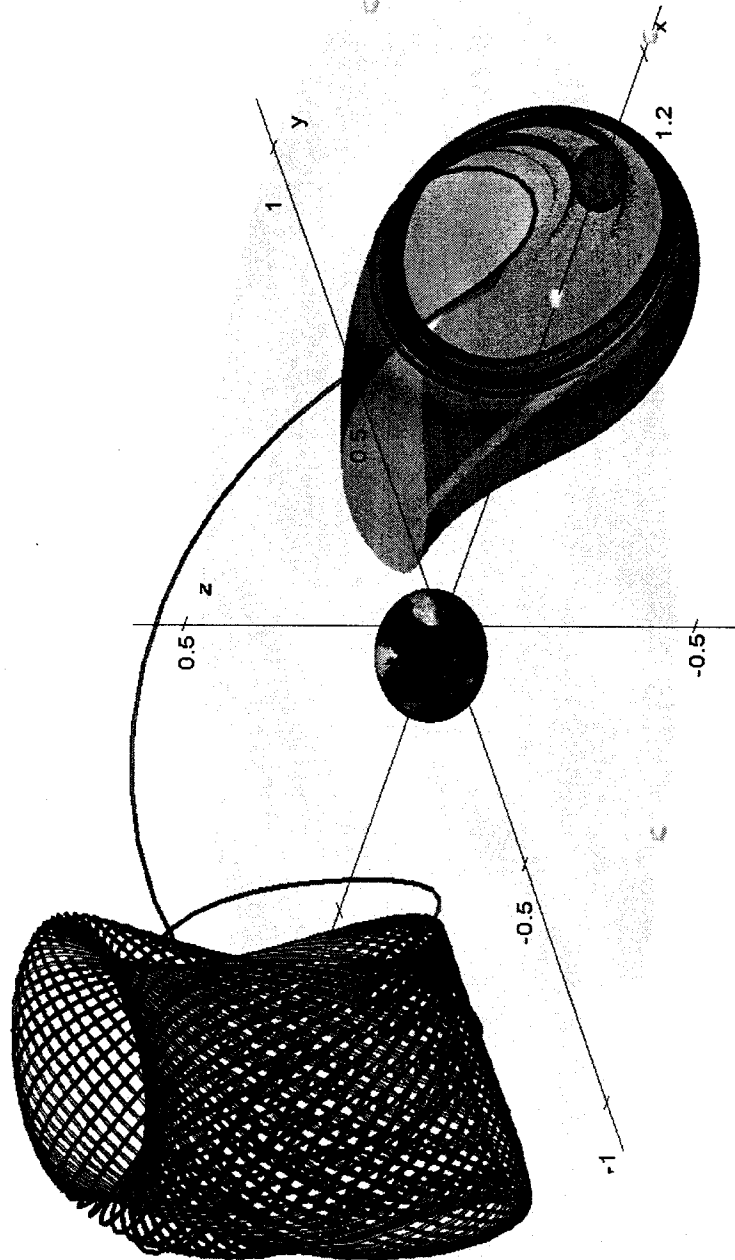


Figure 6.5: A heteroclinic connection from a periodic orbit in family \mathbf{H}_1 to a torus at equilibrium L_3 at energy -1.465585 .

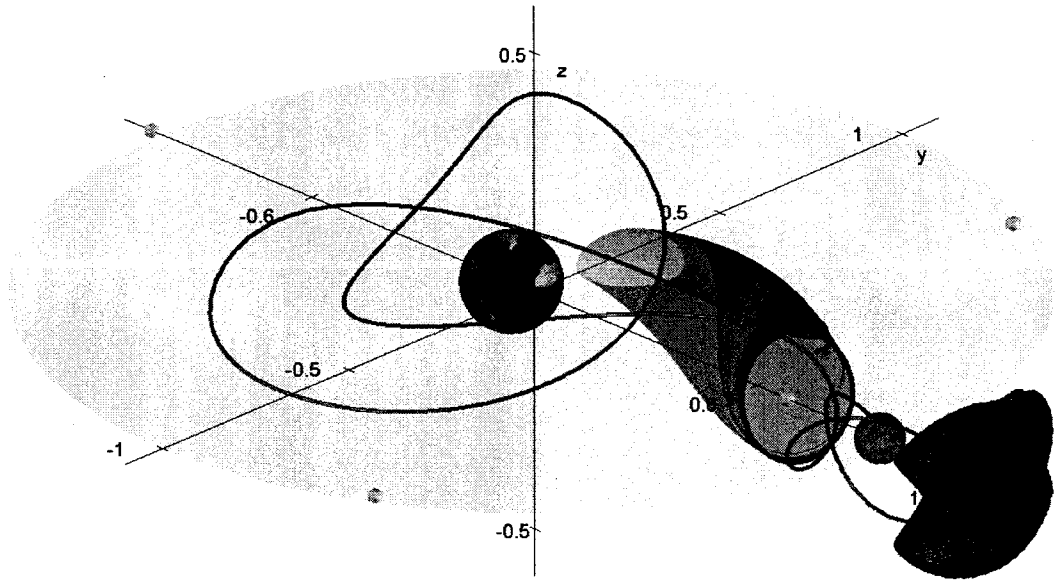


Figure 6.6: A heteroclinic connection from a periodic orbit in family \mathbf{H}_1 to a torus at equilibrium L_2 at energy -1.673190 .

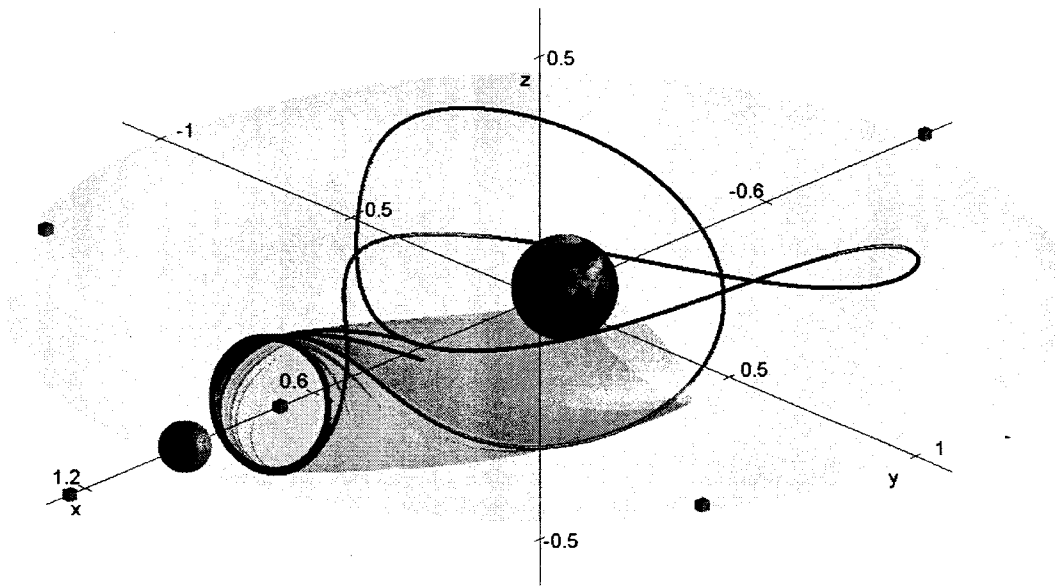


Figure 6.7: A homoclinic connection from a periodic orbit in family \mathbf{H}_1 to \mathbf{H}_1 at energy -1.673174 .

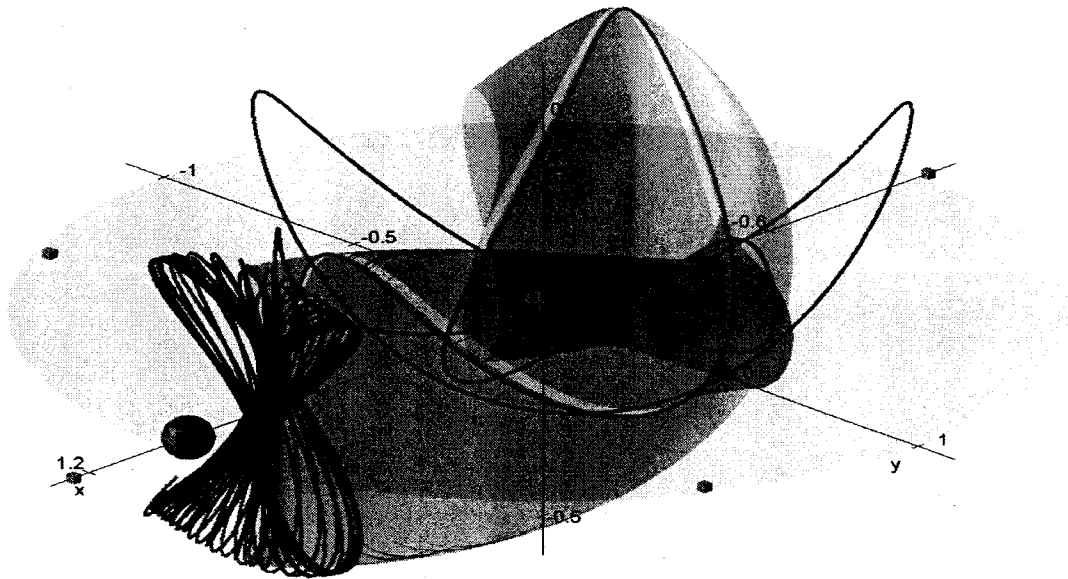


Figure 6.8: A heteroclinic connection from a periodic orbit in family V_1 to a torus at equilibrium L_1 at energy -1.507750 .

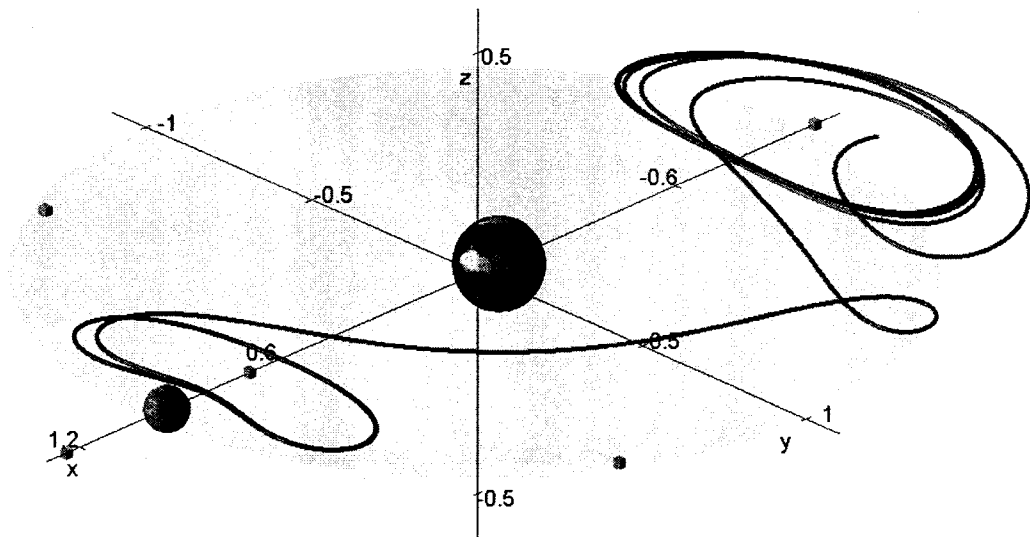


Figure 6.9: A heteroclinic connection from a periodic orbit in family L_1 to L_3 at energy -1.520881 .

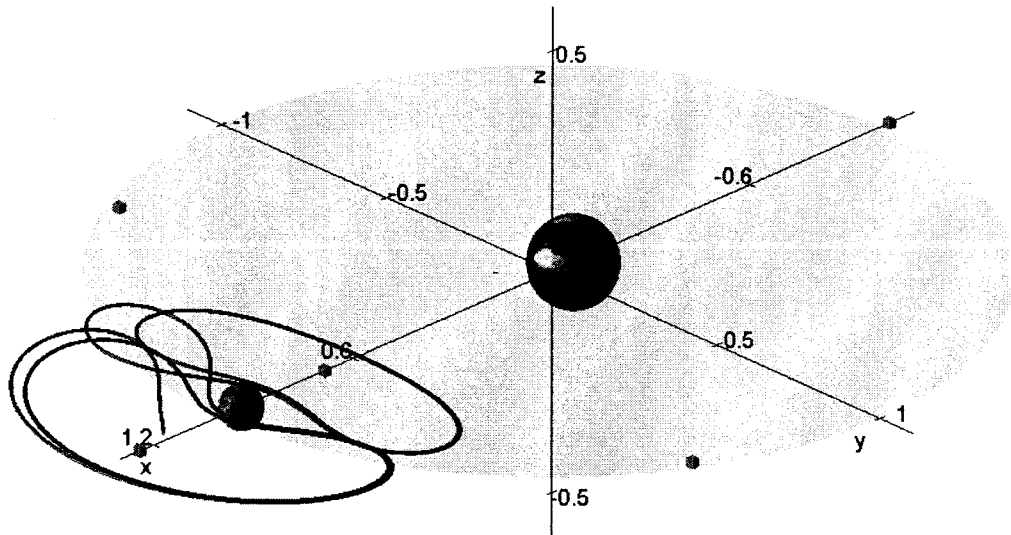


Figure 6.10: A heteroclinic connection from a periodic orbit in family L_1 to L_2 at energy -1.506991 .

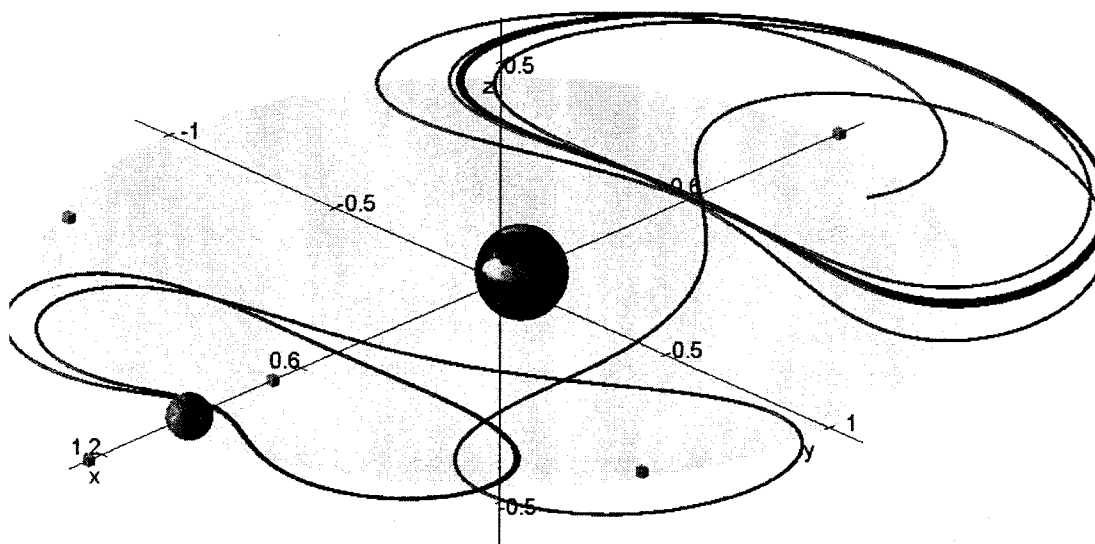


Figure 6.11: A heteroclinic connection from a periodic orbit in family L_1 to L_3 at energy -1.430828 .

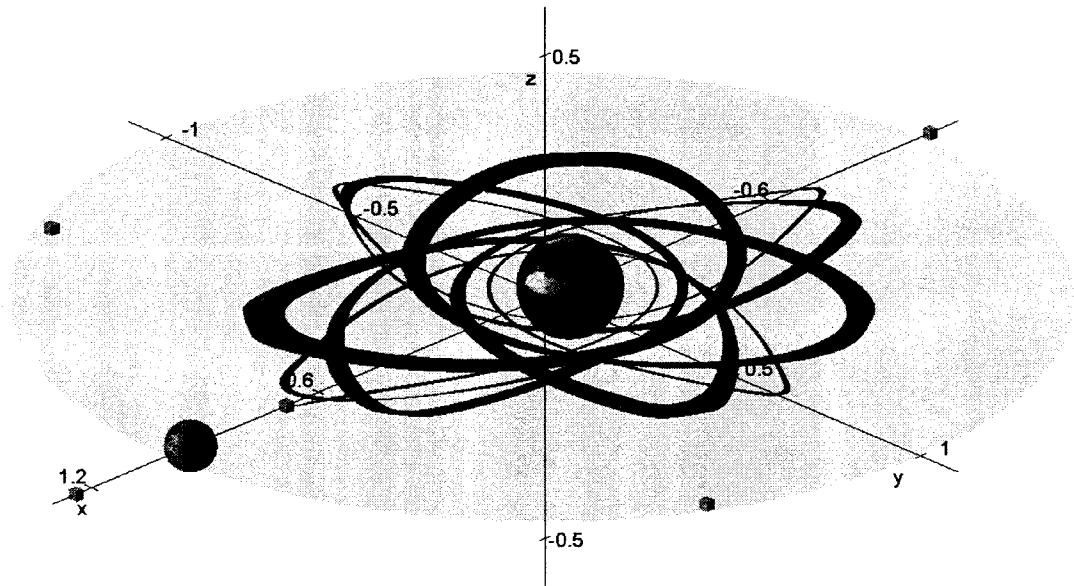


Figure 6.12: An unstable manifold of a periodic orbit in the family L_1 at energy -1.766528 .

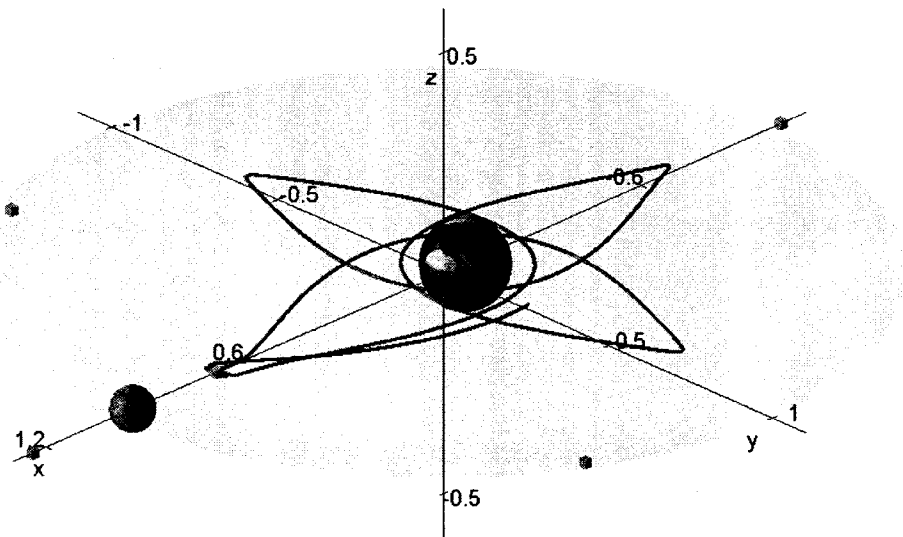


Figure 6.13: A homoclinic connection from a periodic orbit in family L_1 to L_1 at energy -1.764280 .

