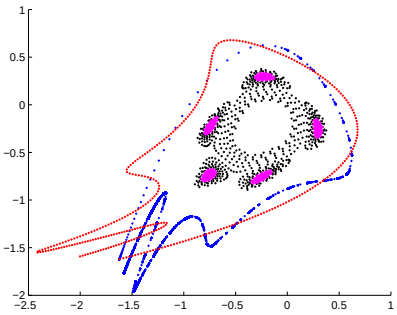
Bifurcations of maps: numerical algorithms and applications

Reza Khoshsiar Ghaziani



Supervisor: Prof. Dr. W. Govaerts

Co-supervisor: Prof. Dr. Yu. A. Kuznetsov, Universiteit Utrecht

In Partial Fulfillment of the Requirements for the Degree
Doctor of Philosophy in Mathematics

Academic year 2007–2008

Faculty of Science

Department of Applied Mathematics and Computer Science

UNIVERSITEIT GENT

FACULTEIT WETENSCHAPPEN

Contents

Introduction

Dynamical systems theory is a classical branch of mathematics which began with Newton around 1665. It provides mathematical models for systems which evolve in time according to a rule, originally expressed in analytical form as a system of equations. Dynamical systems theory is a rapidly growing area, which plays an important role in almost all disciplines of science and engineering including physics, finance, geology, biology and chemistry, to name a few. Dynamical systems as models for physical or biological systems have parameters which must be determined by measurement or data fitting.

In the 1880s, Poincaré studied continuous dynamical systems in connection with a prize competition on the stability of the solar system. He found it convenient to replace the continuous flow of time with a discrete analogue, in which time increases in regular, saltatory jumps. These systems are now called discrete dynamical systems. So, for over a century, dynamical systems have come in two flavors:

- Continuous time differential equations.
- Discrete-time maps.

In this study we will follow the latter route, i.e. we consider discrete dynamical systems expressed as the iteration of a map of the general form: $x \longmapsto f(x, \alpha)$ where $x \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}^p$ are vectors of state variables and parameters, respectively. Usually, it is convenient to introduce a discrete-time dynamical system when events occur or are accounted for only at discrete time periods. For instance, when developing a population model, it may be convenient to work with

annual population changes, and the data is normally available annually rather than continually.

As parameters are varied in a dynamical model, the system may undergo qualitative changes or bifurcations. Bifurcation analysis was introduced by Poincaré and developed to a high art by mathematicians such as Hopf, Andronov, and many others during the 20th century. It allows us to identify and predict changes or metamorphoses in the dynamics of a system, such as stability changes, abrupt transitions, hysteresis, the onset of oscillations, changes in the types of oscillations (such as period-doubling), or extinction. Bifurcation analysis is a very practical science, because it allows us to characterise all of the dynamical behaviour that a process is capable of, and therefore make useful designs and predictions. Many physical and biological systems include free parameters, and bifurcation analysis can help us understand how the behaviour of these systems varies as the parameters change.

There are in essence two approaches to defining bifurcations in smooth systems, analytical and topological. In the analytical approach a bifurcation is defined as the branching, folding or creation of additional paths of solutions of a certain class within a bifurcation diagram, e.g. [53, 20]. In the topological sense, a bifurcation is a parameter value within a class of systems at which the phase portrait is not structurally stable, e.g [49, 62]. A universal unfolding (or topological normal form) of the bifurcation includes a minimal number of terms and parameters to allow all possible structurally stable bifurcation diagrams to be seen at small values of the unfolding parameters.

The main goal in the study of a dynamical system is to find a complete characterization of the geometry of the orbit structure and the change in orbit structure under parameter variation. An aspect of this study is to identify the invariant objects and the local behaviour around them. This local information then needs to be assembled in a consistent way by means of geometric and topological arguments, to obtain a global picture of the system. The aim is to find qualitative and often also quantitative representations of the different types of behaviour that the system may exhibit in dependence on parameters. The main result of such effort is a bifurcation diagram, that is, information on the division of the parameter space into regions of topologically different behaviour together with representatives of phase portraits.

When it comes to study in an explicitly given model how the behaviour changes as a function of parameters, the tool of choice is numerical continuation methods, as opposed to simulation methods in which many orbits are computed by starting from various initial points. The basic idea is to compute a curve that is implicitly defined by a suitable system of equations that defines the dynamical object under consideration.

The codimension (codim for short) of a (sub-)manifold is the difference in dimension between full parameter-space and the (sub-)manifold. The codim of a bifurcation is the highest codim of the (sub-)manifolds that exhibit the bifurcation. In a more practical definition, the codim of a bifurcation is the number of independent conditions determining the bifurcation.

At local bifurcations, the number of steady states can change, or the stability properties of a steady state may change. To study this we compute the Jacobian matrix of the system and compare the behaviour of nonlinear and linear systems in the vicinity of the bifurcation points. The eigenvalues of the Jacobian are called multipliers. These are exponential rates of growth and decay for solutions to the linear system. The geometry of linear and nonlinear systems is similar to one another near a hyperbolic fixed point, i.e. a point with no multipliers on the unit circle in the complex plane. In particular, there are stable and unstable manifolds tangent to the eigendirections of decaying and growing eigenvalues that consist of those trajectories that approach the fixed point in the forward and backward iteration of the map, respectively. By means of center manifold theory, reduction of dimensionality, and the method of normal forms, a dynamical system near a nonhyperbolic fixed point, can be simplified. The computational analysis of local bifurcations usually begins with an attempt to compute the coefficients that appear in the normal form after coordinate transformation. These coefficients, called critical normal form coefficients, determine the direction of branching of new objects and their stability near the bifurcation point.

After locating a codim 1 bifurcation point, the logical next step is to consider the variation of a second parameter to enhance our knowledge about the system and its dynamical behaviour. When system parameters are varied simultaneously, they usually generate a bifurcation curve which can be drawn in a two-parameter plane. We refer to these curves as codim 1 bifurcation curves. These curves can be computed using a minimally extended system which consists of a fixed point

condition and a certain singularity condition.

We often encounter a nongeneric situation in one-parameter problems where the implicit function theorem cannot be applied to ensure the existence of a unique smooth branch of fixed points. It is encountered often in practical problems that exhibit some form of symmetry (equivariance). Usually, such points are simple branch points. The smooth curve that emanates from a branch point can be continued by computing the tangent vector along the new branch.

In codim 2 bifurcation points branches of various codim 1 bifurcation curves are rooted. The problem of branch switching is thus to specify one starting point near the curve from which the continuation code converges to a point on the curve. This can be done by a combination of parameter-dependent center manifold reduction and asymptotic expressions for the new emanating curves.

There are several computational software packages that aid in the bifurcation analysis of cycles of a map. Orbits of maps and one-dimensional invariant manifolds of saddle fixed points can be computed and visualized using DYNAMICS [76, 108, 109] and DsTool [10, 58, 60]. Location and continuation of fixed-point bifurcations is implemented in AUTO [28] and the LBFP-version of LOCBIF [54]. The latter program computes the critical normal form coefficient at LP points and locates some codim 2 bifurcations along branches of codim 1 fixed points and cycles. CONTENT [61] was the first software that computed the critical normal forms coefficients for all three codim 1 bifurcations of fixed points and cycles and allowed to continue these bifurcations in two parameters and to detect all eleven codim 2 singularities along them. Branch switching at PD and BP points is also implemented in AUTO, LOCBIF, and CONTENT. However, only trivial branch switching is possible at codim 2 points and only for two (cusp and 1:1 resonance) of eleven codim 2 bifurcations the critical normal form coefficients are computed by the current version of CONTENT. No software supports switching at codim 2 points to the continuation of the double- , triple- and quadruple-period codim 1 bifurcation curves.

We discuss new and improved algorithms for the bifurcation analysis of fixed points and periodic orbits (cycles) of maps and their implementation in the MATLAB software package CL_MATCONTM. This includes the numerical continuation of fixed points of iterates of the map with one control parameter, detecting and locating their bifurcation points (i.e. LP, PD and NS), and their continuation in two

control parameters, as well as detection and location of all codim 2 bifurcation points on the corresponding curves. For all bifurcations of codim 1 and 2, the critical normal form coefficients are computed with finite directional differences, automatic differentiation and symbolic derivatives of the original map. Using a parameter-dependent center manifold reduction, explicit asymptotics are derived for bifurcation curves of double and quadruple period cycles rooted at codim 2 points of cycles with arbitrary period. These asymptotics are implemented into the software and allow one to switch at codim 2 points to the continuation of the double and quadruple period bifurcations.

In Chapter 1 we introduce CL_MATCONTM, a MATLAB toolbox for continuation and bifurcation analysis of discrete dynamical systems, and describe its functionalities. We first give an overview of the continuation methods used in the software, test functions of bifurcations, and the singularity matrices. We proceed with a short review of software packages for bifurcations of cycles of maps. We continue to consider some specific features of CL_MATCONTM and discuss some components of a continuation process. We give detailed descriptions of the initializations of different curves and the global structures corresponding to different continuation curves, the information flow in the continuation of a solution curve and the initializers of the switching curves.

Chapter 2 is devoted to the analytical study of bifurcations of cycles of a map with one and two parameters. All codim 1 and codim 2 bifurcations are listed. A review of these results are given in [69] and references therein.

In Chapter 3 we describe some techniques to compute bifurcation curves. This includes bifurcation analysis of codim 2 bifurcation points, continuation of codim 1 curves, detection and location of their bifurcation points. We proceed with the branch switching techniques to compute curves of codim 1 bifurcations emanating in codim 2 points. All codim 1 and codim 2 bifurcation points along with their possible branch switchings are depicted in detection and switching diagrams.

In Chapter 4 we present the recursive formulas to compute multilinear forms up to the fifth order and algorithms to compute tensor-vector and vector-tensor-vector products, which are needed not only for the continuation but also for the computation of the critical normal form coefficients at codim 1 and 2 bifurcation points and for branch switching. We proceed this chapter by considering finite difference directional derivatives (FD) and automatic differentiation (AD). If

symbolic derivatives (SD) of the original map are not available, as an alternative finite differences can be used. However not the full tensors are needed, but the multilinear forms evaluated on vectors which can be computed with directional derivatives and central finite differences.

As an alternative to symbolic derivatives and finite differences for computing normal form coefficients, we have implemented automatic differentiation techniques, to compute derivatives of an iterated map, w.r.t state variables. As AD is as accurate as symbolic derivatives, it can be used when SD is not available. Another advantage of using AD is to speed up the computation of critical normal form coefficients for higher iteration numbers of a map. We discuss the technique of automatic differentiation and its implementation in the software CL_MATCONTM. We close this chapter by a comparison of elapsed time and accuracy of the three different differentiation strategies.

Among other things, dynamical systems are used to model biological phenomena. A major goal of biological modeling is to quantify how things change. The combination of nonlinear dynamics and biology has brought about significant advances in both areas, with nonlinear dynamics providing a tool for understanding biological phenomena and biology stimulating developments in the theory of dynamical systems [106]. A very common and useful tool for investigating future demographics is the age-structured population model in which populations are not tracked in their totality, but rather according to their age class. By means of bifurcation theory, we can examine the effect of the parameters to make quantitative predictions.