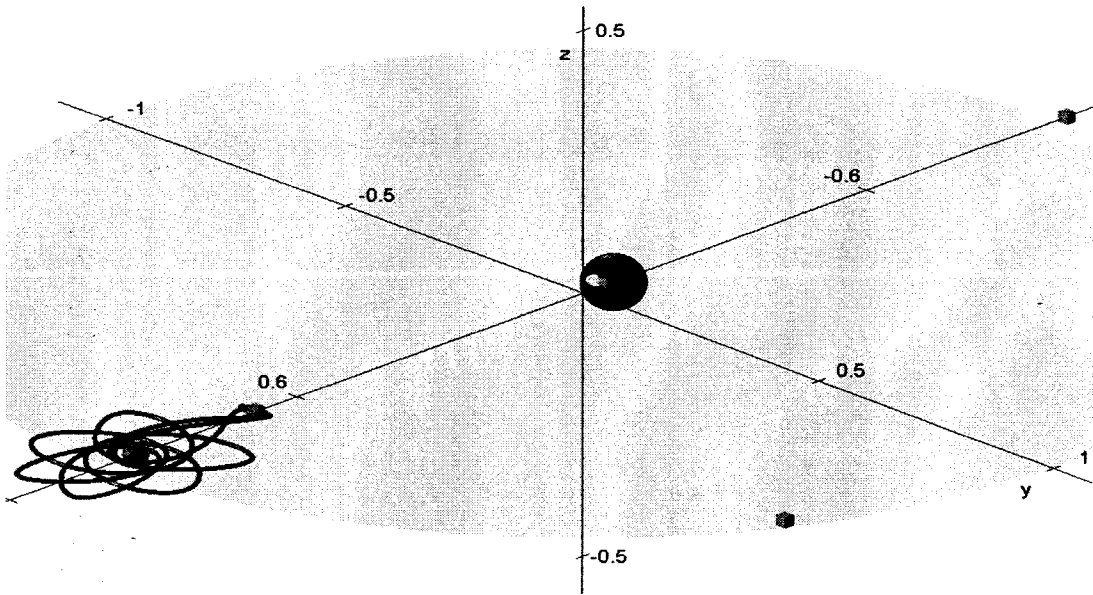


Figure 6.14: An unstable manifold containing homoclinic connections from a periodic orbit in family L_1 to L_1 around the Moon at energy -1.764280 .

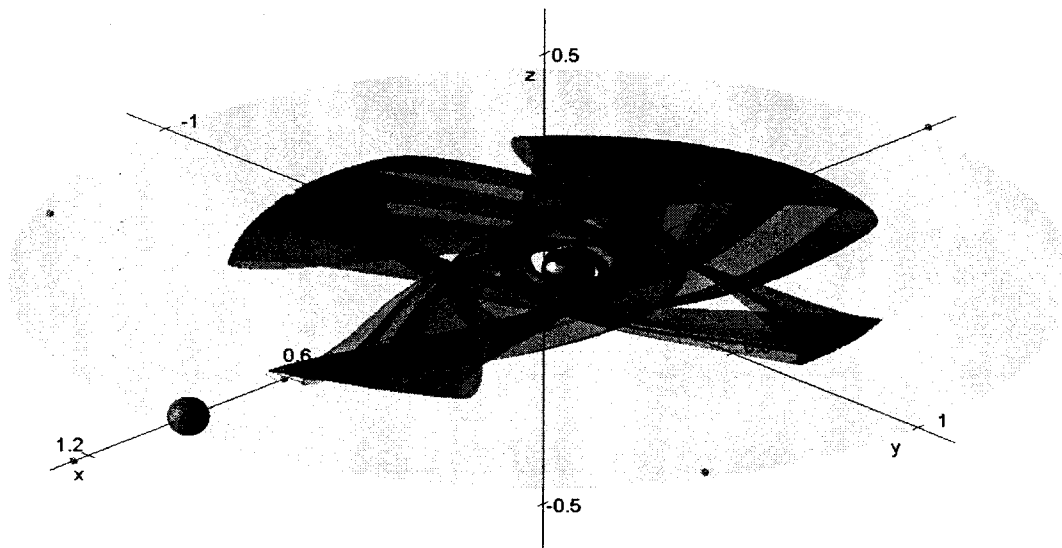


Figure 6.15: An unstable manifold containing homoclinic connections from a periodic orbit in family L_1 to L_1 at energy -1.761132 .

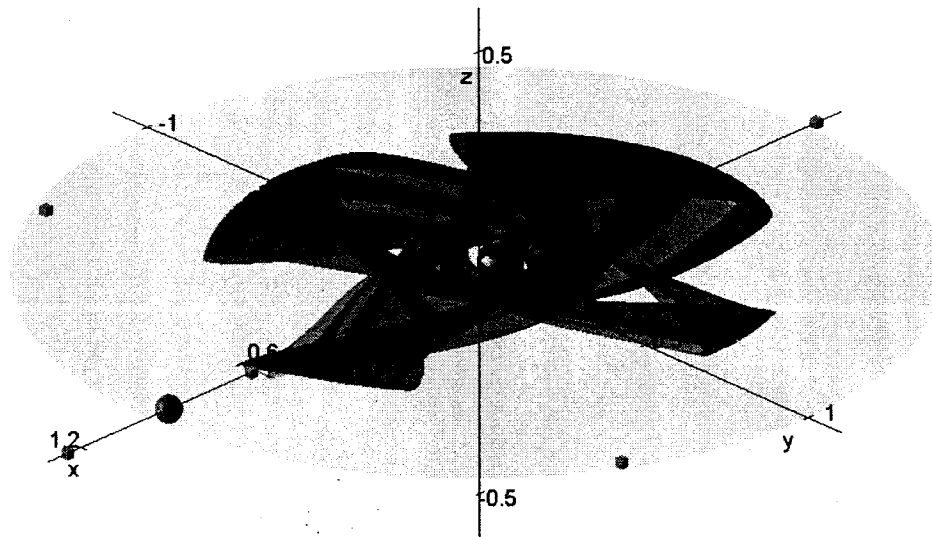


Figure 6.16: An unstable manifold containing homoclinic connections from a periodic orbit in family L_1 to L_1 at energy -1.761125 .

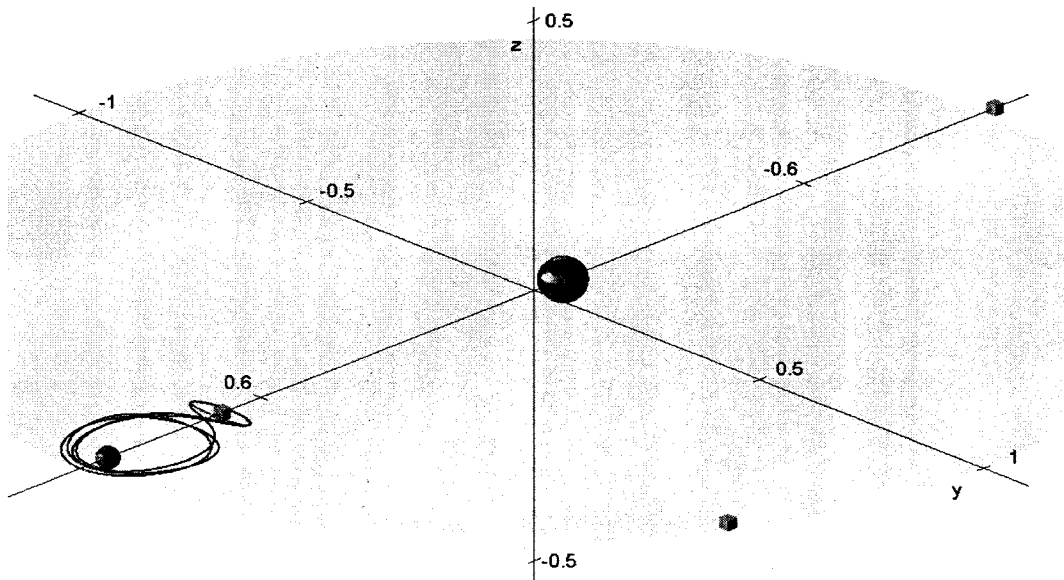


Figure 6.17: A homoclinic connection from a periodic orbit in family L_1 to L_1 around the moon at energy -1.754963 .

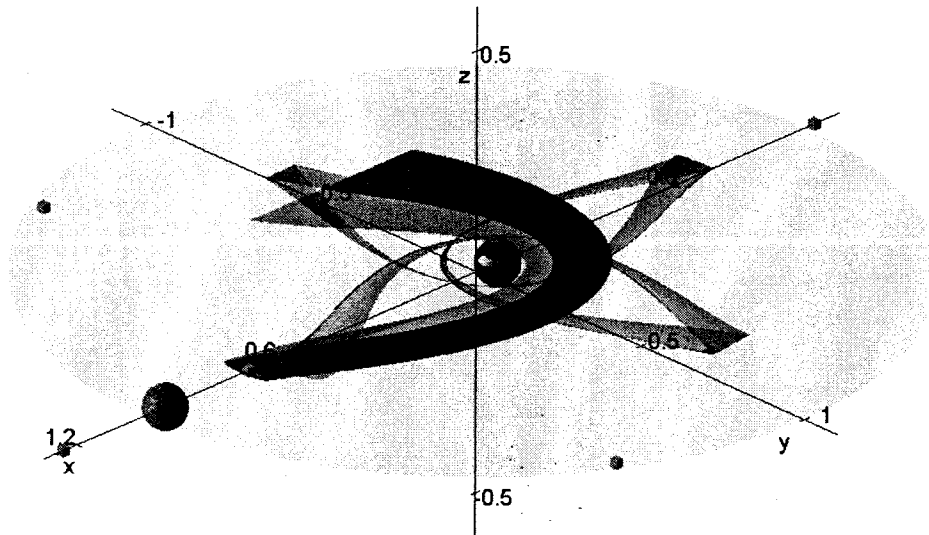


Figure 6.18: An unstable manifold containing homoclinic connections from a periodic orbit in family L_1 to L_1 at energy -1.754963 .

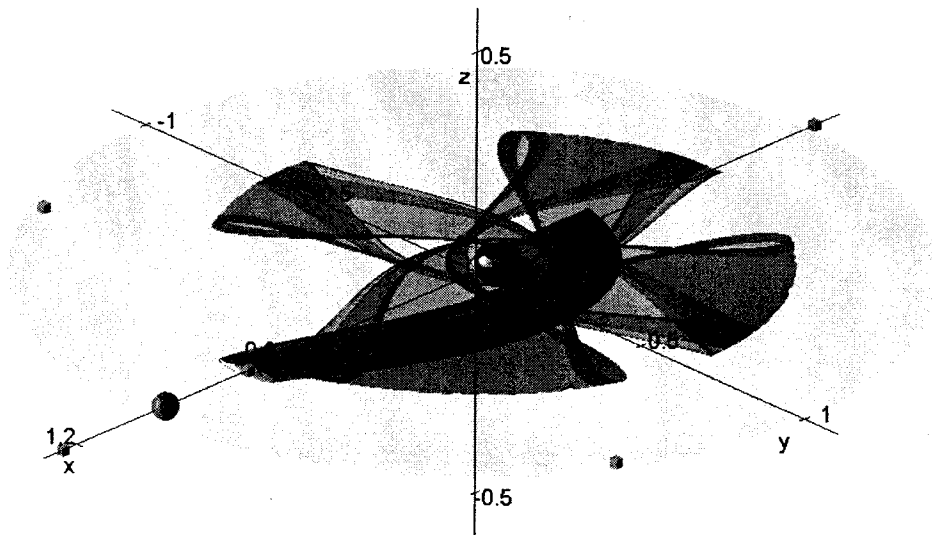


Figure 6.19: An unstable manifold containing homoclinic connections from a periodic orbit in family L_1 to L_1 at energy -1.749074 .

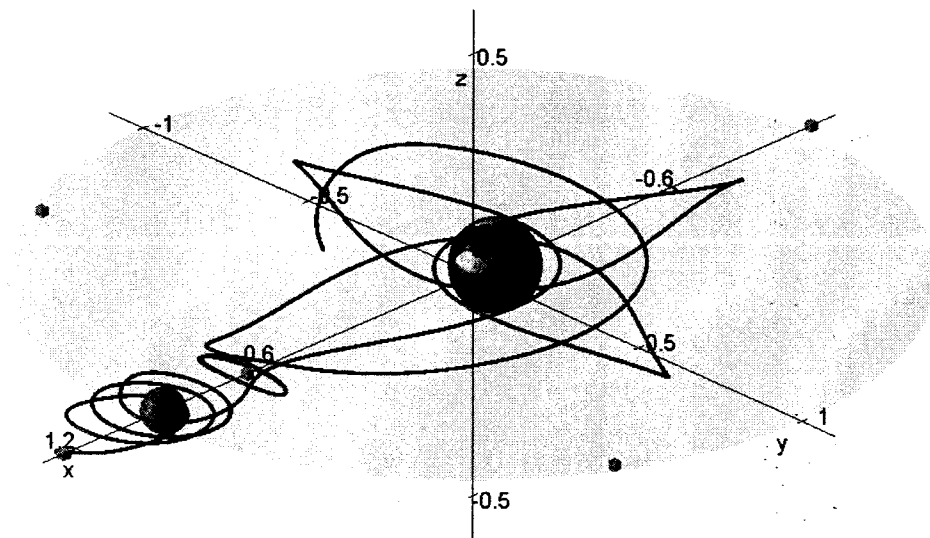


Figure 6.20: A heteroclinic connection from a periodic orbit in family L_2 to L_1 at energy -1.724892 .

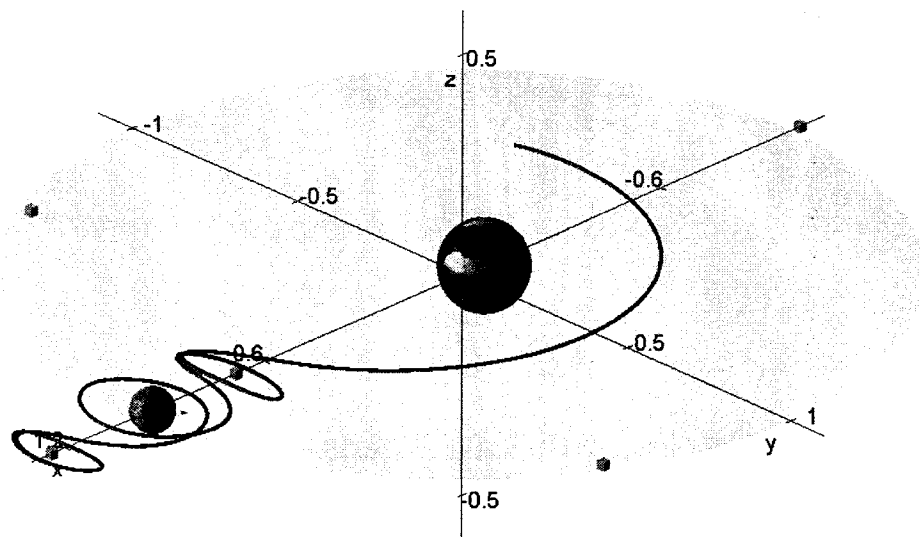


Figure 6.21: A heteroclinic connection from a periodic orbit in family L_2 to L_1 at energy -1.708555 .

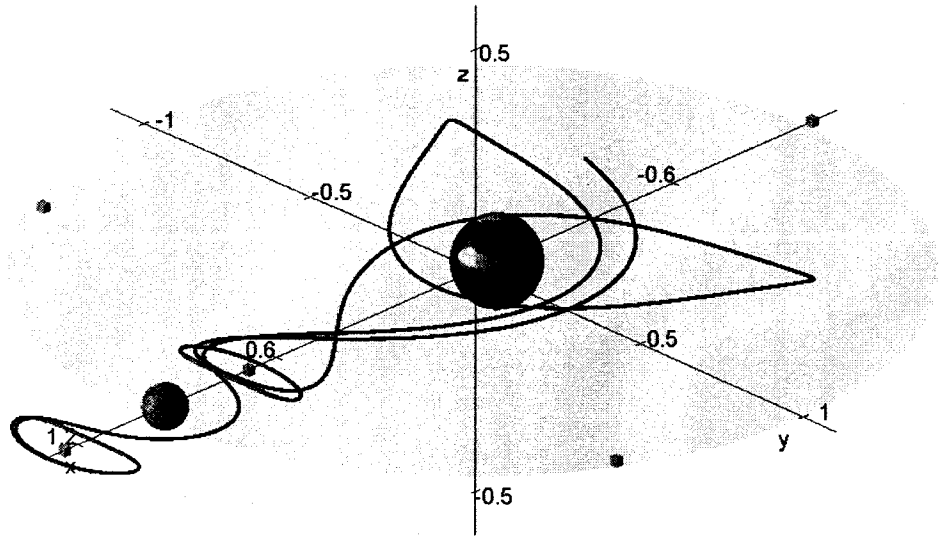


Figure 6.22: A heteroclinic connection from a periodic orbit in family L_2 to L_1 at energy -1.691739 .

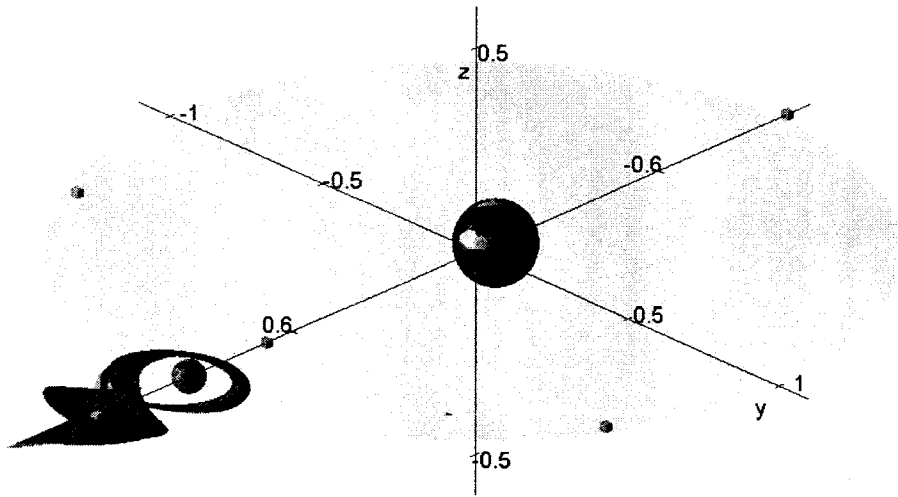


Figure 6.23: An unstable manifold containing homoclinic connections from a periodic orbit in family L_2 to L_2 around the Moon at energy -1.691739 .

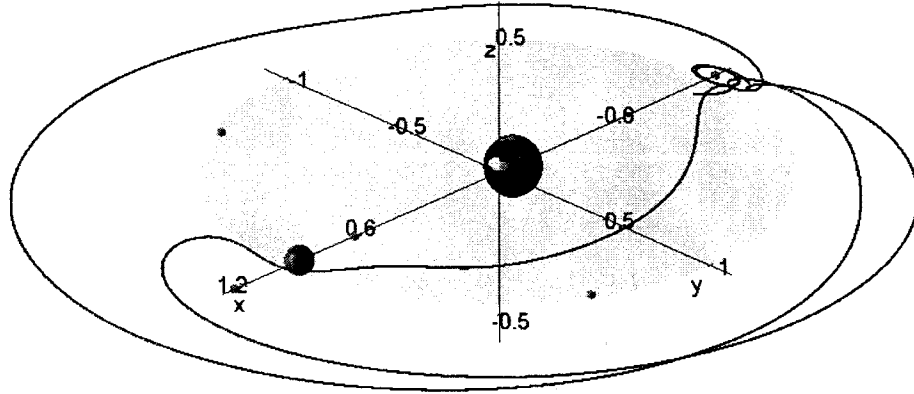


Figure 6.24: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.559424 .

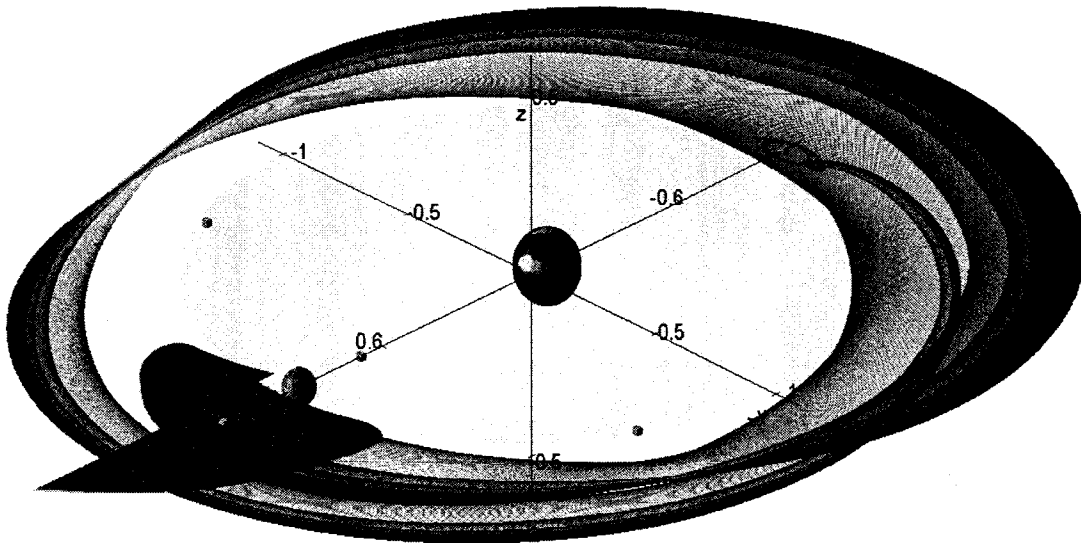


Figure 6.25: An unstable manifold containing heteroclinic connections from a periodic orbit in family L_3 to family L_1 at energy -1.559424 .

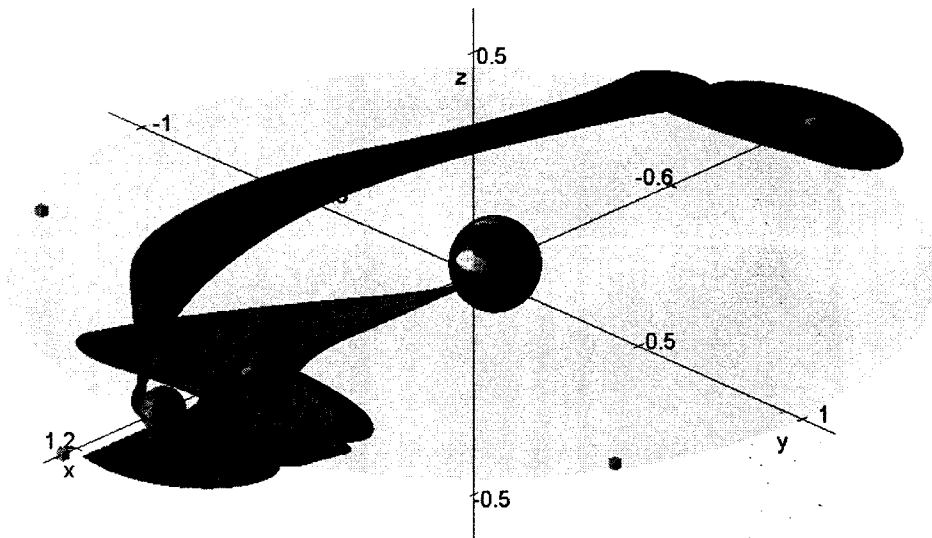


Figure 6.26: An unstable manifold containing heteroclinic connections from a periodic orbit in family L_3 to L_1 at energy -1.548840 .

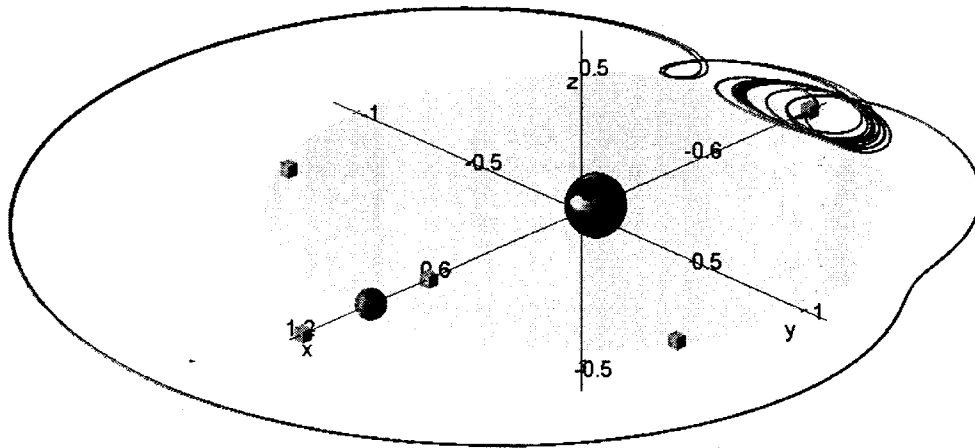


Figure 6.27: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.548840 .

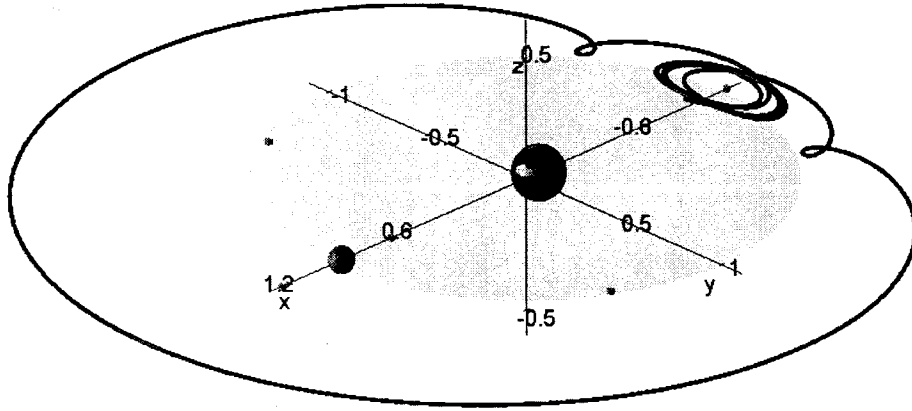


Figure 6.28: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.548840 .

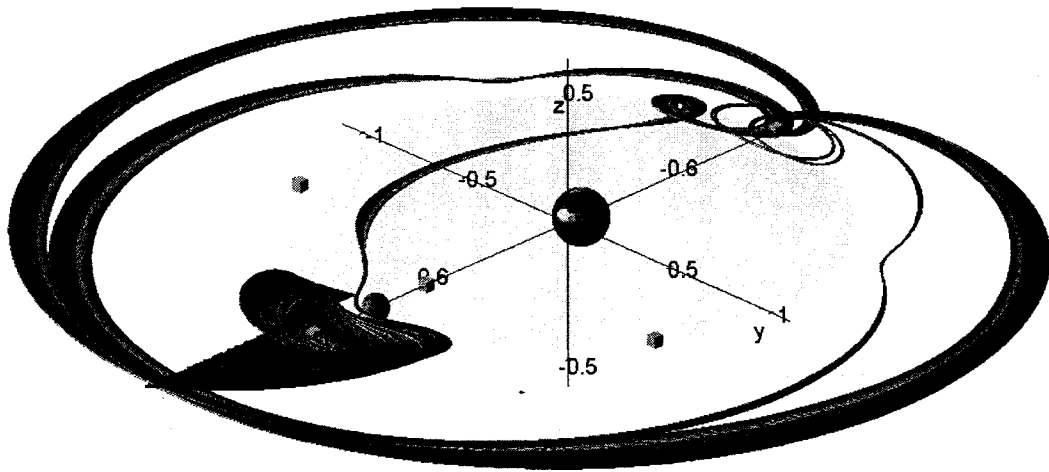


Figure 6.29: An unstable manifold containing homoclinic connections from a periodic orbit in family L_3 to L_3 at energy -1.548840 .

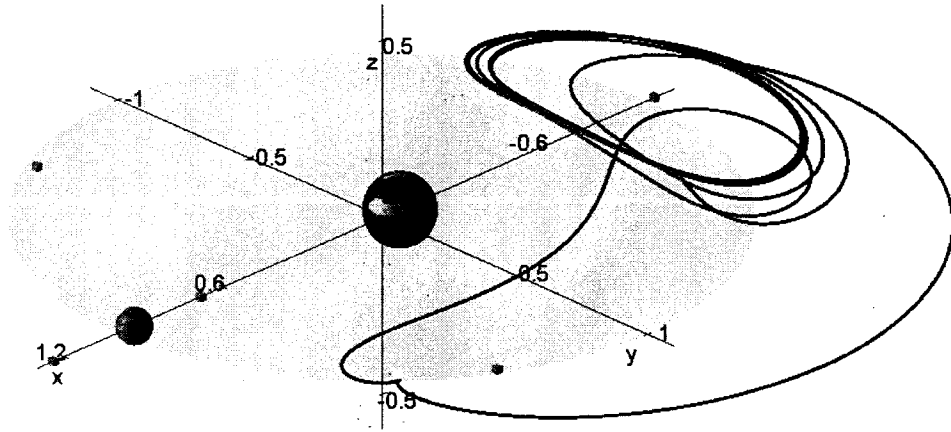


Figure 6.30: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.506991 .

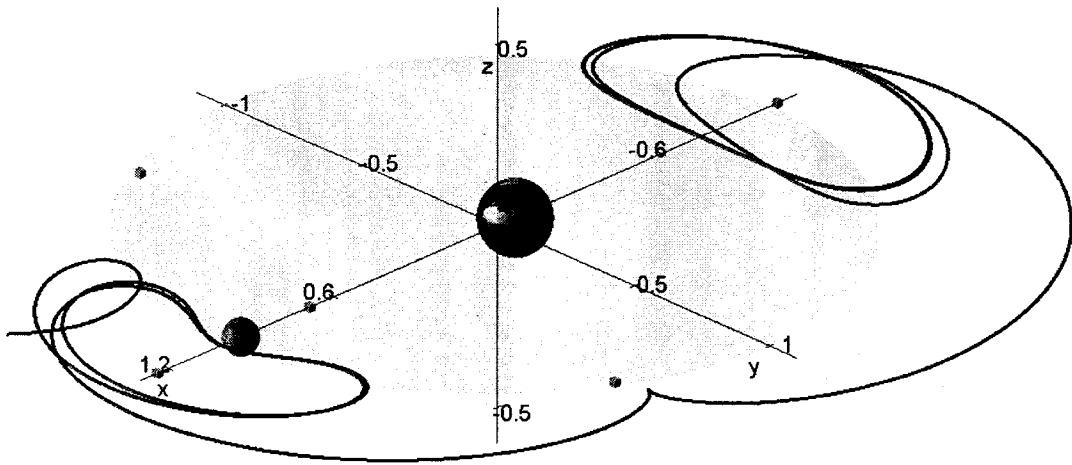


Figure 6.31: A heteroclinic connection from a periodic orbit in the family L_3 to L_2 at energy -1.506991 .

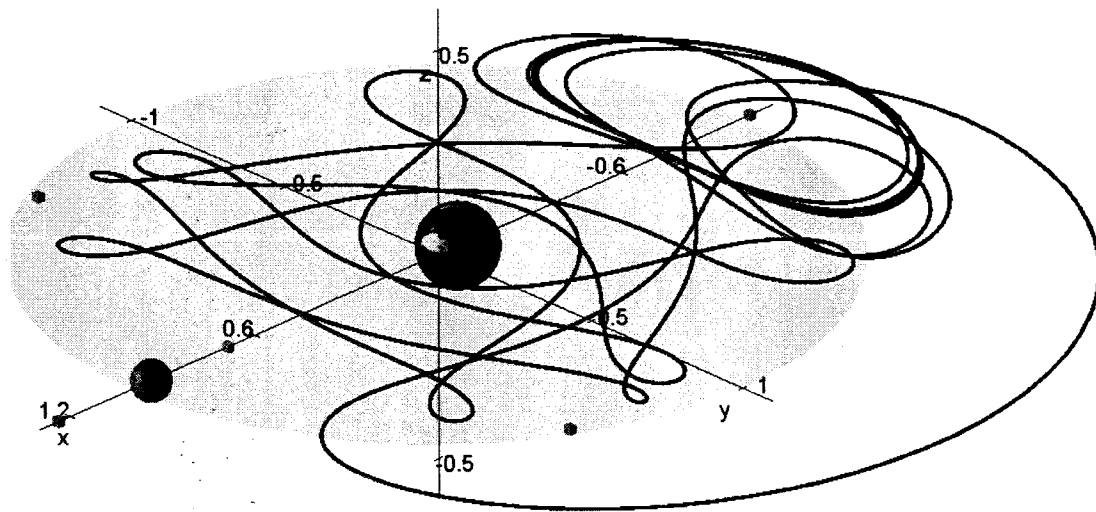


Figure 6.32: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.453254 .

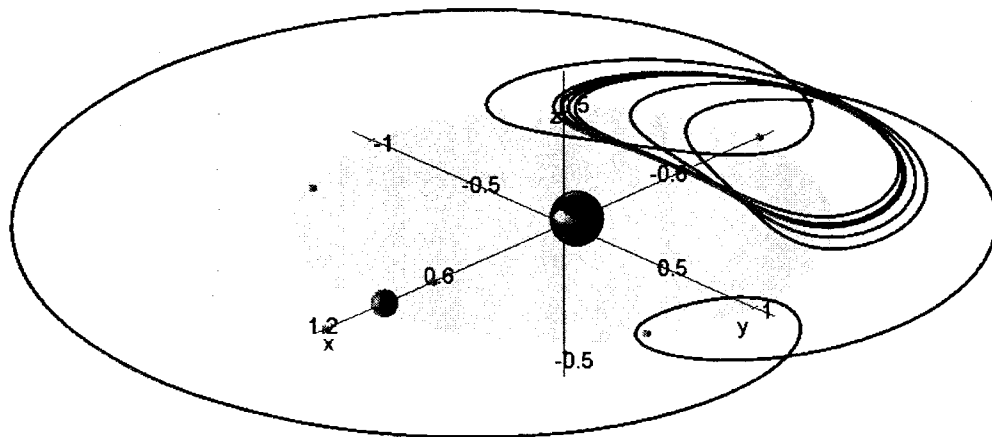


Figure 6.33: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.453254 .

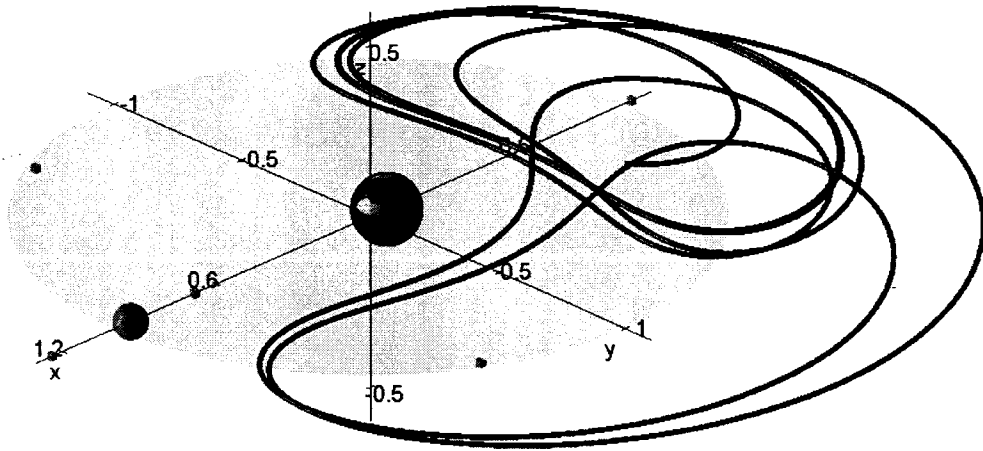


Figure 6.34: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.430830 .