In Chapter 5 we consider two age-structured populations, namely a Leslie-Gower competition model for the interaction of two different species of the flour beetle Tribolium and a cod stock model. The Leslie-Gower model is a discrete time analog of the competition Lotka-Volterra model and is known to possess the same dynamic scenarios of that famous model. The Leslie-Gower model played a key historical role in laboratory experiments that helped to establish the competitive exclusion principle in ecology and in its application to classic laboratory experiments of two competing species of flour beetles. In this model by means of bifurcation analysis, we find that there is an interior region in which the coexistence fixed points are unstable. This region is bounded by a PD curve, where the stability changes. By branch switching we compute the branches of fold curves

of the second iterate that emanate at the GPD points. From the applications point of view, this is the most interesting region because it shows that indeed the two species can coexist even when the competition is strong.

The second model is an age-structured cod stock model in which competitive interactions pass through different physiological stages during their development progresses and therefore competition occurs not only within species, but also within and between stages of different species. Age-structured competitive interactions have always been of interest to ecologists as they have been shown to have important dynamical consequences for intra- and interspecific population interactions. We compute the stability domains of the map and its iterates. We consider the second, third and fourth iterates of the map and their relation to period-doubling, R3 and R4 points. In particular, we compute two different branches of fold curves of the fourth iterate emanating from a R4 point. These curves form stability boundaries of 4-cycles in which the population oscillates between different values.

In economic dynamics, discrete-time models have steadily been gaining in popularity. Discrete-time economic models may be thought of as providing a better framework for economic analysis. The first and still one of the most widely cited models of noncooperative oligopoly behaviour is the Cournot model, developed by the French mathematician Augustin Cournot in 1838. The Cournot model is the fundamental model used to study strategic interactions among quantity-setting firms in an imperfectly competitive market. In the last two decades, there has been an explosion of Cournot-based models of strategic behaviour to analyze various real-world phenomena ranging from horizontal mergers to intra-industry trade. A proper understanding of the Cournot model of imperfect competition and strategic interactions among firms in various contexts is thus essential. In Chapter 6 we study two map models in economics. First we consider two-dimensional (duopoly) and three-dimensional (oligopoly) Cournot models. We compute a closed NS curve in the parameter region of the original map as the stability boundaries of the fixed points. The normal form coefficient changes sign when crossing the Chenciner (CH) bifurcation points. The economically relevant stability region is bounded by the lines, derived by the feasibility condition of the Cournot points, and subcritical parts of the NS curve.

In the second economics model, we consider a two-dimensional map proposed

by M.Kopel that models a nonlinear Cournot duopoly consisting of a market structure between the two opposite cases of monopoly and competition. Synchronization of two dynamics parameters of the Cournot duopoly is obtained in the computation of stability boundaries formed by parts of codim 1 bifurcation curves. We study the dynamics of the map by computing numerically the critical normal form coefficients of all codim 1 and codim 2 bifurcation points and computing the associated two-parameter codim 1 curves rooted in some codim 2 points. It enables us to compute the stability domains of the low-order iterates of the map. We compute regions of the parameter space in which there is multistability of a symmetric 4-cycle, a non-symmetric 4-cycle, a 2-cycle and fixed points of the original map.

Stable and unstable manifolds of invariant manifolds of saddle-type, in particular saddle fixed points, play important roles in organizing the global dynamics. It is well-known that the stable manifolds can form boundaries between different basins of attraction. These manifolds are global, often noncompact objects that can have a very complicated structure. Furthermore, the transverse intersection of stable and unstable manifolds leads to homoclinic or heteroclinic tangles, associated with chaos [80]. Only in special situations it is possible to find stable and unstable manifolds analytically. In general they need to be computed with numerical methods.

In Chapter 7, we discuss the implementation of computational algorithms for one-dimensional invariant manifolds [32, 58] and their transversal intersections to obtain initial connections of homoclinic and heteroclinic orbits. We implement an algorithm to compute the stable manifold of a saddle point of a planar map, without requiring any knowledge of its inverse map, either explicitly or approximately. We use the so called SEARCH CIRCLE algorithm [32] which uses the idea of growing a one-dimensional manifold in steps by adding new points according to the local curvature properties of the manifold and finding a new point close to the last computed point that maps under F to a piece of the manifold that was already computed. The idea for computing an unstable manifold, similar to the algorithm for computing the stable manifold, is to grow the manifold independently of the dynamics in steps as a list of ordered points [58]. At each step a new point is added at a prescribed distance from the last point. New points are found as f-images of suitable points from the part we already computed. The algorithm starts

with a linear approximation of the local manifold and grows the manifold up to a prespecified arclength with a speed depending on the curvature of the manifold.

Transversal intersection of these manifolds provides initial approximations for the homoclinic and heteroclinic connections. We use an improved algorithm for locating and continuing connecting orbits, homo- and heteroclinic, which includes an algorithm for the continuation of invariant subspaces (CIS) as described in [25, 27]. The software then supports the continuation of these connections in one parameter as well as detection and location of limit points along these orbits. Next the software allows one to continue curves of limit points in two parameters, i.e. computing homoclinic and heteroclinic tangencies. We illustrate the algorithms by computing invariant manifolds of the generalized Hénon map. We compute homoclinic and heteroclinic orbits in one parameter and curves of homoclinic and heteroclinic tangencies in two parameters.

Finally, in chapter 8 we discuss open problems and directions for further research.

The results of this thesis were published, accepted for publication or submitted in several specialized journals or proceedings, see [39], [56], [40], [41], [42], [55].

CL_MATCONTM is freely available at http://www.matcont.UGent.be. A user manual is also provided there [43].

Acknowledgments

Working as a Ph.D student at Ghent University was a magnificent experience to me. In all these years, many people were directly or indirectly involved in shaping up my academic career. Here is a small tribute to all those people.