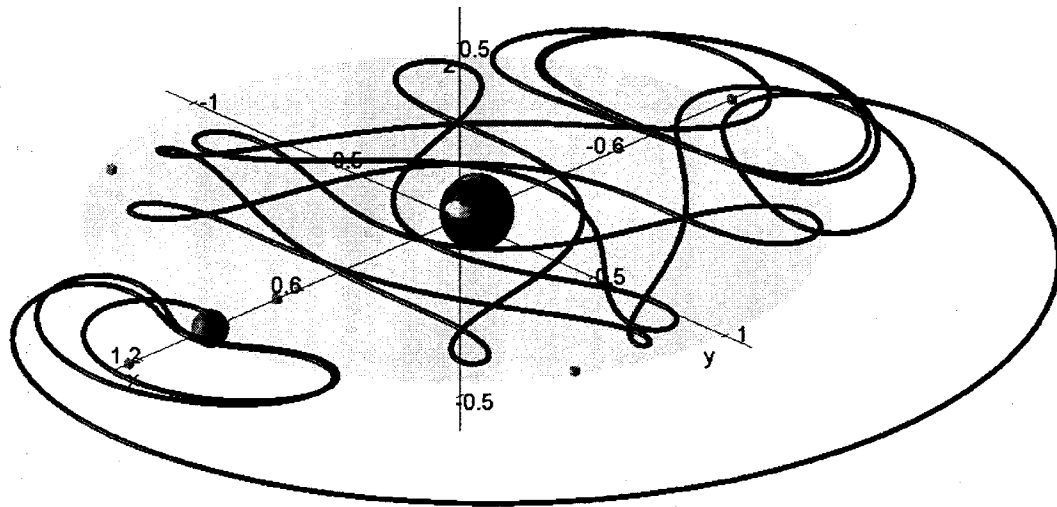


Figure 6.35: A heteroclinic connection from a periodic orbit in family L_3 to L_1 at energy -1.506991 .

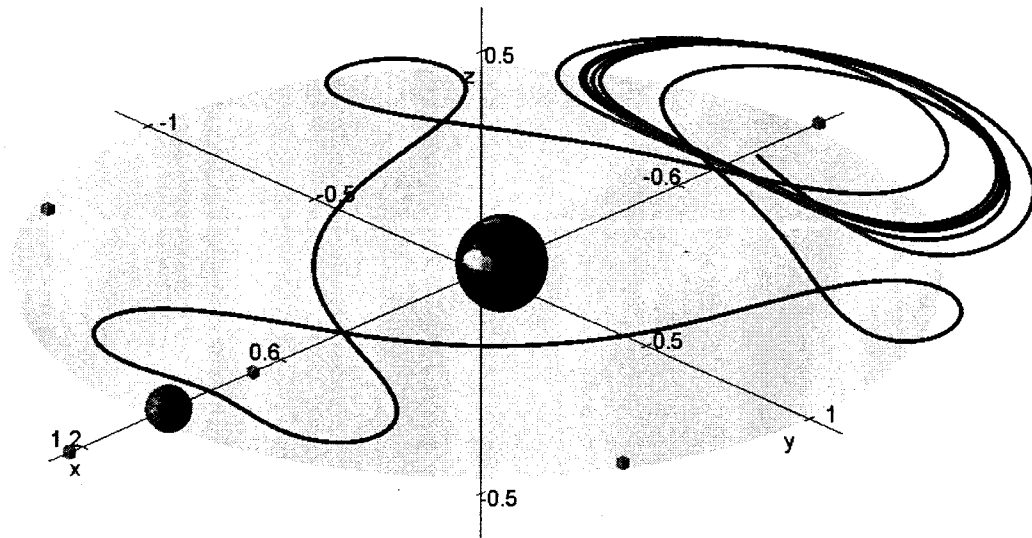


Figure 6.36: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.506991 .

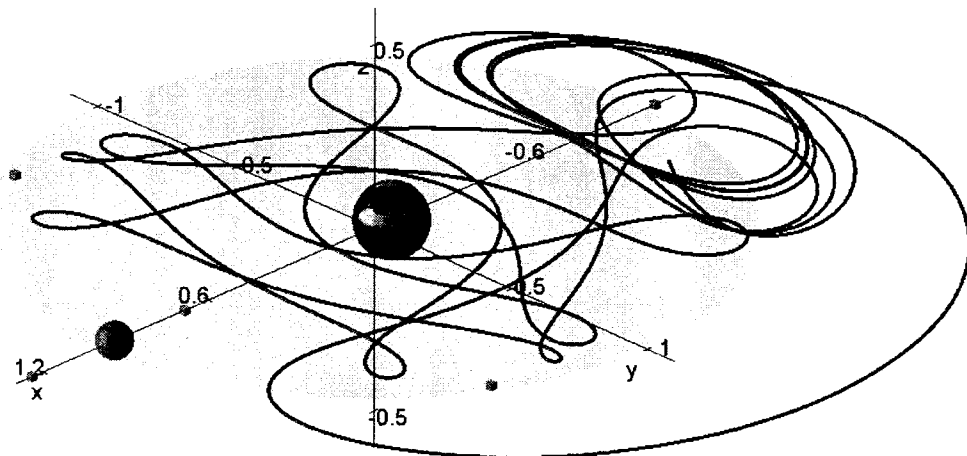


Figure 6.37: A homoclinic connection from a periodic orbit in family L_3 to L_3 at energy -1.506991 .

Chapter 7

Conclusions and Prospects

In this thesis, we started with the basic concepts of dynamical systems. Then we introduced general approaches to solve ODE systems. We discussed the features that distinguish AUTO from general ODE solvers. We also recalled the elementary stability and bifurcation facts, which are crucial for the problems we deal with in this thesis.

We used AUTO to study two problems. The first one is the Lorenz system. The second one is the CR3BP. We do not only use orbit continuation to compute stable/unstable manifold of these systems, but we also use various boundary condition settings to locate homoclinic and heteroclinic connections within the manifolds we compute. We use Python scripts to generate large amount of data. To deal the huge quantity of data, we use scripts for selecting and extracting key results. In addition, the graphical visualization tool QTPlaut greatly reduces our effort in selecting interesting connections, by browsing the manifolds (AUTO solutions) like turning a photo album. Another visualization tool MATPlaut provides high quality graphics processed by QTPlaut. Transparency effects, coloring schemes and marking facilities of the orbits/manifolds can be adjusted through a convenient graphical user interface. QTPlaut and MATPlaut are a combination of tens of thousand lines C/C++ code and MATLAB scripts. They are introduced in a separate document as a user manual.

Our work is far from complete. For the Lorenz system, we should go further in the study of the symbolic dynamics. We should try the continuation of the system with different system parameters. For the CR3BP, we need to find better classification and sorting methods

to clarify the possible connections, both homoclinic and heteroclinic, between periodic orbits and equilibria. In this thesis we only considered the 2-dimensional unstable manifold of the CR3BP. Higher dimensional unstable manifolds are also a direction of further study. We should study how the results are affected by μ .

To visualize the computational results of AUTO better, we need a measure such that the orbits can have a more uniform distance between every pair of adjacent orbits, so that the graphical mesh will look even and smooth. Thus in the orbit continuation process, we need to know information of a computed orbit while computing a new orbit. In our experiments, we found that the Python command line interface cannot deal with a file larger than 2 Giga-bytes. Although we can divide AUTO solutions into smaller ones, the programming becomes very complex and difficult to read. We hope that in future AUTO implementation these two points can be improved.

Bibliography

- [1] A. A. Andronov, A. A. Vitt, S. E. Khaikin. "Theory of Oscillators." Dover Publications, New York, 1987.
- [2] A. Chenciner, R. Montgomery. "A Remarkable Periodic Solution of the Three-Body Problem in the case of Equal Masses." *Annals of Mathematics* 152, p881-901, 2000.
- [3] A. D. Jepson. "Numerical Hopf Bifurcation." PhD thesis, Applied Mathematics, California Institute of Technology, 1981.
- [4] A. E. Roy. "The Foundations of Astrodynamics." New York: Macmillan, 1967.
- [5] B. Hassard, J. Zhang. "Existence of a homoclinic orbit of the Lorenz system by precise shooting." *SIAM J. Math. Anal.* 25, no.1, p179-196, 1994.
- [6] B. Hunt. "Invariants." Appendix B.1 in "The Geometry of Some Special Arithmetic Quotients." Springer-Verlag, New York, 1996.
- [7] B. Krauskopf, H. M. Osinga. "Two-dimensional global manifolds of vector fields." *Chaos* 9 (3), p768-774, 1999.
- [8] B. Krauskopf, H. M. Osinga. "Computing geodesic level sets on global (un)stable manifolds of vector fields." *SIAM Journal on Applied Dynamical Systems*. Vol. 4(2), p546-569, 2003.
- [9] B. Krauskopf, H. M. Osinga, E. J. Doedel, M. E. Henderson, J. Guckenheimer, A. Vladimirovsky, M. Dellnitz, O. Junge. "A survey of methods for computing (un)stable manifolds of vector fields." *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering* , Vol. 15(3), p763-791, 2005.

- [10] B. Krauskopf, H. M. Osinga. "Growing 1D and quasi-2D unstable manifolds of maps." *Journal of Computational Physics* 146 no. 1, p404-419, 1998.
- [11] B. Krauskopf, H. M. Osinga. "The Lorenz manifold as a collection of geodesic level sets." *Nonlinearity* 17 C1-C6, 2004.
- [12] C. Berge. "Topological Spaces Including a Treatment of Multi-Valued Functions, Vector Spaces and Convexity." Dover, New York, 1997.
- [13] C. B. Smith, B. Kuszta, G. Lyberatos, J. E. Bailey. "Period doubling and complex dynamics in an isothermal chemical reaction system." *Chem. Eng. Sci.* 38, p425-430, 1983.
- [14] C. de Boor, B. Swartz. "Collocation at Gaussian Points." *SIAM J. Numer. Anal.* 10, p582-606, 1973.
- [15] C. D. Meyer. "Matrix Analysis and Applied Linear Algebra." SIAM, 2001.
- [16] C. Gotsman, S. Gumhold, L. Kobbelt. "Simplification and Compression of 3d Meshes." *Tutorials on Multiresolution in Geometric Modelling*, p319-361, Springer, 2002.
- [17] C. H. Richardson. "An Introduction to the Calculus of Finite Differences." Van Nostrand, New York, 1954.
- [18] C. Kahlert, O. E. Rössler, A. Varma. "Chaos in a CSTR with two consecutive first-order reactions, one exo, one endothermic. In: *Proceedings Heidelberg workshop*." Springer Series Chem. Physics 1981.
- [19] C. Marchal. "The Three-Body Problem." Elsevier, 1990.
- [20] C. Perelló. "Intertwining invariant manifolds and Lorenz attractor." In *Global theory of dynamical systems (Proc. Internat. Conf., Northwestern Univ., Evanston, Ill., 1979)*, Lecture Notes in Math. 819, Springer-Verlag, Berlin, p375-378, 1979.
- [21] C. Sparrow. "The Lorenz Equations: Bifurcations, Chaos, and Strange Attractors" *Appl. Math. Sci.* 41, Springer-Verlag, New York, 1982.

- [22] C. T. Kelly, R. Suresh. "A new acceleration method for Newton's method at singular points." *SIAM J. Numer. Anal.* 20, p1001-1009, 1983.
- [23] C. Zhang. "Computation and Visualization of Periodic Orbits in the Circular Restricted Three-Body Problem." Master Thesis, Computer Science Department, Concordia University, 2004.
- [24] D. C. Lay. "Linear Algebra and Its Applications ." 3rd ed., Washington, Addison Wesley, 2005.
- [25] D. Viswanath. "Symbolic dynamics and periodic orbits of the Lorenz attractor." *Nonlinearity* 16, p1035-56, 2003.
- [26] D. V. Jorgensen, R. Aris. "On the dynamics of a stirred tank with consecutive reactions." *Chem. Eng. Sci.* 38, p45-53, 1983.
- [27] E. A. Coddington, N. Levinson. "Theory of Ordinary Differential Equations." McGraw-Hill, New York, 1955.
- [28] E. Hopf. "Abzweigung einer periodischen Lösung von einer stationären Lösung eines Differential systems." *Bericht der Math.-Phys Klasse der Sächsischen Akademie der Wissenschaften zu Leipzig* 94, 1942.
- [29] E. J. Doedel, V. Romanov, R. C. Paffenroth, H. B. Keller, D. J. Dichmann, J. Galán-Vioque, A. Vanderbauwhede. "Elemental periodic orbits associated with the libration points in the circular restricted 3-body problem." *Internat. J. Bifur. Chaos Appl. Sci. Engrg.* 17 no.8, 2007.
- [30] E. J. Doedel. "Lecture notes on numerical analysis of nonlinear equations. Numerical continuation methods for dynamical systems", *Understanding Complex Systems*, Springer, Dordrecht, 2007.
- [31] E. J. Doedel, H. B. Keller, J. P. Kernévez. "Numerical Analysis and Control of Bifurcation Problems (II) Bifurcation in Infinite Dimensions." *International Journal of Bifurcation and Chaos* 1(4), p745-772, 1991.

- [32] E. J. Doedel, B. Krauskopf and H. M. Osinga. "Global bifurcations of the Lorenz manifold." *Nonlinearity* 19 no. 12, p2947-2972, 2006.
- [33] E. J. Doedel, R. C. Paffenroth, H. B. Keller, D. J. Dichmann, J. Galán, A. Vanderbauwhede. "Computation of Periodic Solutions of Conservative Systems with Application to the 3-Body Problem." *International Journal of Bifurcation and Chaos* 13(6), p1-29, 2003.
- [34] E. J. Doedel *et al.* "AUTO-07p." <http://cmvl.cs.concordia.ca/auto/>.
- [35] E. J. Doedel. "AUTO, a Program for the Automatic Bifurcation Analysis of Autonomous Systems." *Congr. Numer.* 30, p265-384, 1981.
- [36] E. Reithmeier. "Periodic solutions of Nonlinear Dynamical Systems." Springer, Berlin, 1991.
- [37] E. W. Weisstein. "MathWorld—A Wolfram Web Resource." <http://mathworld.wolfram.com>.
- [38] F. Hausdorff. "Grundzüge der Mengenlehre." von Veit, Leipzig, Germany, 1914. Republished as "Set Theory." 2nd ed. Chelsea, New York, 1962.
- [39] F. J. Muñoz-Almaraz, E. Freire, J. Galán, E. J. Doedel and A. Vanderbauwhede. "Continuation of Periodic Orbits in Conservative and Hamiltonian Systems." *Physica D*, Vol. 181 No. 1-2, p1-38, 2003.
- [40] G. Boole, J. F. Moulton. "A Treatise on the Calculus of Finite Differences." 2nd rev. ed. Dover, New York, 1990.
- [41] H. Anton. "Elementary Linear Algebra (Applications Version)." 9th ed., Hoboken, Wiley International, NJ, 2005.
- [42] H. B. Stewart. "Visualization of the Lorenz system." *Physica D: Nonlinear Phenomena*, Vol 18 Issue 1-3, p479-480, 1986.

- [43] H. Kokubu, R. Roussarie. "Existence of a singularly degenerate heteroclinic cycle in the Lorenz system and its dynamical consequences, Part I." *Journal of Dynamics and Differential Equations* 16, p513-557, 2004.
- [44] H. M. Osinga, B. Krauskopf. "Visualizing the structure of chaos in the Lorenz system." *Comput. Graph.* 26, p815-823, 2002.
- [45] H. M. Osinga, B. Krauskopf. "Crocheting the Lorenz manifold." *Math. Intelligencer* 26, p25-37, 2004.
- [46] H. S. M. Coxeter, S. L. Greitzer. "Geometry Revisited." *Math. Assoc. Amer.*, Washington, DC, 1967.
- [47] H. T. Croft, K. J. Falconer, R. K. Guy. "Unsolved Problems in Geometry." Springer-Verlag, New York, 1991.
- [48] J. Barrow-Green. "Poincaré and the Three Body Problem." *Amer. Math. Soc.*, 1996.
- [49] J. Guckenheimer, P. Worfolk. "Dynamical Systems: Some Computational Problems, in Bifurcations and Periodic Orbits of Vector Fields." *NATO Advanced Science Institutes Series C: Mathematical and Physical Sciences*, 408 Kluwer Academic Publishers Group, Dordrecht, 1993.
- [50] J. Guckenheimer, P. Holmes. "Nonlinear Oscillations, Dynamical Systems and Bifurcation of Vector Fields." Springer-Verlag, New York, 1983.
- [51] J. Guckenheimer. "Dimension Estimates for Attractors." In *Fluids and Plasmas: Geometry and Dynamics* (Boulder, Colo., 1983), *Amer. Math. Soc.*, Providence, RI, p357-367, 1984.
- [52] J. Guckenheimer, P. Holmes. "Nonlinear oscillations, dynamical systems, and bifurcations of vector fields." Revised and corrected reprint of the 1983 original. *Applied Mathematical Sciences*, 42. Springer-Verlag, New York, 1990.
- [53] J. Guckenheimer. "Numerical Analysis of Dynamical Systems." *Amer. Math. Soc.*, Providence, RI, 1999.

- [54] J. H. Hubbard. "Differential Equations: A Dynamical Systems Approach Part I", Springer Verlag, New York, 1991.
- [55] J. H. Hubbard. "Differential Equations: A Dynamical Systems Approach Part II: Higher-Dimensional Systems", Springer Verlag, New York, 1995.
- [56] J. K. Hale. "Ordinary Differential Equations." Wiley-Interscience, New York, 1969.
- [57] J. L. Kaplan, J. A. Yorke. "Preturbulence, a regime observed in a fluid flow model of Lorenz ." Math. Phys. 67, P93-108, 1979.
- [58] J. M. A. Danby. "Fundamentals of Celestial Mechanics." The MacMillan Company, 1962.
- [59] J. M. Lee. "Introduction to topological manifolds. (English summary) Graduate Texts in Mathematics, 202." Springer-Verlag, New York, 2000.
- [60] J. M. T. Thompson, H. B. Stewart. " Nonlinear Dynamics and Chaos. Geometrical Methods for Engineers and Scientists." John Wiley, Chichester, 1986
- [61] J. Moser. "On the theory of quasiperiodic motions." SIAM Review 8, p145-172, 1966.
- [62] J. Naimark. "On some cases of periodic motions depending on parameters." Dokl. Akad. Nauk. SSSR 129, p736-739, 1959.
- [63] J. P. England. "Advances in computing global invariant manifolds." PhD thesis, Engineering Mathematics, University of Bristol, 2005.
- [64] J. Rinzel, R. N. Miller. "Numerical calculation of stable and unstable periodic solutions to the Hodgkin-Huxley equations." Math. Biosci. 49, p27-59, 1980.
- [65] J. R. Munkres. "Topology: A First Course." 2nd ed. Prentice-Hall, Upper Saddle River, NJ, 2000.
- [66] K. Lust. "Improved Numerical Floquet multipliers." Int. J. Bifurcation and Chaos. 11(9), p2389-2410, 2001.