First and foremost, I would like to express my appreciation to my supervisor Prof. Willy Govaerts, who gave me the opportunity to do my Ph.D. Willy, I'm grateful for your careful and professional guidance, patience, understanding and perpetual encouragement, and for putting up with me for four years. I feel truly lucky and privileged to have been your student and indebted to you more than you know.

To my associate supervisor, I was delighted to interact with Prof. Yuri Kuznetsov at Utrecht University. Yuri, I sincerely thank you for your substantial contribution to the development of this work, your valuable supervision, guidance and suggestions.

I have collaborated with some colleagues for whom I have great regard and I wish to extend my warmest thanks to them. My deepest gratitude goes to Dr. Hil Meijer at Twente University for his great influence on my work, constructive comments, valuable discussions and his friendly helps. His valuable comments also as a member of reading committee improved my thesis substantially.

It is also a pleasure particularly to thank Prof. John Pryce at Cranfield University for his collaboration and for many helpful discussions. I benefited from his great expertise in automatic differentiation.

My special thanks go to the other members of my examination committee who

monitored my work and took the effort to read and provide valuable comments on earlier versions of this manuscript: Prof. André Vanderbauwhede, Prof. Joris Van der Jeugt, Prof. Marnix Van Daele, Dr. Veerle Ledoux. I thank you all.

I would also like to convey thanks to the Iranian Ministry of Science and Technology for providing the financial support of my PhD. My special appreciation goes to the former rector of Sharhrekord University Dr. A. Firouz and my colleagues at Shahrekord University for all their support, in particular Dr. B. Shareghi, Dr. M. Rismanchian, Dr. J. Asadollahi, Dr. H. Mansoory, Dr. Gh. Rezaeezadeh, Dr. Kh. Hessami, and Dr. H. Shahgholian.

Thanks also to all my colleagues at the department of applied mathematics and computer science for providing a good working atmosphere. I want also to acknowledge the support of the department and specially the head of department, Prof. Guido Vanden Berghe, for my participation in various international conferences.

Furthermore, my thanks go to Annick Dhooge and Bart Sautois for helping me to get started with and developing CL_MATCONTM. Special thanks to Leila Kheibarshekan for her comments on my work and her friendship.

I thank my officemates Guy Baele, Adriaan Peeters and Virginie Dewitte and my ex-officemate Jan Degraer, for their friendship and for making my graduate experience memorable and fun.

During my stay in Belgium, I met many wonderful people, in particular Nicole and Jacqueline Van Herreweghe, who made me feel at home in Belgium. I will always cherish the wonderful time spent with them.

I would like to thank all my Iranian friends for their help, support, and friendship. The following people deserve a special mention: Dr. A. Nemetollahi, Dr. V. Rouhid, M. Rafieiolhossaini, Dr. M. Karimi, Dr. A. Shiri, S. Karimi, F. Fayazpour, A. Rezayazdi, E. Hossayni, K. Zendehdel, A. Sadeghi, M. Babanezhad, A. Gholamina and A. Pezeshki.

My deepest gratitude goes to my family for their love and support throughout my life. I am indebted to my father, for his care and love. Although he is no longer with us, he is forever remembered. I cannot find a suitable word to fully acknowledge my mother for her care, support and prayers. I specially thank my

brothers and sisters who have supported me throughout my studies.

My deep love and appreciation goes to the family of my wife, my father and mother-in-law in particular, for their continuous and unconditional support before and during my PhD.

My profound appreciation goes to my wife Maryam whose dedication, continual love and support, and persistent confidence in me has taken the load off my shoulders. I thank her for sharing her intelligence, passion and ambition with me. This thesis is dedicated to her.

I cannot finish this text without apologizing to all the people whom I did not mention here but were responsible, in one way or another, for the completion of this thesis.

Reza Khoshsiar Ghaziani
March 2008

Chapter 1

Features of CL_MATCONTM

To fix the notation, we consider a discrete map of the general form

$$x \mapsto f(x, \alpha), \tag{1.1}$$

where $x \in \mathbb{R}^n$ is a state variable vector and $\alpha \in \mathbb{R}^p$ is a parameter vector. We assume that f is sufficiently smooth so that all partial derivatives are well defined. The K-th iterate of (1.1) at some parameter value is defined by

$$x \mapsto f^{(K)}(x, \alpha), \;\; f^{(K)} : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n, \tag{1.2}$$

where

$$g(x, \alpha) := f^{(K)}(x, \alpha) = \underbrace{f(f(f(\cdots f}_{K \text{ times}}(x, \alpha), \alpha), \alpha), \alpha). \tag{1.3}$$

For each x_0, the iteration (1.3) generates a sequence of points defining the orbit, or trajectory of x_0, under the map g. The bifurcation analysis of (1.3) usually starts with fixed points. Numerically we continue fixed points of this map, i.e. solutions to the equation

$$F(x, \alpha) \equiv g(x, \alpha) - x = 0, \tag{1.4}$$

with one control parameter. While varying the parameter, one may encounter codim 1 bifurcations of fixed points, i.e. critical parameter values where the stability of the fixed point changes. The eigenvalues of the Jacobian matrix J of

$f^{(K)}$ are called multipliers. Fixed points can be classified according to the orbit behavior of points in their vicinity. A fixed point x is said to be asymptotically stable (or an attractive point of g), if for increasing K, trajectories of points near the asymptotically stable fixed point tend toward it. A fixed point is asymptotically stable if $|\lambda| < 1$ for every multiplier λ. The multipliers of g govern the contracting and expanding directions of the map g in a vicinity of x. Eigenvalues larger in absolute value than 1 lead to expansions, whereas eigenvalues smaller than 1 correspond to contractions. If there exists a multiplier λ with $|\lambda| > 1$, then the fixed point is unstable. If all the multipliers are outside the unit circle, x is a repulsive point. The trajectories of points from a neighborhood of a repulsive point move away from it. If some multipliers of J are inside and some are outside the unit circle, x is said to be a saddle point.