- [67] M. Dellnitz, A. Hohmann. "The Computation of Unstable Manifolds Using Subdivision and Continuation." In H. W. Broer *et al.* (eds.), *Progress in Nonlinear Differential Equations and Their Applications*, Birkhäuser Verlag, Basel Switzerland, Vol. 19, p449-459, 1996.
- [68] M. Abramowitz, I. A. Stegun. "Differences" in "Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables." 9th printing. Dover, New York, 1972.
- [69] M. E. Johnson, M. S. Jolly, I. G. Kevrekidis. "Two-dimensional invariant manifolds and global bifurcations: some approximation and visualization studies." *Numerical Algorithms*, Springer Netherlands, Vol. 14, Numbers 1-3 April, p125-140, 1997.
- [70] M. Kubíček, M. Holodniok. "Numerical determination of bifurcation points in steady-state and periodic solutions—numerical algorithms and examples." *ISNM 70*, p247-270, 1984. In:[91].
- [71] M. Urabe. "Nonlinear Autonomous Oscillations." Academic Press, New York, 1967.
- [72] Minnesota Space Frontier Society. "The Earth-Moon System."
<http://www.freemars.org/l5/aboutl5.html> .
- [73] O. Ore, S. L. Greitzer. "Graphs and their uses." Random House, New York, 1963.
- [74] P. Glendinning, C. Sparrow. "Local and global behavior near homoclinic orbits." *J. Stat. Phys.* 35 p645-96, 1984.
- [75] P. Hartman. "Ordinary Differential Equations." Wiley, New York, 1964.
- [76] P. J. Olver. "Classical Invariant Theory." Cambridge University Press, Cambridge, England, 1999.
- [77] R. Chen. "Computation of invariant manifolds using AUTO-07p ." Master Thesis, Computer Science Department, Concordia University, 2007.
- [78] R. D. Russell, J. Christiansen. "Adaptive Mesh Selection Strategies for Solving Boundary Value Problems." *SIAM J. Numer. Anal.* 15, p59-80, 1978.

- [79] R. H. Abraham, C. D. Shaw. "Dynamics—The Geometry of Behavior. Part three: global behavior." Aerial Press, Santa Cruz, 1985.
- [80] R. S. Sacker. "On invariant surfaces and bifurcations of periodic solutions of ordinary differential equations." *Comm. Pure Appl. Math* 18, p717-732, 1965.
- [81] R. Seydel. "Practical Bifurcation and Stability Analysis: From Equilibrium to Chaos." Second Edition, Springer-Verlag, New York, 1994.
- [82] S. G. Krantz. "Handbook of Complex Variables." Birkhäuser, Boston, MA, 1999.
- [83] S. H. Friedberg, A. J. Insel, L. E. Spence. "Linear Algebra", Prentice , New Delhi, 2007.
- [84] S. J. Leon. "Linear Algebra With Applications." 7th ed., Macmillan, Inc., New York, 1980.
- [85] S. N. Rasband. "Invariant Manifolds." in "Chaotic Dynamics of Nonlinear Systems." Wiley, New York, 1990.
- [86] S. P. Hastings, W. C. Troy. "A proof that the Lorenz equations have a homoclinic orbit." *J. Differential Equations* 113 no.1, p166-188, 1994.
- [87] S. P. Hastings, W. C. Troy. "A shooting approach to the Lorenz equations." *Bull. Amer. Math. Soc. (N.S.)* 27 no. 2, p298-303, 1992.
- [88] S. S. Rao. "Applied Numerical Methods for Engineers and Scientists." Prentice Hall, 2002.
- [89] S. Wiggins. "Invariant Manifolds: Linear and Nonlinear Systems." in "Introduction to Applied Nonlinear Dynamical Systems and Chaos." Springer-Verlag, New York, 1990.
- [90] T. F. Fairgrieve, A. D. Jepson. "O. K. Floquet Multipliers." *SIAM J. Numer. Anal.* 28(5), p1446-1462, 1991.
- [91] T. Küpper, H. D. Mittelmann, H. Weber. "Numerical Methods for Bifurcation Problems. Proceedings of a conference in Dortmund." 1983, ISNM 70 Birkhäuser, Basel 1984.

- [92] T. N. Chan. "Numerical Bifurcation Analysis of Simple Dynamical Systems." Master Thesis, Computer Science Department, Concordia University, 1983.
- [93] GTS. "GNU Triangulated Surface Library: An Open Source Free Software Library." <http://gts.sourceforge.net/> .
- [94] The MathWorks Company, "MATLAB - The Language Of Technical Computing", US, <http://www.mathworks.com/products/matlab/> .
- [95] The Mesh Viewer. "A Lightweight Application for Displaying Three Dimensional Models", <http://mview.sourceforge.net/> .
- [96] Nokia, "QT: A Cross-platform Application and UI framework", Norway, <http://qt.nokia.com/> .
- [97] U. M. Ascher, J. Christiansen, R. D. Russell. "Collocation software for boundary value ODEs." ACM Trans. Math. Software Vol. 7, p209-222, 1981.
- [98] U. M. Ascher, J. Christiansen, R. D. Russell. "COLSYS: Collocation Software for Boundary-Value ODEs." ACM Trans. Math. Software Vol. 7, p223-229, 1981.
- [99] U. M. Ascher, R. M. M. Mattheij, R. D. Russell. "*Numerical solution of boundary value problems for ordinary differential equations.*" Prentice-Hall, 1988; SIAM, 1995.
- [100] U. M. Ascher, L. R. Petzold. "Computer Methods for ODEs and DAEs." SIAM, 1998.
- [101] V. Franceschini, C. Tcbaldi. "Sequences of infinite bifurcations and turbulence in a five-mode truncation of the Navier-Stokes equations." J. Phys. 21, p707, 1979.
- [102] V. Romanov. "Elemental Periodic Solutions of the Circular Restricted Three-Body Problem." Master Thesis, Computer Science Department, Concordia University, 2005.
- [103] V. I. Arnold. "Geometrical Methods in the Theory of Ordinary Differential Equations." Springer, New York, 1983.
- [104] V. Szebehely. "Theory of Orbits. The Restricted Problem of Three Bodies." Academic Press, New York, 1967.

- [105] W. Hahn. "Stability of Motion ." Springer, Berlin, 1967.
- [106] Wikipedia, the free encyclopedia, <http://en.wikipedia.org/wiki>.
- [107] X. Chen. "Lorenz equations, I: Existence and nonexistence of homoclinic orbits." SIAM J. Math. Anal. 27, p1057-1069, 1996. "II: Randomly rotated homoclinic orbits and chaotic trajectories." Disc. Cont. Dynam. Sys. 2 , p121-140, 1996. "III: Existence of hyperbolic sets." preprint.
- [108] Y. A. Kuznetsov. "Elements of Applied Bifurcation Theory." Third Edition, Springer, New York, 2004.

Appendix A

Computing Heteroclinic Connections of the Lorenz Manifold

In this chapter, we list the technical details about how to compute 1,024 heteroclinic connections of the Lorenz manifold, starting from $\mathbf{0}$ the ending at the secondary equilibria.

A.1 The equation file `man.f`

```
C-----  
C  man :  Stable manifold of the origin in the Lorenz model  
C-----  
  
      SUBROUTINE FUNC(NDIM,U,ICP,PAR,IJAC,F,DFDU,DFDP)  
C  -----  
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)  
      DIMENSION U(NDIM),PAR(*),F(NDIM)  
      rho  = PAR(1);beta  = PAR(2);sigma = PAR(3); T=PAR(11)  
      F(1)= T * (sigma * (U(2)- U(1)))  
      F(2)= T * (rho*U(1) - U(2) - U(1)*U(3))  
      F(3)= T * (U(1)*U(2) - beta*U(3))  
      RETURN
```



```

END

C-----
SUBROUTINE STPNT(NDIM,U,PAR,time)
C -----
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
DIMENSION U(NDIM),PAR(*)
DIMENSION V1(3),V2(3),W1(3),W2(3)
rho = 28.; beta = 8.d0/3.d0; sigma= 10.
PAR(1) = rho; PAR(2) = beta; PAR(3) = sigma
C Get eigenvectors at the origin:
CALL EIGVO(NDIM,PAR,V1,V2)
C Set initial approximate solution (for small T)
rad = 0.1; theta = 0.; T = -1e-5; RL= 1e-5
PI = 4*DATAN(1.d0); angle= 2*PI*theta
cs = DCOS(angle); sn = DSIN(angle)
U(1)= rad * ( cs*V2(1) + sn*V1(1) )
U(2)= rad * ( cs*V2(2) + sn*V1(2) )
U(3)= rad * ( cs*V2(3) + sn*V1(3) )
PAR(4) = rad; PAR(5) = theta
C At time=1 (at one of the nonzero equilibria):
x1 = -SQRT(beta*(rho - 1)); y1 = x1; z1 = rho - 1
CALL EIGV1(NDIM,PAR,W1)
PAR(8)=(U(1)-x1)*W1(1)+(U(2)-y1)*W1(2)+(U(3)-z1)*W1(3)
C At time=1 (at the other nonzero equilibrium):
x2 = SQRT(beta*(rho - 1)); y2 = x2; z2 = rho - 1
CALL EIGV2(NDIM,PAR,W2)
PAR(9)=(U(1)-x2)*W2(1)+(U(2)-y2)*W2(2)+(U(3)-z2)*W2(3)
PAR(10) = RL; PAR(11) = T
RLnom = 1030.; Tnom = -14.; Cnom = 300.
PAR(21) = RLnom; PAR(22) = Tnom; PAR(23) = Cnom
PAR(24) = (RLnom-RL)*ABS(T-Tnom) - Cnom
RETURN
END

```

```

C-----
SUBROUTINE BCND(NDIM,PAR,ICP,NBC,U0,U1,FB,IJAC,DBC)
C -----
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
DIMENSION PAR(*),ICP(*),U0(NDIM),U1(NDIM),FB(NBC),DBC(NBC,*)
DIMENSION V1(3),V2(3),W1(3),W2(3)
PI = 4*DATAN(1.d0)
rho = PAR(1); beta = PAR(2); sigma = PAR(3)
rad = PAR(4); theta = PAR(5); angle = 2*PI*theta
cs = DCOS(angle); sn = DSIN(angle)
C At time=0
CALL EIGVO(NDIM,PAR,V1,V2)
FB(1)= U0(1) - rad * ( cs*V2(1) + sn*V1(1) )
FB(2)= U0(2) - rad * ( cs*V2(2) + sn*V1(2) )
FB(3)= U0(3) - rad * ( cs*V2(3) + sn*V1(3) )
C At time=1 (at one of the nonzero equilibria):
x1 =-SQRT(beta*(rho - 1)); y1 = x1; z1 = rho - 1
CALL EIGV1(NDIM,PAR,W1)
FB(4)= (U1(1)-x1)*W1(1)+(U1(2)-y1)*W1(2)+(U1(3)-z1)*W1(3)-PAR(8)
C At time=1 (at the other nonzero equilibrium):
x2 = SQRT(beta*(rho - 1)); y2 = x2; z2 = rho - 1
CALL EIGV2(NDIM,PAR,W2)
FB(5)= (U1(1)-x2)*W2(1)+(U1(2)-y2)*W2(2)+(U1(3)-z2)*W2(3)-PAR(9)
C A constraint on T and L:
RL = PAR(10); T = PAR(11); RLnom= PAR(21)
Tnom = PAR(22); Cnom = PAR(23)
FB(6)= (RLnom-RL)*ABS(T-Tnom) - Cnom - PAR(24)
EPSD = 50.0; PAR(7) = 1.0
IF (ABS(PAR(8)) .LE. 1e-5 .OR. ABS(PAR(9)).LE. 1e-5 ) THEN
PAR(7) = 0.0
ENDIF
IF (PAR(25) .GE. EPSD .OR. PAR(26) .GE. EPSD )THEN
PAR(7)=1.0

```

```

        ENDIF
    RETURN
    END

C-----
    SUBROUTINE ICND(NDIM,PAR,ICP,NINT,U,UOLD,UDOT,UPOLD,FI,IJAC,DINT)
C  -----
    IMPLICIT DOUBLE PRECISION (A-H,O-Z)
    DIMENSION U(NDIM),UOLD(NDIM),UDOT(NDIM),UPOLD(NDIM)
    DIMENSION FI(NINT),DINT(NINT,*),ICP(*),PAR(*)
    DIMENSION FF(3)
    CALL FUNC(NDIM,U,ICP,PAR,0,FF,DFDU,DFDP)
    FI(1)=SQRT(FF(1)**2 + FF(2)**2 + FF(3)**2 ) - PAR(10)
    RETURN
    END

C-----
    SUBROUTINE PVLS(NDIM,U,PAR)
C  -----
    IMPLICIT DOUBLE PRECISION (A-H,O-Z)
    DIMENSION PAR(*),U(NDIM)
    DIMENSION U1(3),F1(3),WL(3),WR(3)
    U1(1) = GETP("BV1", 1, U)
    U1(2) = GETP("BV1", 2, U)
    U1(3) = GETP("BV1", 3, U)
    CALL FUNC(NDIM,U1,ICP,PAR,0,F1,DFDU,DFDP)
    CALL ScaleV(NDIM,F1)
C At time=1 (at one of the nonzero equilibria):
    CALL EIGV1(NDIM,PAR,WL)
    PAR(18) = F1(1)*WL(1) + F1(2)*WL(2) + F1(3)*WL(3)
C At time=1 (at the other nonzero equilibrium):
    CALL EIGV2(NDIM,PAR,WR)
    PAR(19) = F1(1)*WR(1) + F1(2)*WR(2) + F1(3)*WR(3)
    rho  = PAR(1); beta  = PAR(2); sigma = PAR(3)
    x1 =-SQRT(beta*(rho - 1)); y1 = x1; z1 = rho - 1

```

```

tmp = (U1(1)-x1)**2; tmp = (U1(2)-y1)**2 + tmp
tmp = (U1(3)-z1)**2 + tmp; PAR(25) = SQRT(tmp)
x2 = SQRT(beta*(rho - 1)); y2 = x2; z2 = rho - 1
tmp = (U1(1)-x2)**2; tmp = (U1(2)-y2)**2 + tmp
tmp = (U1(3)-z2)**2 + tmp; PAR(26) = SQRT(tmp)
PAR(6) = U1(1); PAR(27) = U1(2); PAR(28) = U1(3)

RETURN

END

```

C-----

```

SUBROUTINE EIGVO(NDIM,PAR,V1,V2)

```

C -----

```

IMPLICIT DOUBLE PRECISION (A-H,O-Z)

```

```

DIMENSION PAR(*),V1(NDIM),V2(NDIM)

```

```

COMMON /FIRSTO/ ifirst

```

```

rho = PAR(1); beta = PAR(3); sigma= PAR(3)

```

C Stable eigenvalues:

```

d = (sigma+1.d0)**2 + 4.d0*sigma*(rho-1.d0); sq = DSQRT(d)

```

```

ev1 =-beta; ev2 = (-sigma - 1.d0 - sq)/2.d0

```

```

IF(ev1.GE.0d0 .OR. ev2.GE.0d0)THEN

```

```

WRITE(6,*)" Error in user subroutine EIGV"

```

```

STOP

```

```

ENDIF

```

C Normalized stable eigenvector 1

```

V1(1)= 0.; V1(2)= 0.; V1(3)= 1.

```

C Normalized stable eigenvector 2

```

V2(1)=ev2+1.d0; V2(2)=rho; V2(3)= 0.; ss=0.d0

```

```

DO i=1,3

```

```

ss=ss+V2(i)**2

```

```

ENDDO

```

```

ss=DSQRT(ss)

```

```

DO i=1,3

```

```

V2(i)=V2(i)/ss

```

```

ENDDO

```

C Scale the normalized stable eigenvector 2:

```
sc=DABS(EV1)/DABS(EV2)
```

```
DO i=1,3
```

```
V2(i)=sc*V2(i)
```

```
ENDDO
```

C Write the eigenvalues/vectors upon first call:

```
IF(ifirst.NE.1234)THEN
```

```
WRITE(9,101);WRITE(9,102)ev1,V1(1),V1(2),V1(3)
```

```
WRITE(9,102)ev2,V2(1),V2(2),V2(3)
```

```
101 FORMAT(" Stable eigenvalues/vectors at the origin:")
```

```
102 FORMAT(4X,1P4E19.10)
```

```
ifirst=1234
```

```
ENDIF
```

```
RETURN
```

```
END
```

C-----

```
SUBROUTINE EIGV1(NDIM,PAR,V1)
```

C -----

```
IMPLICIT DOUBLE PRECISION (A-H,O-Z)
```

```
DIMENSION PAR(*),V1(NDIM)
```

```
DIMENSION A(3,3),WR(3),WI(3),Z(3,3),FV1(3)
```

```
INTEGER IV1(3)
```

```
COMMON /FIRST1/ ifirst
```

```
rho = PAR(1); beta = PAR(2); sigma= PAR(3)
```

C One of the nonzero stationary points:

```
x1 =-SQRT(beta*(rho - 1)); y1 = x1; z1 = rho - 1
```

C The transpose Jacobian at the nonzero stationary point

```
A(1,1) =-sigma; A(2,1) = sigma; A(3,1) = 0
```

```
A(1,2) = rho - z1; A(2,2) =-1; A(3,2) =-x1
```

```
A(1,3) = y1; A(2,3) = x1; A(3,3) =-beta
```

```
CALL RG(3,3,A,WR,WI,1,Z,IV1,FV1,IER)
```

```
er0 = WR(1); er1 = WR(2); er2 = WR(3)
```

```
ei0 = WI(1); ei1 = WI(2); ei2 = WI(3)
```

```

IF(ei0.EQ.0.d0 .AND. er0.LT.0.d0)THEN
  DO i=1,3
    V1(i) = Z(i,1)
  ENDDO
ELSE
  WRITE(6,*)" Real part complex e.v. in EIGV1 not real, negative"
  WRITE(9,*)" Real part complex e.v. in EIGV1 not real, negative"
  STOP
ENDIF
C Normalize:
  ss=DSQRT(V1(1)**2 + V1(2)**2 + V1(3)**2)
  V1(1)=V1(1)/ss; V1(2)=V1(2)/ss; V1(3)=V1(3)/ss
C Set the orientation
  IF(V1(1).LT.0.)THEN
    DO i=1,3
      V1(i)=-V1(i)
    ENDDO
  ENDIF
C Write the eigenvalue/vector upon first call:
  IF(ifirst.NE.1234)THEN
    WRITE(9,101);WRITE(9,102)er0,ei0;WRITE(9,102)er1,ei1
    WRITE(9,102)er2,ei2;WRITE(9,103);WRITE(9,104)V1(1),V1(2),V1(3)
101  FORMAT(/," Adjoint eigenvalues at nonzero SS1:")
102  FORMAT(4X,1P2E19.10)
103  FORMAT(/," Adjoint stable eigenvector at nonzero SS1:")
104  FORMAT(4X,1P3E19.10)
    ifirst=1234
  ENDIF
RETURN
END
C-----
SUBROUTINE EIGV2(NDIM,PAR,V1)
C -----