While following a curve of fixed points, three codimension 1 singularities related to stability changes can generically occur, namely a *limit point* (*fold*, LP), a *period-doubling* (*flip*, PD) and a *Neimark-Sacker* (NS) point. Encountering such a bifurcation one may use the formulas for the normal form coefficients derived via the center manifold reduction to analyse the bifurcation. A nongeneric situation occurs at a branch point (BP) where the Jacobian matrix $[F_x(x,\alpha), F_\alpha(x,\alpha)]$ of (1.4) is rank deficient. Here the implicit function theorem cannot be applied to ensure the existence of a unique smooth branch of solutions. However, it is encountered often in practical problems that exhibit some form of symmetry (equivariance).

This chapter is organized as follows: We first give a review of the existing software packages for numerical bifurcations of maps. We proceed with some aspects of numerical methods, test functions for bifurcations and the singularity matrix. Then we describe the MATLAB toolbox CL_MATCONTM and its functionalities.

1.1 Software for bifurcations of maps

There are several standard software packages supporting bifurcation analysis of iterated maps. Orbits of maps and one-dimensional invariant manifolds of saddle fixed points can be computed and visualized using DYNAMICS [76] and DsTool [10]. Location and continuation of fixed-point bifurcations is implemented in

AUTO [28] and the LBFP-version of LOCBIF [54]. The latter program computes the critical normal form coefficient at LP points and locates some codim 2 bifurcations along branches of codim 1 fixed points and cycles. CONTENT [61] was the first software that computed the critical normal form coefficients for all three codim 1 bifurcations of fixed points and cycles and allowed to continue these bifurcations in two parameters and to detect all eleven codim 2 singularities along them. Branch switching at PD and BP points is also implemented in AUTO, LOCBIF, and CONTENT. However, only trivial branch switching is possible at codim 2 points and only for two (cusp and 1:1 resonance) of eleven codim 2 bifurcations the critical normal form coefficients are computed by the current version of CONTENT. No software supports switching at codim 2 points to the continuation of the double- and quadruple-period bifurcation curves.

1.2 Some aspects of numerical continuation methods

Numerical continuation

In general, numerical continuation methods are used to compute solution manifolds of nonlinear systems of the form:

$$G(X) = 0, \tag{1.5}$$

where $X \in \mathbb{R}^{n+p}$ and $G : \mathbb{R}^{n+p} \to \mathbb{R}^n$ is a sufficiently smooth function. The solutions of this equation consist of regular pieces, which are joined at singular solutions. The regular pieces are curves when $p = 1$, surfaces when $p = 2$ and p-manifolds in general.

We use numerical continuation methods for analyzing the solutions of (1.5) when restricted to the case $p = 1$. In fact, we construct solution curves Γ in

$$\{X : G(X) = 0\}, \tag{1.6}$$

by generating sequences of points $X_i, i = 1, 2, ...$ along the solution curve Γ satisfying a chosen tolerance criterion. The general idea of a continuation method is that of a predictor-corrector scheme [7]. Starting with an initial point on the continuation path, the goal is to trace the remainder of the path in steps. At each step,

the algorithm first predicts the next point on the path, and subsequently corrects the predicted point towards the solution curve. A variant of Newton's method is used for the corrector step. For details of the continuation method used in CL_MATCONTM, we refer to [29, 30]. We note that these publications deal with the original ODE work, but the continuation method is identical.

Test functions for bifurcations

Let $X = X(s)$ be a smooth, local parameterization of a solution curve of (1.5) where $p = 1$. Suppose that $s = s_0$ corresponds to a bifurcation point. A smooth scalar function $\psi : \mathbb{R}^{n+1} \to \mathbb{R}^1$ defined along the curve is called a test function, a tool to detect singularities on a solution branch, for the corresponding bifurcation if $g(s_0) = 0$, where $g(s) = \psi(X(s))$. The test function ψ has a regular zero at s_0 if $\frac{dg}{ds}(s_0) \neq 0$. A bifurcation point is detected between two successive points X_0 and X_1 on the curve if $\psi(X_0)\psi(X_1) < 0$. To solve the system

$$\begin{cases} G(X) = 0 \\ \psi(X) = 0 \end{cases} \tag{1.7}$$

we use a one-dimensional secant method to locate $\psi(X) = 0$ along the curve. Notice that this involves Newton corrections at each intermediate point.

1.3 CL_MATCONTM

1.3.1 Functionalities

CL_MATCONTM is a continuation toolbox in MATLAB [68], to continue fixed points of an iterated map. It supports the following functionalities:

- continuation of fixed points of maps and iterates of maps with respect to a control parameter
- detection of fold, flip, Neimark-Sacker and branch points on curves of fixed points

- computation of normal form coefficients for fold, flip and Neimark-Sacker bifurcations
- continuation of fold, flip and Neimark-Sacker bifurcations in two control parameters
- detection of all eleven codim 2 fixed point bifurcations on curves of fold, flip and Neimark-Sacker bifurcations
- computation of normal form coefficients for all codim 2 bifurcations of fixed points
- switching to the period doubled branch in a flip point
- branch switching at branch points of fixed points
- switching to branches of codim 1 bifurcations rooted in codim 2 points
- computation of one-dimensional invariant manifolds of saddle fixed points. We remark that these manifolds are grown by special algorithms which are not related to the continuation process
- continuation of homoclinic and heteroclinic connections and detection and location of fold points along these connections
- continuation of fold curves of homoclinic connections (homoclinic tangencies) in two parameters
- continuation of fold curves of heteroclinic connections (heteroclinic tangencies) in two parameters
- automatic differentiation for normal form coefficients of codim 1 and codim 2 bifurcations