```

```

IMPLICIT DOUBLE PRECISION (A-H,O-Z)
DIMENSION PAR(*),V1(NDIM)
DIMENSION A(3,3),WR(3),WI(3),Z(3,3),FV1(3)
INTEGER IV1(3)
COMMON /FIRST2/ ifirst
rho = PAR(1); beta = PAR(2); sigma= PAR(3)
C One of the nonzero stationary points:
x1 = SQRT(beta*(rho - 1)); y1 = x1; z1 = rho - 1
C The transpose Jacobian at the nonzero stationary point
A(1,1) = -sigma; A(2,1) = sigma; A(3,1) = 0
A(1,2) = rho - z1; A(2,2) = -1; A(3,2) = -x1
A(1,3) = y1; A(2,3) = x1; A(3,3) = -beta
CALL RG(3,3,A,WR,WI,1,Z,IV1,FV1,IER)
er0 = WR(1); er1 = WR(2); er2 = WR(3)
ei0 = WI(1); ei1 = WI(2); ei2 = WI(3)
IF(ei0.EQ.0.d0 .AND. er0.LT.0.d0)THEN
DO i=1,3
V1(i) = Z(i,1)
ENDDO
ELSE
WRITE(6,*)" Real part complex e.v. in EIGV2 not real, negative"
WRITE(9,*)" Real part complex e.v. in EIGV2 not real, negative"
STOP
ENDIF
C Normalize:
ss=DSQRT(V1(1)**2 + V1(2)**2 + V1(3)**2)
V1(1)=V1(1)/ss; V1(2)=V1(2)/ss; V1(3)=V1(3)/ss
C Set the orientation
IF(V1(1).LT.0.)THEN
DO i=1,3
V1(i)=-V1(i)
ENDDO
ENDIF

```

C Write the eigenvalue/vector upon first call:

```
      IF(ifirst.NE.1234)THEN
          WRITE(9,101); WRITE(9,102)er0,ei0; WRITE(9,102)er1,ei1
          WRITE(9,102)er2,ei2;WRITE(9,103);WRITE(9,104)V1(1),V1(2),V1(3)
101     FORMAT(/," Adjoint eigenvalues at nonzero SS2:")
102     FORMAT(4X,1P2E19.10)
103     FORMAT(/," Adjoint stable eigenvector at nonzero SS2:")
104     FORMAT(4X,1P3E19.10)
          ifirst=1234
      ENDIF
      RETURN
      END
```

C-----

```
      SUBROUTINE ScaleV(NDIM,V)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION V(NDIM)
```

C Scale the vector V so that its discrete L2-norm becomes 1.

```
      SS=0.d0
      DO i=1,NDIM
          SS=SS+V(i)*V(i)
      ENDDO
      SS=DSQRT(SS)
      DO i=1,NDIM
          V(i)=V(i)/SS
      ENDDO
      RETURN
      END
```

C-----

```
      SUBROUTINE FOPT
      RETURN
      END
```

C-----

A.2 Computing the candidate orbits

The computation is done by calling AUTO script with the command

```
auto start.auto
```

A.2.1 File start.auto

```
import os
def detectMX(solution,logName,files):
#detect "MX", report an error and terminate the program
    if solution["Type name"] == "MX":
        print "\n Abnormal termination at step No.",files," ."
        os._exit(99)
def ifASolutionExist(pattern):
#solutionName: input a string, return if this solution exist
    fn2rm1 = 's.%(file)s' % {'file':pattern}
    fn2rm2 = 'b.%(file)s' % {'file':pattern}
    fn2rm3 = 'd.%(file)s' % {'file':pattern}
    return os.path.exists(fn2rm1) and os.path.exists(fn2rm2) \
and os.path.exists(fn2rm3)
#clean previous computation
files = 1
while ifASolutionExist('man%(d1)04d'%(d1:files)):
    dl('man%(d1)04d'%(d1:files));files = files + 1
print "\n***Generate starting data***"
run(e='man',c='man.1',sv='start')
logName = 'notset';files = 0;solution=sl('start')[-1];
detectMX(solution,logName,files)
while True:
    print "\nFile number:",files," ."
    theta = solution.PAR(5)
    if abs(theta) > 1:
        print "\nEnd, |theta| > 1";break
```

```

files = files + 1;lab=solution["Label"];
run(c='man.2',IRS=lab,s=solution)
@rl
sv('man%(d1)04d'%'d1':files})
solution=sl('man%(d1)04d'%'d1':files})[-1]
detectMX(solution,logName,files)
#candidates computed, select heteroclinic connections
cmds = 'auto run.auto';print cmds;os.system(cmds)
print "\n***Clean the directory***"
cl()

```

A.2.2 Constants file c.man.1

```

3 4 0 0 NDIM,IPS,IRS,ILP
5 11 10 8 9 24 NICP,(ICP(I),I=1,NICP)
200 4 3 0 1 0 6 1 NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
9999 -1e6 1e6 0 1e9 NMX,RLO,RL1,AO,A1
9999 2 3 9 7 3 0 NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC
1e-9 1e-9 1e-7 EPSL,EPSU,EPSS
-0.1 1e-4 10. 1 DS,DSMIN,DSMAX,IADS
3 NTHL,/((I,THL(I)),I=1,NTHL)
10 0.
11 0.
24 0.
0 NTHU,/((I,THU(I)),I=1,NTHU)
1 NUZR,/((I,UZR(I)),I=1,NUZR)
-24 0.

```

A.2.3 Constants file c.man.2

```

3 4 2 0 NDIM,IPS,IRS,ILP
7 5 11 10 8 9 18 19 NICP,(ICP(I),I=1,NICP)
300 4 3 0 1 0 6 1 NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
99999 -1.01 1.01 0 1e9 NMX,RLO,RL1,AO,A1
99999 2 2 9 5 3 0 NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC

```

```

1e-9 1e-9 1e-7  EPSL,EPSU,EPSS
1e-1 1e-4 10.  1    DS,DSMIN,DSMAX,IADS
3  NTHL,/((I,THL(I)),I=1,NTHL)
10 0.
11 0.
24 0.
0  NTHU,/((I,THU(I)),I=1,NTHU)
4  NUZR,/((I,UZR(I)),I=1,NUZR)
8  0.
9  0
-5 1.
-5 -1.

```

A.3 Computing the candidate orbits

After getting the 'man' files, we compute the candidate orbits longer until their distances to the objective equilibrium meet preset limit. Call the script with command:

```
auto run.auto
```

After running the script, the symbol sequence of each orbit in the 'het' files will be saved in file *.seq. The sequence file's name is as the setting in file man.f. For example, 'R1030T14C300.seq' means we set the nominal arclength 1030, the nominal time 14 and the nominal constant 300 in the equation file 'man.f'.

A.3.1 Script run.auto

```

import os
def detectMX(solution,logName,files):
#detect "MX", report an error and terminate the program
    if solution["Type name"] == "MX":
        print "\n Abnormal termination at step No.",files," ."

```

```

        os._exit(99)
def ifASolutionExist(pattern):
#solutionName: input a string, return if this solution exist
    fn2rm1 = 's.%(file)s' % {'file':pattern}
    fn2rm2 = 'b.%(file)s' % {'file':pattern}
    fn2rm3 = 'd.%(file)s' % {'file':pattern}
    return os.path.exists(fn2rm1) and os.path.exists(fn2rm2) \
and os.path.exists(fn2rm3)
Tc=-19.0; epsd=1e-4; epsf=0.5; Dc = 3
if True:
    RLnom = 1030.; Tnom = -14.; Cnom = 300.
het = 'R%(d1)dT%(d2)dC%(d3)d'%(d1':RLnom,'d2':abs(Tnom),'d3':Cnom}
dl(het)
files = 1
while ifASolutionExist('het%(d1)04d'%(d1':files}):
    dl('het%(d1)04d'%(d1':files))
    files = files + 1
files = 1; ld(e='man')
sol_str = 'man%(d1)04d'%(d1':files}
het_str = 'het%(d1)04d'%(d1':files}
f1 = open('%(s1)s.seq'%(s1':het),'w');f1.close()
f3 = open('end8trace.txt','w');f4 = open('end9trace.txt','w')
count = 1
while ifASolutionExist(sol_str):
    count = count + 1; ct8 = 0; ct9 = 0;solutions = sl(sol_str)
for sol in solutions:
    lab=sol["Label"]
    if abs(sol.PAR(8)) < epsd and abs(sol.PAR(25)) < 20 :
print " Restart Label ",lab
print "PAR(8):",sol.PAR(8),"\\tPAR(18):", \
abs(sol.PAR(18)),"\\tPAR(11):",sol.PAR(11),"\\tPAR(25):",sol.PAR(25)
run(c='man.3',IRS=lab,s=sol); sol2 = sl()[-1]
print "PAR(8):",sol2.PAR(8),"\\tPAR(18):", \

```

```

abs(sol2.PAR(18)), "\tPAR(11):", sol2.PAR(11), "\tPAR(25):", sol2.PAR(25)
if sol2.PAR(25) < Dc:
    ct8 = ct8 + 1
    f3.write('%(d1)6d %(s1)20s %(d2)6d\n' \
%{'d1':count-1, 's1':lab, 'd2':ct8})
    ap(het_str); print "***Solution", lab, ": saved***"
    elif abs(sol.PAR(9)) < epsd and abs(sol.PAR(26)) < 20:
print " Restart Label ", lab
print "PAR(9):", sol.PAR(9), "\tPAR(19):", \
abs(sol.PAR(19)), "\tPAR(11):", sol.PAR(11), "\tPAR(26):", sol.PAR(26)
run(c='man.4', IRS=lab, s=sol); sol2 = sl()[-1]
print "PAR(9):", sol2.PAR(9), "\tPAR(19):", \
abs(sol2.PAR(19)), "\tPAR(11):", sol2.PAR(11), "\tPAR(26):", sol2.PAR(26)
if sol2.PAR(26) < Dc:
    ct9 = ct9 + 1
    f4.write('%(d1)6d %(s1)20s %(d2)6d\n' \
%{'d1':count-1, 's1':lab, 'd2':ct9})
    ap(het_str); print "***Solution", lab, ": saved***"
rl(het_str); cmds = '@count %(s1)s' % {'s1':het_str}
    print cmds; os.system(cmds)
    seq_str = '%(s1)s.seq' % {'s1':het_str}; f2 = open(seq_str, 'r')
    lines = f2.readlines(); f2.close()
    f1 = open('%(s1)s.seq' % {'s1':het}, 'a')
    ct = 1
    for line in lines:
        f1.write('%(s1)15s %(d1)6d %(l)s' % {'s1': 's.' + (ss)s' \
%{'ss':het_str}, 'd1':ct, 'l':line})
        ct = ct + 1
    f1.close(); cmds = 'rm %(s1)s.seq' % {'s1':het_str}; print cmds
    os.system(cmds); files = files + 1
    sol_str = 'man%(d1)04d' % {'d1':files}; het_str = 'het%(d1)04d' % {'d1':files}
f3.close(); f4.close()
print "\n***Clean the directory***"; cl()

```

A.3.2 Utility file count.f

```
C=====
C      Utility Program for Counting Sequences for the Lorenz Data
C=====

PARAMETER (NDIMX=3,NCOLX=4,NTSTX=5000,NPARX=50,NSEQX=2048)
PARAMETER(NTPLX=NTSTX*NCOLX+1)
DOUBLE PRECISION UU(NTPLX,NDIMX)
DOUBLE PRECISION UO(NDIMX),U1(NDIMX),RLDOT(NPARX),PAR(NPARX)
INTEGER ICP(NPARX)
INTEGER IBR,NTOT,ITP,LAB,NFPR,ISW,NTPL,NAR,NROWPR,NTST,NCOL,NPAR
CHARACTER*1 SEQ(NSEQX)

OPEN(28,FILE='fort.28',STATUS='old')
1  CONTINUE
READ(28,*,END=99)IBR,NTOT,ITP,LAB,NFPR,ISW,NTPL, &
*      NAR,NROWPR,NTST,NCOL,NPAR
WRITE(6,*)IBR,NTOT,ITP,LAB,NFPR,ISW,NTPL, &
*      NAR,NROWPR,NTST,NCOL,NPAR
WRITE(6,*)NTST,NCOL,NPAR
IF(NTST.GT.NTSTX .OR. NCOL.GT.NCOLX .OR. NPAR.GT.NPARX)THEN
WRITE(6,*)NTST,NCOL,NPAR
WRITE(6,*)NTSTX,NCOLX,NPARX
WRITE(6,*)" Error: Dimension exceeded."
STOP
ENDIF

C Read the solution:
NDIM=NAR-1
DO j=1,NTPL
READ(28,*) T,(UU(j,i),i=1,NDIM)
ENDDO

C Skip UDOTPS:
DO j=1,NTPL+2
READ(28,*)
ENDDO
```

```

C Read parameter values:
      READ(28,*) (PAR(i),i=1,NPAR)
      rho=PAR(1); beta=PAR(2);x1=-SQRT(beta*(rho - 1));
      x2= SQRT(beta*(rho - 1))

C Initialize:
      NDIM=NAR-1
      DO i=1,NDIM
          UO(i)=0.; U1(i)=0.
      ENDDO

      DO i=1,NSEQX
          SEQ(i)=" "
      ENDDO

C Do the counting:
      n=0

      DO j=1,NTPL
          DO i=1,NDIM
              UO(i)=U1(i)
          ENDDO

          DO i=1,NDIM
              U1(I)=UU(j,i)
          ENDDO

          IF(UO(1).GT.x1 .AND. U1(1).LE.x1)THEN
              n=n+1
              SEQ(n)="l"
          ELSEIF(UO(1).LT.x2 .AND. U1(1).GE.x2)THEN
              n=n+1
              SEQ(n)="r"
          ENDIF
      ENDDO

C Simplify the symbol sequences by using "R" and "L":
      IF(SEQ(n).EQ."l")THEN
          DO i=n-1,1,-1
              IF(SEQ(i).EQ."r")THEN

```

```

        n0=i+1
        SEQ(n0)="L"
        GOTO 2
    ENDIF
ENDDO
n0=1
SEQ(1)="L"
ELSEIF(SEQ(n).EQ."r")THEN
DO i=n-1,1,-1
    IF(SEQ(i).EQ."l")THEN
        n0=i+1
        SEQ(n0)="R"
        GOTO 2
    ENDIF
ENDDO
n0=1
SEQ(1)="R"
ENDIF
C Write the symbol sequences:
2   WRITE(6,102) LAB,(SEQ(i),i=1,n0)
    WRITE(11,102)LAB,(SEQ(i),i=1,n0)
    GOTO 1
99  RETURN
102 FORMAT(I6," : ",80A1)
    END
C   -----
    SUBROUTINE SKIP(IUNIT,NSKIP,EOF)
C Skips the specified number of lines on fort.IUNIT.
    LOGICAL EOF
    EOF=.FALSE.
    DO 1 I=1,NSKIP
        READ(IUNIT,*,END=2)
1   CONTINUE

```



```

        RETURN
2     CONTINUE
        EOF=.TRUE.
        RETURN
        END

```

C=====

A.3.3 Constants file c.man.3

```

3 4 316 0 NDIM,IPS,IRS,ILP
5 11 5 10 9 24 NICP,(ICP(I),I=1,NICP)
300 4 3 0 1 0 6 1 NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
99999 -26. 1e9 0 1e9 NMX,RLO,RL1,A0,A1
99999 2 2 9 7 3 0 NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC
1e-6 1e-6 1e-5 EPSL,EPSU,EPSS
-5e-1 1e-4 10. 1 DS,DSMIN,DSMAX,IADS
3 NTHL,/((I,THL(I)),I=1,NTHL)
10 0.
11 0.
24 0.
0 NTHU,/((I,THU(I)),I=1,NTHU)
7 NUZR,/((I,UZR(I)),I=1,NUZR)
-11 -30
-5 1.
-5 -1.
-25 2.5
-26 2.5
-25 100
-26 100

```

A.3.4 Constants file c.man.4

```

3 4 80 0 NDIM,IPS,IRS,ILP
5 11 5 10 8 24 NICP,(ICP(I),I=1,NICP)
300 4 3 0 1 0 6 1 NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT

```

99999 -26. 1e9 0 1e9 NMX,RLO,RL1,A0,A1
99999 2 2 9 7 3 0 NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC
1e-6 1e-6 1e-5 EPSL,EPSU,EPSS
-5e-1 1e-4 10. 1 DS,DSMIN,DSMAX,IADS
3 NTHL,/((I,THL(I)),I=1,NTHL)
10 0.
11 0.
24 0.
0 NTHU,/((I,THU(I)),I=1,NTHU)
7 NUZR,/((I,UZR(I)),I=1,NUZR)
-11 -30
-5 1.
-5 -1.
-25 2.5
-26 2.5
-25 100
-26 100

A.4 Checking if the symbol sequences are complete

First we can run Python script `mlvl.py` to detect the max length of a complete set of symbol sequences.

For example,

```
python mlvl.py R1030T14C300.seq
Length 1 passed!
Length 2 passed!
Length 3 passed!
Length 4 passed!
Length 5 passed!
Length 6 passed!
Length 7 passed!
Length 8 passed!
Length 9 passed!
Length 10 NOT passed!
```

A.4.1 Python script `mlvl.py` (max level)

```
import os,sys,shutil
if len(sys.argv) < 1:
    print "Usage python gs.py seq_filename!";os._exit(99)
seq_filename = sys.argv[1]
if not os.path.exists(seq_filename):
    print seq_filename," does not exist!"; os._exit(99)
f1 = open(seq_filename,'r'); lines = f1.readlines(); f1.close()
level = 1
while True:
    str_list = []; hit_list = []; l = level
```

```

for i in range(1,2**l+1,1):
    r = i%2
    if r==1:
        s='L'
    else:
        s='R'
    d = (i-1)/2; l2 = l - 1
    while l2>= 1:
        r = d%2
        if r == 0:
            s = '%(s1)s%(s2)s'%(s1:'l',s2':s}'
        else:
            s = '%(s1)s%(s2)s'%(s1:'r',s2':s}'
        d = (d)/2; l2 = l2 - 1
    se = s[level-1:level].lower(); s2 = s[level-1:level]
    nb = True
    for j in range(level-2,-1,-1):
        if nb:
            if not s[j:j+1] == se:
                s2 = '%(s1)s%(s2)s'%(s1':s[j:j+1],s2':s2}'
                nb = False
        else:
            s2 = '%(s1)s%(s2)s'%(s1':s[j:j+1],s2':s2}'
    str_list.append(s2); hit_list.append(0)
i1 = 0; i2 = 0; i3 = 0
for line in lines:
    i2 = i1; i3 = i3 + 1; s2 = line.split()[3]
    if len(s2) > level:
        continue
    found = False
    while True:
        if str_list[i2] == s2:
            hit_list[i2] = hit_list[i2]+1; found = True

```

```

        break
    i2 = i2+1
    if i2 >= len (str_list):
        break
    if found:
        i1 = i2
        if i2 >= len (str_list):
            break
found = True; i = 0; mct = 0
for hit in hit_list:
    if hit == 0:
        found = False
        mct = mct+1
    i = i+1
if found:
    print "Length ",level," passed!"; level = level + 1
else:
    print "Length ",level," NOT passed!"; break

```

A.4.2 Record the sequences

If the set with desired length is not complete, we can use AUTO script `trace.auto` to detect the sequences that are missing. It will create three log files. For examples, `R1030T14C300.trace1` gives the complete set of symbol sequences meeting a given length with both compact form and verbose form. `R1030T14C300.trace2` lists the computed sequences and missing sequences of a given length. `R1030T14C300.trace3` records a sequence's origin solution 'het' file and the label in the 'het' file. The script can be called by command line.

```

auto trace.auto sequencefilename sequence_length

```

A.4.3 AUTO script `trace.auto`

```

import os,sys,shutil
def ifASolutionExist(pattern):

```

```

#solutionName: input a string, return if this solution exist
    fn2rm1 = 's.%(file)s' % {'file':pattern}
    fn2rm2 = 'b.%(file)s' % {'file':pattern}
    fn2rm3 = 'd.%(file)s' % {'file':pattern}
    return os.path.exists(fn2rm1) and os.path.exists(fn2rm2) \
and os.path.exists(fn2rm3)
if len(sys.argv) < 3:
    print "Usage: auto trace.auto sequencefilename sequence_length"; os._exit(99)
solutionName = sys.argv[2]; seq_filename = '%(s1)s.seq'%(s1':solutionName}
if not os.path.exists(seq_filename):
    print seq_filename," does not exist!"; os._exit(99)
level = int(sys.argv[3])
if level < 1:
    print "Level must be greater than 0!"; os._exit(99)
print "Maximum sequence length ",level
str_list = []; fullstr_list = []; hit_list = []; l = level
f2 = open('%(s1)s.trace1'%(s1':solutionName},'w')
for i in range(1,2**l+1,1):
    r = i%2
    if r==1:
        s='L'
    else:
        s='R'
    d = (i-1)/2; l2 = l - 1
    while l2>= 1:
        r = d%2
        if r == 0:
            s = '%(s1)s%(s2)s'%(s1':l',s2':s}
        else:
            s = '%(s1)s%(s2)s'%(s1':r',s2':s}
        d = (d)/2; l2 = l2 - 1
#zip
se = s[level-1:level].lower(); s2 = s[level-1:level]; nb = True

```

```

for j in range(level-2,-1,-1):
    if nb:
        if not s[j:j+1] == se:
            s2 = '%(s1)s%(s2)s'%(s1':s[j:j+1], 's2':s2}
            nb = False
        else:
            s2 = '%(s1)s%(s2)s'%(s1':s[j:j+1], 's2':s2}
    str_list.append(s2); fullstr_list.append(s); hit_list.append(0)
    f2.write('%(d1)5d | %(s1)15s | %(s2)15s\n'%(d1':i, 's1':s, 's2':s2))
f2.close(); f1 = open(seq_filename, 'r'); lines = f1.readlines()
i1 = 0; i2 = 0; i3 = 0; print "file length ", len(lines)
f4 = open('%(s1)s.trace2'%(s1':solutionName}, 'w')
f5 = open('%(s1)s.trace3'%(s1':solutionName}, 'w')
for line in lines:
    i2 = i1; i3 = i3 + 1; s2 = line.split()[3]
    f4.write('%(d1)5d | %(s1)25s'%(d1':i3, 's1':s2))
    if len(s2) > level:
        f4.write('\n'); continue
    found = False
    while True:
        if str_list[i2] == s2:
            hit_list[i2] = hit_list[i2]+1; found = True
            f4.write(' | %(d1)5d | %(s1)25s | %(d2)5d*'%(d1' \
:i2+1, 's1':str_list[i2], 'd2':hit_list[i2]))
            f5.write('%(d1)6d %(s1)20s %(s2)10s %(s3)15s %(s4)15s \n' \
'%(d1':i2+1, 's1':line.split()[0], 's2':line.split()[1], 's3': \
line.split()[3], 's4':fullstr_list[i2]))
            break
        i2 = i2+1
        if i2 >= len (str_list):
            break
    if found:
        i1 = i2

```

```

        if i2 >= len (str_list):
            break
        f4.write('\n')
found = True; i = 0; mct = 0
print "\nComputed sequences for sequence length ",level
f4.write("\nComputed sequences for sequence length %(d1)d .\n" \
%{'d1':level})
print "%(s0)20s %(s1)15s %(s2)15s"%(s0: 'sequence', \
's1: 'Time PAR(10)', 's2: 'Arc PAR(11)')
f4.write("%(s0)20s %(s1)15s %(s2)15s"%(s0: 'sequence', \
's1: 'Time PAR(10)', 's2: 'Arc PAR(11)'))
for hit in hit_list:
    if hit > 0:
        found = False; mct = mct+1; print str_list[i]
        str = "%(s0)20s\n"%(s0: str_list[i]); f4.write(str)
        i = i+1
if found:
    print "There are no Computed sequences!"
    f4.write("There are no Computedsequences!")
else:
    print "Total ", mct, " Computed sequences."
    f4.write("Total %(d1)d Computed sequences."%{'d1':mct})
found = True; i = 0; mct = 0
print "\n\nMissing sequences for sequence length ",level
f4.write("\nMissing sequences for sequence length %(d1)d .\n" \
%{'d1':level})
for hit in hit_list:
    if hit == 0:
        found = False; mct = mct+1; print str_list[i]
        f4.write(str_list[i]); f4.write('\n')
        i = i+1
if found:
    print "There are no missing sequences!";

```



```

    f4.write("There are no missing sequences!")
else:
    print "Total ", mct, " missing sequences.";
    f4.write("Total %(d1)d missing sequences."%{'d1':mct})
f4.close(); f5.close();f1.close(); os._exit(99)

```

A.5 Supplement missing sequences manually

Sometimes, for a given sequence length, there are only few sequences are missing. We can calculate them by decreasing the continuation step size or increasing the accuracy. For example, after running all the above scripts, we find for length ten there is only one sequence *llrrrrrL* is missing. By checking file R1030T14C300.trace2, we find its two adjacent sequences *llrrrrrL* and *llrrrrrL* are all in s.het0009. So we can seek in file s.man0009 the original solutions. We select a label near *llrrrrrL* and continue for some steps with smaller 'DS' and higher accuracy. We will get some additional candidates and we can find a desired sequence among them.

Then we use '@count man0009+' to compute more candidates. Again we continue the

```

©R man 21 man0009
©sv man0009+
©rl man0009+

```

orbits with longer time to test if a candidate is a connection by '@R man 3 man0009+'. Save the connections by '@sv het0009+'. We may use QTPlaut or PLAUT04 to help find the label of the missing sequence, since we have only one orbit missing. We then add line

```

126          s.het0009+          1          llrrrrrL          llrrrrrL

```

to file R1030T14C300.trace3. The label in het0009+ may vary.

A.5.1 Constants file c.man.21

```

3 4 234 0 NDIM,IPS,IRS,ILP
7 5 11 10 8 9 18 19  NICP,(ICP(I),I=1,NICP)
300 4 3 0 1 0 6 1  NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT

```

```

20000 -1.01 1.01 0 1e9 NMX,RLO,RL1,AO,A1
99999 2 2 9 5 3 0 NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC
1e-9 1e-9 1e-7 EPSL,EPSU,EPSS
1e-2 1e-5 10. 1 DS,DSMIN,DSMAX,IADS
3 NTHL,/((I,THL(I)),I=1,NTHL)
10 0.
11 0.
24 0.
0 NTHU,/((I,THU(I)),I=1,NTHU)
4 NUZR,/((I,UZR(I)),I=1,NUZR)
8 0.
9 0
-5 1.
-5 -1.

```

A.6 Extract the connections to one solution

All the connections are saved in the 'het' files. The file names and labels are recorded in file R1030T14C300.trace3. We can extract all the connections to a single solution by AUTO script extract.auto seqfile should be seqfile.trace3.

```
auto extract.auto seqfile svname
```

A.6.1 AUTO script extract.auto

```

import os,sys,shutil
def ifASolutionExist(pattern):
#solutionName: input a string, return if this solution exist
    fn2rm1 = 's.%(file)s' % {'file':pattern}
    fn2rm2 = 'b.%(file)s' % {'file':pattern}
    fn2rm3 = 'd.%(file)s' % {'file':pattern}
    return os.path.exists(fn2rm1) and os.path.exists(fn2rm2) \
and os.path.exists(fn2rm3)
if len(sys.argv) < 4:

```

```

    print "Usage: auto extract.auto seqfile svname"; os._exit(99)
seqName = sys.argv[2]; seqName = '%(s1)s.trace3'%(s1':sys.argv[2]}
if not os.path.exists(seqName):
    print seqName," does not exist!"; os._exit(99)
f1 = open(seqName,'r'); seqs = f1.readlines(); f1.close();
svname = sys.argv[3]; soln = 'notInited';
for seq in seqs:
    sgs = seq.split(); soln2 = sgs[1].split('.'); lab = int(sgs[2])-1
    soln3 = soln2[1]
    if not soln ==soln3:
        soln = soln3; print soln; sols = sl(soln)
    sol= sols[lab]
    if abs(sol.PAR(8)) < 1e-3:
        run(e='man',c='man.32',s=sol)
    if abs(sol.PAR(9)) < 1e-3:
        run(e='man',c='man.42',s=sol)
    ap(svname)
rl(svname); cmds = '@count %(s1)s' % {'s1':svname}; print cmds
os.system(cmds); print "\n***Clean the directory***"; cl(); os._exit(99)

```

A.6.2 Constants file c.man.32

```

3 4 316 0 NDIM,IPS,IRS,ILP
5 11 5 10 9 24 NICP,(ICP(I),I=1,NICP)
300 4 3 0 1 0 6 1 NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
99999 -26. 1e9 0 1e9 NMX,RL0,RL1,A0,A1
99999 2 2 9 7 3 0 NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC
1e-6 1e-6 1e-5 EPSL,EPSU,EPSS
-1e-1 1e-6 10. 1 DS,DSMIN,DSMAX,IADS
3 NTHL,/((I,THL(I)),I=1,NTHL)
10 0.
11 0.
24 0.
0 NTHU,/((I,THU(I)),I=1,NTHU)

```

```

7  NUZR,/((I,UZR(I)),I=1,NUZR)
-11 -100
-5 1.
-5 -1.
-25 2.
-26 2.
-25 100
-26 100

```

A.6.3 Constants file c.man.42

```

3 4 80 0 NDIM,IPS,IRS,ILP
5 11 5 10 8 24 NICP,(ICP(I),I=1,NICP)
300 4 3 0 1 0 6 1 NTST,NCOL,IAD,ISP,ISW,IPLT,NBC,NINT
99999 -26. 1e9 0 1e9 NMX,RL0,RL1,A0,A1
99999 2 2 9 7 3 0 NPR,MXBF,IID,ITMX,ITNW,NWTN,JAC
1e-6 1e-6 1e-5 EPSL,EPSU,EPSS
-1e-1 1e-6 10. 1 DS,DSMIN,DSMAX,IADS
3 NTHL,/((I,THL(I)),I=1,NTHL)
10 0.
11 0.
24 0.
0 NTHU,/((I,THU(I)),I=1,NTHU)
7 NUZR,/((I,UZR(I)),I=1,NUZR)
-11 -100
-5 1.
-5 -1.
-25 2
-26 2
-25 100
-26 100

```

Appendix B

Computing CR3BP Unstable Manifold

In this chapter, we give the technical details of computing the CR3BP unstable manifold. This chapter is an extension of AUTO demo 'r3b'.

B.1 The general procedure

First, we have to use demo 'r3b' to compute the different periodic solution families. For a given family, for example the family L_2 , we can call following line command to get the 1D unstable labels listed in file 'L2-compute.log'.

```
auto work.auto 0 L2
```

Then for a given label xx , we call the following command line .

```
auto work.auto 1 L2 xx eps
```

This run will compute a starting orbit of the 1D unstable manifold. 'eps' is the ϵ . It can also perform limited correction if $PAR(5) < 0$ when computing the adjoint eigenfunction. We span the unstable manifold (the continuation of ϵ).

```
auto work.auto 2 L2 xx eps
```

After this step, we will get a series of AUTO solutions spanning different solutions in the 'start' solution. Then we use QTPlaut to browse these files. QTPlaut has a function that will record a file called 'BK.LIST'. For example, a line written to 'BK.LIST'

```
s.uL2_029_+1e-5_0052 17 L2-HOMO
```

means in solution file 's.uL2_029_+1e-5_0052' (L2 family label 29, $\epsilon = +1e - 5$, starting label 52) has a homoclinic connection saved as label 17. 'L2-HOMO' is a comment added by the user.

```
auto work.auto 4 L2 xx eps
```

will backup the results to a separate folder according to 'BK.LIST'. At the same time, each solution's key parameters are put to a log.

```
auto work.auto 5 L2 xx eps
```

will clean the computing result in the current folder.

B.2 Scripts

B.2.1 AUTO script work.auto

```
import os,sys,shutil
def bkSingleSolution (uSolution,all_flg,special_flg, \
familyLog_fds,bk_lines,fn,fm,stbl,mult,eng,fullBkFolder):
    if ifASolutionExist(uSolution):
        print uSolution,'exists ... '
```

```

print 'Performing data analysis ',uSolution,' ...'
u_branches = sl(uSolution)
u_maxlabel = int(u_branches[-1]["Label"])
u_i = 1
while True:
    if u_i > u_maxlabel:
        break
    # write a line to logs
    # if this label u_i is in the BK list
    j = 0; found = False; rmk = ' '; eng2 = 0
    dist = 0; ptime = 0; arcl = 0
    for bline in bk_lines:
        h = bline.split()
        if fn == h[0][2:] and u_i == int( h[1]):
            print u_i,',',int( h[1])
            found = True
            if len(h) >=3 :
                c=0
                for th in h:
                    if c>= 2:
                        rmk = th + ' '
                    c=c+1
            break
        j=j+1
    eng2 = u_branches[u_i-1]["Parameters"][2]
    dist = u_branches[u_i-1]["Parameters"][5]
    ptime = u_branches[u_i-1]["Parameters"][10]
    arcl = u_branches[u_i-1]["Parameters"][11]
    lb = u_branches[u_i-1]["Label"]
    line2w = '%(fm)6s %(fn)25s %(lb)5s %(stbl)6s\
%(mult) 13s %(eng)13e %(dist)13e %(time)13e %(arcl)13e\
%(eng2)13e %(rmk)6s\n' %{'fm':fm,'fn':fn,'lb':lb,'stbl':stbl,\
'mult':mult,'eng':float(eng),'eng2':float(eng2),'dist':float\

```

```

(dist),'time':float(pTime),'arcl':float(arcl),'rmk':rmk}

    if found:
        #write to all logs
        all_flg.write(line2w)
        #write to special logs
        special_flg.write(line2w)
        familyLog_fds.write(line2w)
        #back up this file
        print 'back up files to folder ',fullBkFolder,uSolution
        copyPathSolution (uSolution,fullBkFolder,uSolution)
    else:
        #write to all logs
        all_flg.write(line2w)
    u_i = u_i+1
def generateLogs(solutionName):
    cmds = '@fl %(s1)s > flqtmp.txt' % {'s1':family}
    print cmds; os.system(cmds)
    branches = sl(solutionName)
    maxlabel = int(branches[-1]["Label"]); print maxlabel
    f1 = open('flqtmp.txt','r')
    f2 = open('%(s1)s-compute.log'% {'s1':family},'w')
    f3 = open('%(s1)s-brief.log'% {'s1':family},'w')
    flqlines = f1.readlines()
    i = 1
    str = '%(s0)8s %(s1)5s %(s2)18s %(s3)18s %(s4)9s\tRemarks\t\n'%\
{'s0':'Computed','s1':'Label','s2':'Energy','s3':'Multiplier',\
's4':'Stability'}
    f2.write(str); f3.write(str)
    while True:
        if i >= maxlabel:
            break
        abspart= [' ',' ',' ',' ',' ',' ',' ',' ']
        fabspart= [' ',' ',' ',' ',' ',' ',' ',' ']

```



```

hh = flqlines[(i-1)*7+1].split()
abspart[0]= hh[8]; fabspart[0]= float(hh[8])
hh = flqlines[(i-1)*7+2].split()
abspart[1]= hh[8]; fabspart[1]= float(hh[8])
hh = flqlines[(i-1)*7+3].split()
abspart[2]= hh[8]; fabspart[2]= float(hh[8])
hh = flqlines[(i-1)*7+4].split()
abspart[3]= hh[8]; fabspart[3]= float(hh[8])
hh = flqlines[(i-1)*7+5].split()
abspart[4]= hh[8]; fabspart[4]= float(hh[8])
hh = flqlines[(i-1)*7+6].split()
abspart[5]= hh[8]; fabspart[5]= float(hh[8])
j = 0; dtmp = fabspart[0]; flct = 0
for pt in fabspart:
    if pt > dtmp:
        flct = j; dtmp = pt
    j= j+1
    str = '%(s0)8s %(s1)5s %(s2)18s %(s3)18s %(s4)9s\t\t\n' \
%{'s0':'0', 's1':branches[i]["Label"], 's2':branches[i]["Parameters"]\
[2], 's3':abspart[flct], 's4':flqlines[(i-1)*7].split()[4]}
    if int (flqlines[(i-1)*7].split()[4]) == 5:
        f2.write(str)
        f3.write(str)
        i = i+1
f1.close(); f2.close(); f3.close()
cmds = 'rm flqtmp.txt'; print cmds; os.system(cmds)
def chTime(fileName,newTime):
    print "new time %(d1)f"%{'d1':newTime}
    f = open(fileName,'r'); lines = f.readlines();s f.close()
    h2 = "-11 %(d1)f \n"%{'d1':newTime}
    f = open(fileName,'w'); j = 0 ; NUZRat = -1
    for line in lines:
        i = line.find("-11"); found = line.find("NUZR")