We proceed now to describe the computational kernel of the software

1.3.2 Continuer

A solution curve can be continued by using a *continuer* with the syntax:

$$[x, v, s, h, f] = cont(@curve, x0, v0, options)$$

curve is a MATLAB m-file where the problem is specified, cf. §1.3.3.
x0 and *v0* are respectively the initial point and the tangent vector at the initial point where the continuation starts.
options is a structure as described in §1.3.4.
The function returns:
x and v, i.e. the points and their tangent vectors along the curve. Each column in x and v corresponds to a point on the curve.
s is an array whose structure contain information on detected singularities. This structure has the following fields:

s.index	index of the singularity point in x.
s.label	label of the singularity.
s.data	any kind of extra information.
s.msg	a string containing a message for this particular singularity.

A special point on a bifurcation curve that is specified by a user function has a structure as follows:

s.index	index of the detected singular point defined by the user function.
s.label	a string that is in *UserInfo.label, label of the singularity.*
s.data	an empty tangent vector, values of the test and user functions in the singular point.
s.msg	a string that is set in *UserInfo.name.*

h is used for output of the algorithm, currently this is a matrix with for each point a column with the following components (in that order) :

- Stepsize:
  Stepsize used to calculate this point (zero for initial point and singular points).

- Half the number of correction iterations, rounded up to the next integer
  For singular points this is the number of locator iterations

- User function values :
  The values of all active user functions.
- Test function values :
  The values of all active test functions.

In general, *f* can be anything depending on which curve file is used. However, in CL_MATCONTM, *f* always contains the multipliers if they were computed during the continuation. Multipliers are computed when *options* is set by :

options=contset(options,'Multipliers',1),

see §1.3.4 for more details.
It is also possible to extend the most recently computed curve with the same options (also the same number of points) as it was first computed. The syntax to extend this curve is:

[x, v, s, h, f] = cont(x, v, s, h, f, cds)

x, *v*, *s*, *h* and *f* are the results of the previous call to the continuer and cds is the global variable that contains the curve description of the most recently computed curve. The function returns the same output as before, extended with the new results.

1.3.3 Curve file

The continuer uses special m-files where the type of the solution branch is defined. CL_MATCONTM contains eight curve files namely *fixedpointmap.m*, *limitpointmap.m*, *perioddoublingmap.m*, *neimarksackermap.m*, *heteroclinic*, *homoclinic*, *heteroclinicT*, *homoclinicT* in which defining systems for fixed points, fold, flip and Neimark-Sacker, heteroclinic orbits, homoclinic orbits, heteroclinic and homoclinc tangencies of cycles of maps are defined, respectively.

A curve file contains some sections as *curve_func*, *jacobian*, *hessians*, *adapt*,... etc. In some cases the problem definition uses auxiliary entities like bordering vectors and it may be needed to adapt them during the continuation. In *adapt* these entities are adapted. If *cds.options.Adapt* has a value n, then after n computed points a call to *[reeval,x,v]=feval(cds.curve_adapt,x,v)* will be made.

1.3.4 Options

In the continuation we use the $options$ structure which is initially created with $contset$:
options = contset
will initialize the structure. The continuer stores the handle to the options in the variable *cds.options*. Options can then be set using
options = contset(options, optionname, optionvalue),
where $optionname$ is an option from the following list.

InitStepsize the initial stepsize (default: 0.01)

MinStepsize the minimum stepsize to compute the next point on the curve (default: 10^{-5})

MaxStepsize the maximum stepsize (default: 0.1)

MaxCorrIters maximum number of correction iterations (default: 10)

MaxNewtonIters maximum number of Newton-Raphson iterations before switching to Newton-Chords in the corrector iterations (default: 3)

MaxTestIters maximum number of iterations to locate a zero of a test function (default: 10)

Increment the increment to compute first-order derivatives numerically (default: 10^{-5})

FunTolerance tolerance of function values: $||F(x)|| \leq FunTolerance$ is the first convergence criterium of the Newton iteration (default: 10^{-6})

VarTolerance tolerance of coordinates: $||\delta x|| \leq VarTolerance$ is the second convergence criterium of the Newton iteration (default: 10^{-6})

TestTolerance tolerance of test functions (default: 10^{-5})

Singularities boolean indicating the presence of singularities (default: 0)

MaxNumPoints maximum number of points on the curve (default: 300)

Backward boolean indicating the sense of the continuation (sense of the initial tangent vector) v_0 (default: 0)

CheckClosed number of points indicating when to start to check if the curve is closed (0 = do not check) (default: 50)

Adapt number of points indicating when to adapt the problem while computing the curve (default: 1=adapt always)

IgnoreSingularity vector containing indices of singularities which are to be ignored (default: empty)

Multipliers boolean indicating the computation of the multipliers (default: 0)

Userfunctions boolean indicating the presence of user functions (default: 0)

UserfunctionsInfo is an array with structures containing information about the user functions. This structure has the following fields:

.label	label of the user function (must consist of four characters, including possibly trailing spaces)
.name	name of this particular user function
.state	boolean indicating whether the user function has to be evaluated or not

AutDerivative boolean indicating the use of automatic differentiation in the computation of normal form coefficients (default: 1)

AutDerivativeIte an integer number that indicates the use of automatic differentiation when the iteration number of the map equals or exceeds this number (default: 24)

For the options MaxCorrIters, MaxNewtonIters, MaxTestIters, Increment, FunTolerance, VarTolerance, TestTolerance and Adapt the default values are in most cases good.