```

```

    if found >= 0:
        NUZRat = j
    print i
    if i >= 0 and j== NUZRat+1:
        f.write(h2); print h2
    else:
        f.write(line); print line
    j = j+1
f.close()
def chIRS(fileName,newIRS):
    f = open(fileName,'r'); lines = f.readlines(); f.close()
    h = lines[0].split()
    h2 = '%(s1)s %(s2)s %(s3)s %(s4)s NDIM,IPS,IRS,ILP\n\'\'
%{'s1':h[0], 's2':h[1], 's3':newIRS, 's4':h[3]}
    f = open(fileName,'w'); i = 0
    for line in lines:
        if i >0:
            f.write(line)
        else:
            f.write(h2)
        i=i+1
    f.close()
def ifASolutionExist(pattern):
    #solutionName: input a string, return if this solution exist
    fn2rm1 = 's.%(file)s' % {'file':pattern}
    fn2rm2 = 'b.%(file)s' % {'file':pattern}
    fn2rm3 = 'd.%(file)s' % {'file':pattern}
    return os.path.exists(fn2rm1) and os.path.exists(fn2rm2) and \
os.path.exists(fn2rm3)
def copyPathSolution (src,dstpath,dest):
    if ifASolutionExist(src):
        shutil.copy('s.%(src)s'%'{'src':src}, '%(dstpath)s/s.%(dest)s'\'
%{'dstpath':dstpath, 'dest':dest})

```

```

        shutil.copy('b.%(src)s'%'src':src},'%(dstpath)s/b.%(dest)s'\
%{'dstpath':dstpath,'dest':dest})

        shutil.copy('d.%(src)s'%'src':src},'%(dstpath)s/d.%(dest)s'\
%{'dstpath':dstpath,'dest':dest})

def removeFile(fn2rm):
    if os.path.exists(fn2rm) :
        os.remove(fn2rm)

def removeASolution(pattern):
    fn2rm = 's.%(file)s' % {'file':pattern}; removeFile(fn2rm)
    fn2rm = 'b.%(file)s' % {'file':pattern}; removeFile(fn2rm)
    fn2rm = 'd.%(file)s' % {'file':pattern}; removeFile(fn2rm)

def removeATmpSolution(pattern):
    fn2rm = 's.%(file)s~' % {'file':pattern}; removeFile(fn2rm)
    fn2rm = 'b.%(file)s~' % {'file':pattern}; removeFile(fn2rm)
    fn2rm = 'd.%(file)s~' % {'file':pattern}; removeFile(fn2rm)

control = int(sys.argv[2])

if control == 0:
    family = sys.argv[3]
    if not ifASolutionExist(family):
        print "run r3b.auto to compute ",family,"!"; os._exit(99)
    if not os.path.exists( '%(s1)s-brief.log'%'s1':family)) \
or control == 0:
        generateLogs(family)
        print "BRIEF"; cmds = 'cat %(s1)s-brief.log'%'s1':family}
        print cmds; os.system(cmds)
        print "TO COMPUTE"
        cmds = 'cat %(s1)s-compute.log'%'s1':family}
        print cmds; os.system(cmds)

if control == 1:
    family = sys.argv[3]; startlbl = sys.argv[4]; eps = sys.argv[5]

```

```

    cmds = 'autox ext.py %(s1)s %(s2)s %(s3)s' % \
{'s1':family,'s2':startlbl,'s3':eps}
    print cmds; os.system(cmds)
    print '@r flq ext'; os.system('@r flq ext')
    print '@sv flq'; os.system('@sv flq')
    removeATmpSolution('flq'); solution=sl('flq')[-1]
    p = solution["Parameters"]; print "\nPAR(5):",p[4],"\n"
    if p[4] <= 0.0:
        print "@r flq ext PAR(5)<0"; os._exit(99)
    else:
        os.system('autox data.py')
    cmds = 'cp c.man.%(s1)s.0 c.man.%(s1)s.10' % {'s1':family}
    print cmds; os.system(cmds)
    cmds = '@R man %(s1)s.10' % {'s1':family}; print cmds
    os.system(cmds); solution=sl()[-1]
    if solution["Type name"] == "MX":
        print " restart step 0"; branches = sl()
        maxlabel = int(branches[-1]["Label"])
        print branches[maxlabel-2]["Parameters"][10]
        chTime('c.man.%(s1)s.10' % {'s1':family},\
float(branches[maxlabel-2]["Parameters"][10])*0.99)
    cmds = '@R man %(s1)s.10' % {'s1':family}; print cmds
    os.system(cmds)
    solution=sl()[-1]
    if solution["Type name"] == "MX":
        print "\n Abnormal termination at ",cmds; os._exit(99)
    cmds = '@sv start%(s1)s_%(s2)03d_%(s3)s' % \
{'s1':family,'s2':int(startlbl),'s3':eps}
    print cmds; os.system(cmds)
    removeATmpSolution('start%(s1)s_%(s2)03d_%(s3)s' % \
{'s1':family,'s2':int(startlbl),'s3':eps})

#if flq par(5) <=0

```

```

if control == 10:
    family = sys.argv[3];startlbl = sys.argv[4]; eps = sys.argv[5]
    if len(sys.argv) == 7 :
        flq = sys.argv[6]
        cmds = 'autox ext.py %(s1)s %(s2)s %(s3)s %(s4)s' %\
{'s1':family,'s2':startlbl,'s3':eps,'s4':flq}
    else:
        cmds = 'autox ext.py %(s1)s %(s2)s %(s3)s 50' %\
{'s1':family,'s2':startlbl,'s3':eps}

    print cmds; os.system(cmds)
    print '@R flq 2 ext'; os.system('@R flq 2 ext')
    print '@sv flq'; os.system('@sv flq'); removeATmpSolution('flq')
    print '@R flq 3'; os.system('@R flq 3')
    print '@sv flq'; os.system('@sv flq'); removeATmpSolution('flq')
    solution=sl('flq')[-1]; p = solution["Parameters"]
    print "\nPAR(5):",p[4],"\n"
    if p[4] <= 0.0:
        print "@r flq ext PAR(5)<0"; os._exit(99)
    else:
        print 'autox data.py flq 4'; os.system('autox data.py flq 4')
        cmds = 'cp c.man.%(s1)s.0 c.man.%(s1)s.10' % {'s1':family}
        print cmds; os.system(cmds)
        cmds = '@R man %(s1)s.10' % {'s1':family}; print cmds
        os.system(cmds); solution=sl()[-1]
        if solution["Type name"] == "MX":
            print " restart step 0"; branches = sl()
            maxlabel = int(branches[-1]["Label"])
            print branches[maxlabel-2]["Parameters"][10]
            chTime('c.man.%(s1)s.10' % {'s1':family},\
branches[maxlabel-2]["Parameters"][10])

        cmds = '@R man %(s1)s.10' % {'s1':family}; print cmds
        os.system(cmds); solution=sl()[-1]
        if solution["Type name"] == "MX":

```

```

        print "\n Abnormal termination at ",cmds; os._exit(99)
    cmds = '@sv start%(s1)s_%(s2)03d_%(s3)s' % {'s1':family,\
's2':int(startlbl),'s3':eps}
    print cmds; os.system(cmds)
    removeATmpSolution('start%(s1)s_%(s2)03d_%(s3)s' %\
{'s1':family,'s2':int(startlbl),'s3':eps})

#span unstable manifold
if control == 2:
    family = sys.argv[3]; startlbl = sys.argv[4]; eps = sys.argv[5]
    startSolution = 'start%(s1)s_%(s2)03d_%(s3)s' % \
{'s1':family,'s2':int(startlbl),'s3':eps}
    if not ifASolutionExist(startSolution):
        print "solution ",startSolution," does not exist!";os._exit(99)
    instStartSolution = sl(startSolution)[-1]
    maxlabel = int(instStartSolution["Label"])
    cmds = 'cp c.man.%(s1)s.1 c.man.%(s1)s.11' % {'s1':family}
    print cmds; os.system(cmds)
    if len(sys.argv) == 6:
        if len(sys.argv) == 7:
            i = int (sys.argv[6])
        else:
            i = 2
    while True:
        if i > maxlabel:
            break
        str_rstlabel = '%(d1).d'%'{'d1':i}
        print "restart label ",str_rstlabel
        chIRS('c.man.%(s1)s.11'%'{'s1':family},str_rstlabel)
        cmds = '@R man %(s1)s.11 %(s2)s' %\
{'s1':family,'s2':startSolution}
        print cmds; os.system(cmds); solution=sl()[-1]
        if solution["Type name"] == "MX":

```

```

        print "\n Abnormal termination at ",cmds
        os._exit(99)
        cmds = '@sv u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d' %\
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':i}
        print cmds; os.system(cmds)
        removeATmpSolution('u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d' %\
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':i})
        r1('u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d' %\
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':i})
        i=i+1

#unstable
if control == 3:
    family = sys.argv[3]; startlbl = sys.argv[4]
    eps = sys.argv[5]; fileNo = int(sys.argv[6] )
    str_rstlabel= sys.argv[7]
    startSolution = 'u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':fileNo}
    if not ifASolutionExist(startSolution):
        print "solution ",startSolution," does not exist!";os._exit(99)
    chIRS('c.man.%(s1)s.3'%'{'s1':family},str_rstlabel)
    print "extend longer orbit"
    cmds = '@R man %(s1)s.3 %(s2)s' % {'s1':family,'s2':startSolution}
    print cmds; os.system(cmds); solution=s1()[-1]
    if solution["Type name"] == "MX":
        print " restart step 0"
        branches = s1(); maxlabel = int(branches[-1]["Label"])
        if maxlabel <= 2:
            print "Not enough branches to restart"; os._exit(99)
        print branches[maxlabel-2]["Parameters"][10]
        chTime('c.man.%(s1)s.8' % {'s1':family},\
branches[maxlabel-2]["Parameters"][10])
        cmds = '@R man %(s1)s.8' % {'s1':family}; print cmds

```

```

os.system(cmds); solution=sl()[-1]
if solution["Type name"] == "MX":
    print "\n Abnormal termination at ",cmds; os._exit(99)
cmds = '@sv u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d_rst' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':fileNo}
print cmds; os.system(cmds)
branches = sl(); maxlabel = int(branches[-1]["Label"])
start2File = 'u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d_rst' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':fileNo}
chIRS('c.man.%(s1)s.4'%(s1):family,maxlabel); print "Sweep DS+"
cmds = '@R man %(s1)s.4 %(s2)s' % {'s1':family,'s2':start2File}
print cmds; os.system(cmds)
if solution["Type name"] == "MX":
    print "\n Abnormal termination at ",cmds; os._exit(99)
cmds = '@sv u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d_DS+' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':fileNo}
print cmds; os.system(cmds)
chIRS('c.man.%(s1)s.5'%(s1):family,maxlabel); print "Sweep DS-"
cmds = '@R man %(s1)s.5 %(s2)s' % {'s1':family,'s2':start2File}
print cmds; os.system(cmds)
if solution["Type name"] == "MX":
    print "\n Abnormal termination at ",cmds; os._exit(99)
cmds = '@sv u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d_DS-' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':fileNo}
print cmds; os.system(cmds)

#bk by BK.list
if control == 4:
    family = sys.argv[3]; startlbl = sys.argv[4]; eps = sys.argv[5]
    startSolution = 'start%(s1)s_%(s2)03d_%(s3)s' % \
{'s1':family,'s2':int(startlbl),'s3':eps}
    if not ifASolutionExist(startSolution):
        print "solution ",startSolution," does not exist!";os._exit(99)

```



```

bkFolder = '%(s1)s_%(s2)03d_%(s3)s' % {'s1':family, \
's2':int(startlbl), 's3':eps}
fullBkFolder = "%(hm)s/%(fd)s" % {'hm':os.getcwd(), 'fd':bkFolder}
if ( not os.path.exists(fullBkFolder) ):
    os.mkdir(fullBkFolder)
#Backup constant file
cmds = 'cp c.man.%(s1)s.* %(s2)s' % {'s1':family, 's2':fullBkFolder}
print cmds; os.system(cmds)
cmds = '@fl %(s1)s > flqtmp.txt' % {'s1':family}
print cmds; os.system(cmds)
cmds = 'cp flqtmp.txt %(s2)s/%(s1)s_flq.txt' % \
{'s1':family, 's2':fullBkFolder}
print cmds; os.system(cmds)
logFile_all = '%(s1)s_alllogs.txt'%(s1':startSolution}
all_flg = open(logFile_all, 'w')
line = '%(fm)6s|%(fn)-25s|%(lb)5s|%(stbl)6s|%(mult)13s|%(eng)13s|\
%(dist)13s|%(time)13s|%(arcl)13s|%(eng2)13s|%(rmk)6s\n' % \ {'fm':'Family', 'fn':\
' File Name', 'lb':'LAB', 'stbl':'Stable', 'mult':'Multiplier', 'eng':'Energy', \
'eng2':'End Eng', 'dist':'Distance', 'time':'End time', 'arcl':'Arc-length', 'rmk':'Remarks'}
all_flg.write(line)
line = '%(fm)6s|%(fn)25s|%(lb)5s|%(stbl)6s|%(mult)13s|%(eng)13s|\
%(dist)13s|%(time)13s|%(arcl)13s|%(eng2)13s|%(rmk)6s\n' % \
{'fm':'-', 'fn':'-', 'lb':'-', 'stbl':'-', 'mult':'-', 'eng':\
'PAR(3)', 'eng2':'PAR(3) 2', 'dist':'PAR(6)', 'time':'PAR(11)', \
'arcl':'PAR(12)', 'rmk':''}
all_flg.write(line)
logFile_special = '%(s1)s_speciallogs.txt'%(s1':startSolution}
special_flg = open(logFile_special, 'w')
line1 = '%(fm)6s|%(fn)-25s|%(lb)5s|%(stbl)6s|%(mult)13s|\
%(eng)13s|%(dist)13s|%(time)13s|%(arcl)13s|%(eng2)13s|\
%(rmk)6s\n' % {'fm':'Family', 'fn':' File Name', \
'lb':'LAB', 'stbl':'Stable', 'mult':'Multiplier', 'eng':'Energy', \
'eng2':'End Eng', 'dist':'Distance', 'time':'End time', \

```

```

'arcl': 'Arc-length', 'rmk': 'Remarks'}

    special_flg.write(line1)

    line2 = '%(fm)6s|(fn)25s|(lb)5s|(stbl)6s|(mult)13s|%(
(eng)13s|(dist)13s|(time)13s|(arcl)13s|(eng2)13s|(rmk)6s\n'\
%{'fm': '- ', 'fn': '- ', 'lb': '- ', 'stbl': '- ', 'mult': '- ', 'eng': '\
'PAR(3)', 'eng2': 'PAR(3) 2', 'dist': 'PAR(6)', 'time': 'PAR(11)', \
'arcl': 'PAR(12)', 'rmk': ''}

    special_flg.write(line2)

    familyLog = '%(fm)s_speciallog.txt' % {'fm': family}

    if not os.path.exists(familyLog):

        familyLog_fds = open(familyLog, 'w')

        familyLog_fds.write(line1); familyLog_fds.write(line2)

    else:

        familyLog_fds = open(familyLog, 'a')

    brief_info_file = '%(s1)s-brief.log' % {'s1': family}

    brief_info = open(brief_info_file, 'r')

    brief_lines = brief_info.readlines()

    bk_file = 'BK.LIST'; bk_fds = open(bk_file, 'r')

    bk_lines = bk_fds.readlines()

    instStartSolution = sl(startSolution)[-1]

    maxlabel = int(instStartSolution["Label"])

    fm = family

    stbl = brief_lines[int(startlbl)-1].split()[4]

    mult = brief_lines[int(startlbl)-1].split()[3]

    eng = brief_lines[int(startlbl)-1].split()[2]

    i = 1; print 'Performing data analysis ', ' ...'

    while True:

        if i > maxlabel:

            break

        str_rstlabel = '%(d1).d' % {'d1': i}

        uSolution = 'u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d' % \
{'s1': family, 's2': int(startlbl), 's3': eps, 'd4': i}

        fn = uSolution

```

```

        bkSingleSolution(uSolution,all_flg,special_flg,\
familyLog_fds,bk_lines,fn,fm,stbl,mult,eng,fullBkFolder)
        uSolution = 'u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d_rst' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':i}
        fn = uSolution
        bkSingleSolution(uSolution,all_flg,special_flg,\
familyLog_fds,bk_lines,fn,fm,stbl,mult,eng,fullBkFolder)
        uSolution = 'u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d_DS-' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':i}
        fn = uSolution
        bkSingleSolution(uSolution,all_flg,special_flg,\
familyLog_fds,bk_lines,fn,fm,stbl,mult,eng,fullBkFolder)
        uSolution = 'u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d_DS+' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':i}
        fn = uSolution
        bkSingleSolution(uSolution,all_flg,special_flg,\
familyLog_fds,bk_lines,fn,fm,stbl,mult,eng,fullBkFolder)
        i=i+1
        familyLog_fds.close(); bk_fds.close()
        special_flg.close(); all_flg.close()
        cmds = 'cp %(s1)s %(s2)s/%(s1)s' % \
{'s1':startSolution,'s2':fullBkFolder}
        print cmds
        copyPathSolution (startSolution,fullBkFolder,\
startSolution)
        #back up the logs
        cmds = 'cp %(s1)s %(s2)s/%(s1)s' % \
{'s1':logFile_special,'s2':fullBkFolder}
        print cmds; os.system(cmds)
        cmds = 'rm %(s1)s' % {'s1':logFile_special}
        print cmds; os.system(cmds)
        cmds = 'cp %(s1)s %(s2)s/%(s1)s' % \
{'s1':logFile_all,'s2':fullBkFolder}

```

```

print cmds; os.system(cmds)

cmds = 'rm %(s1)s' % {'s1':logFile_all}; print cmds

os.system(cmds)

cmds = 'cp %(s1)s %(s2)s/%(s1)s' % {'s1':'BK.LIST','s2':fullBkFolder}
print cmds; os.system(cmds)

#remove solutions
if control == 5:
    family = sys.argv[3]; startlbl = sys.argv[4]; eps = sys.argv[5]
    startSolution = 'start%(s1)s_%(s2)03d_%(s3)s' % \
{'s1':family,'s2':int(startlbl),'s3':eps}
    if not ifASolutionExist(startSolution):
        print "solution ",startSolution," does not exist!";os._exit(99)
    print 'clean solutions of last computation',startSolution
    instStartSolution = sl(startSolution)[-1]
    maxlabel = int(instStartSolution["Label"])
    i = 1
    while True:
        if i > maxlabel:
            break
        str_rstlabel = '%(d1).d'%'{'d1':i}
        uSolution = 'u%(s1)s_%(s2)03d_%(s3)s_%(d4)04d' % \
{'s1':family,'s2':int(startlbl),'s3':eps,'d4':i}
        if ifASolutionExist(uSolution):
            print 'Remove solution ',uSolution
            removeASolution(uSolution)
            removeASolution('%(s1)s_rst'%'{'s1':uSolution})
            removeASolution('%(s1)s_DS+'%'{'s1':uSolution})
            removeASolution('%(s1)s_DS-'%'{'s1':uSolution})
            i=i+1
        if ifASolutionExist(startSolution):
            removeASolution(startSolution)
print 'rm -f *~'; os.system('rm *~'); os._exit(99)

```