Options also contains some fields which are not set by the user but frozen or filled by calls to the curvefile, namely:

MoorePenrose boolean indicating the use of the Moore-Penrose continuation as the Newton-like corrector procedure (default: 1)

SymDerivative the highest order symbolic derivative which is present (default: 0)

SymDerivativeP the highest order symbolic derivative with respect to the parameter(s) which is present (default: 0)

Testfunctions boolean indicating the presence of test functions and singularity matrix (default: 0)

WorkSpace boolean indicating to initialize and clean up user variable space (default: 0)

Locators boolean vector indicating the user has provided his own locator code to locate zeroes of test functions. Otherwise the default locator will be used (default: empty)

ActiveParams vector containing indices of the active parameter(s) (default: empty)

1.3.5 Singularity matrix

Suppose we have two singularities S_1 and S_2, and test functions ψ_1 and ψ_2. Assume that ψ_1 vanishes at both S_1 and S_2 while ψ_2 generically vanishes only at S_2. Then we need to require that ψ_2 does not vanish at S_1, i.e. we need the possibility to require that in some singularities certain test functions do not vanish. To represent all singularities we use a singularity matrix, i.e. a compact way to describe the relation between the singularities and the test functions. Suppose we have n_s singularities and n_t test functions. Then the singularity matrix S is an $n_s \times n_t$ matrix, such that:

$$S_{ij} = \begin{cases} 0 & \text{for singularity i testfunction j must vanish} \\ 1 & \text{for singularity i testfunction j must notvanish} \\ otherwise & \text{for singularity i ignore test function j} \end{cases} \quad (1.8)$$

1.3.6 Directories

To start CL_MATCONTM the MATLAB main directory must be the root directory of CL_ MATCONTM where *init.m* is set. The files of the toolbox are organized in the following subdirectories

- **Continuer**
  Here are all the main files for the continuer which are needed to calculate and plot any curve.

- **FixedPointMap**
  Here are all files needed to do a continuation of fixed points of iterates of a map. This includes in particular the initializers and the fixed point curve definition file.

- **LimitPointMap**
  Here are all files needed to do a fold continuation. This includes in particular the initializers and the fold curve definition file.

- **PeriodDoublingMap**
  Here are all files needed to do a flip continuation. This includes in particular the initializers and the flip curve definition file.

- **NeimarkSackerMap**
  Here are all files needed to do a Neimark-Sacker continuation. This includes in particular the initializers and the Neimark-Sacker point curve definition file.

- **MultilinearForms**
  Here are all files needed to compute the critical normal form coefficients for all codim 1 and codim 2 bifurcation points both numerically with finite directional differences and using symbolic derivatives of the original map.

- **AD**
  Contains all files needed to use automatic differentiation in the computation of multilinear forms.

- **Homoclinic**
  Here are all files needed to continue a homoclinic connection.
- **Heteroclinic**
  Contains all files needed to continue a heteroclinic connection.
- **HomoclinicT**
  Here are all files needed to continue a fold curve of homoclinic connections.
- **HeteroclinicT**
  Here are all files needed to continue a fold curve of heteroclinic connections.
- **InvManifolds**
  Contains the files to compute one-dimensional stable and unstable manifolds.
- **Systems**
  Here are all example system definitions.
- **Testruns**
  Here are all example testruns.

The only files which are not in any of these directories are *init.m* and *cpl.m*. The function *init.m* must be called before any continuation in the toolbox, so that MATLAB can find all the needed functions. The function *cpl.m* is used to plot the results obtained in a continuation run. It can provide 2D or 3D plots. For instance the commands *cpl(x24,v24,s24,[3 1])* and *cpl(x4,v4,s4,[3 1 2])* create the 2D and 3D plots of Figure 2.1 and Figure 2.2, respectively.
A sketch of the data flow in CL_MATCONTM is visualized in Figure 1.1.

1.3.7 The mapfile of the map

A solution curve must be initialized before doing a continuation. Each curve file has its own initializers which use a $mapfile$ where the map is defined, see for instance § 2.1.1. A $mapfile$ contains at least the following sections:

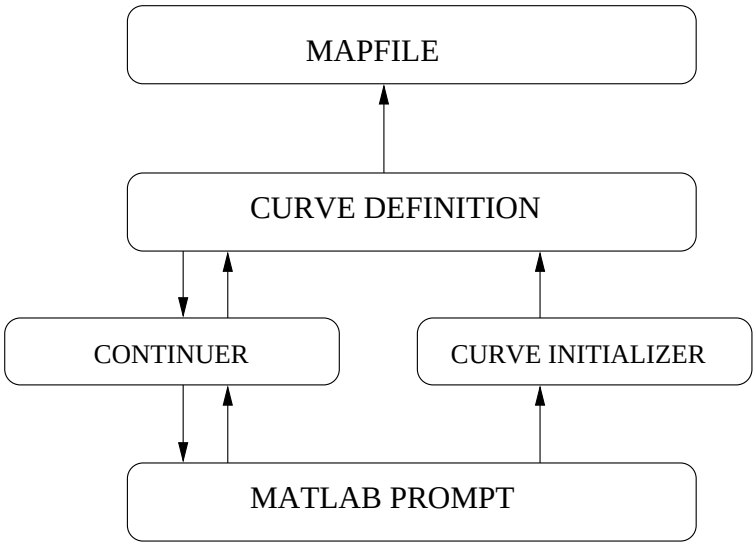


Figure 1.1: Continuation process in CL_MATCONTM.

init, fun_eval, jacobian, jacobianp, hessians, hessiansp, der3, der4 , der5.
A *mapfile* may also contain one or more sections that describe user functions.
In order to illustrate the elements of a *mapfile*, we give two m-files for M_{TN}, the map of a truncated normal form defined by:

$$M_{TN} : \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} \longmapsto \begin{pmatrix} -1 & 1 \\ \beta_1 & -1+\beta_2 \end{pmatrix} \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} + \begin{pmatrix} 0 \\ C\xi_1^3 + D\xi_1^2\xi_2 \end{pmatrix}, \tag{1.9}$$

using symbolic and numeric derivatives respectively. These files were created using MATCONT for ODEs by defining the problem in the 'System' window and then choosing the options symbolically or numerically to use symbolic or numeric derivatives, respectively. However, a *mapfile* can also be defined in CL_MATCONTM using the MATLAB editor. The user function sections can be created and added to a *mapfile* using the menu 'User function' in MATCONT or just by defining the user functions in the body of a *mapfile*. We note that user

functions, like test functions, must be scalar functions.

First we give the mapfile of M_{TN} using symbolic derivatives.

```
%------------------------------------------------
%The mapfile of the truncated normal form map
% with symbolic derivatives
%------------------------------------------------
function out = Tnfmap
out{1} = @init;
out{2} = @fun_eval;
out{3} = @jacobian;
out{4} = @jacobianp;
out{5} = @hessians;
out{6} = @hessiansp;
out{7} = @der3;
out{8} = @der4;
out{9} = @der5;
out{10}= @userf1;
out{11}= @userf2;
%------------------------------------------------
function dydt = fun_eval(t,kmrgd,beta1,beta2,CC,DD)
dydt=[-kmrgd(1)+kmrgd(2);; beta1*kmrgd(1)+(-1+beta2)*
kmrgd(2)+CC*kmrgd(1)^3+DD*kmrgd(1)^2*kmrgd(2);;];

%------------------------------------------------
function [tspan,y0,options] = init
handles = feval(Tnfmap);
y0=[0,0];
options = odeset('Jacobian',handles(3),'JacobianP',
handles(4),'Hessians',handles(5),'HessiansP',handles(6));
tspan = [0 10];

%------------------------------------------------
function jac = jacobian(t,kmrgd,beta1,beta2,CC,DD)
jac=[[-1,1];[beta1+3*C*kmrgd(1)^2+2*DD*kmrgd(1)*
     kmrgd(2),-1+beta2+DD*kmrgd(1)^2]];
%------------------------------------------------
function jacp = jacobianp(t,kmrgd,beta1,beta2,CC,DD)
jacp=[[0,0,0,0];[kmrgd(1),kmrgd(2),kmrgd(1)^3,
```

```
kmrgd(1)^2*kmrgd(2)]];
%------------------------------------------------
function hess = hessians(t,kmrgd,beta1,beta2,CC,DD)
hess1=[[0,0];[6*CC*kmrgd(1)+2*DD*kmrgd(2),
2*DD*kmrgd(1)]];
hess2=[[0,0];[2*DD*kmrgd(1),0]];
hess(:,:,1) =hess1;
hess(:,:,2) =hess2;
%------------------------------------------------
function hessp = hessiansp(t,kmrgd,beta1,beta2,CC,DD)
hessp1=[[0,0];[1,0]];
hessp2=[[0,0];[0,1]];
hessp3=[[0,0];[3*kmrgd(1)^2,0]];
hessp4=[[0,0];[2*kmrgd(1)*kmrgd(2),kmrgd(1)^2]];
hessp(:,:,1) =hessp1;
hessp(:,:,2) =hessp2;
hessp(:,:,3) =hessp3;
hessp(:,:,4) =hessp4;
%------------------------------------------------
function tens3  = der3(t,kmrgd,beta1,beta2,CC,DD)
tens31=[[0,0];[6*CC,2*DD]];
tens32=[[0,0];[2*DD,0]];
tens33=[[0,0];[2*DD,0]];
tens34=[[0,0];[0,0]];
tens3(:,:,1,1) =tens31;
tens3(:,:,1,2) =tens32;
tens3(:,:,2,1) =tens33;
tens3(:,:,2,2) =tens34;
%------------------------------------------------
function tens4  = der4(t,kmrgd,beta1,beta2,CC,DD)
tens41=[[0,0];[0,0]];
tens42=[[0,0];[0,0]];
tens43=[[0,0];[0,0]];
tens44=[[0,0];[0,0]];
tens45=[[0,0];[0,0]];
tens46=[[0,0];[0,0]];
tens47=[[0,0];[0,0]];
tens48=[[0,0];[0,0]];
```

```
tens4(:,:,1,1,1) =tens41;
tens4(:,:,1,1,2) =tens42;
tens4(:,:,1,2,1) =tens43;
tens4(:,:,1,2,2) =tens44;
tens4(:,:,2,1,1) =tens45;
tens4(:,:,2,1,2) =tens46;
tens4(:,:,2,2,1) =tens47;
tens4(:,:,2,2,2) =tens48;
%------------------------------------------------
function tens5  = der5(t,kmrgd,beta1,beta2,CC,DD)
tens51=[[0,0];[0,0]];
tens52=[[0,0];[0,0]];
tens53=[[0,0];[0,0]];
tens54=[[0,0];[0,0]];
tens55=[[0,0];[0,0]];
tens56=[[0,0];[0,0]];
tens57=[[0,0];[0,0]];
tens58=[[0,0];[0,0]];
tens59=[[0,0];[0,0]];
tens510=[[0,0];[0,0]];
tens511=[[0,0];[0,0]];
tens512=[[0,0];[0,0]];
tens513=[[0,0];[0,0]];
tens514=[[0,0];[0,0]];
tens515=[[0,0];[0,0]];
tens516=[[0,0];[0,0]];
tens5(:,:,1,1,1,1) =tens51;
tens5(:,:,1,1,1,2) =tens52;
tens5(:,:,1,1,2,1) =tens53;
tens5(:,:,1,1,2,2) =tens54;
tens5(:,:,1,2,1,1) =tens55;
tens5(:,:,1,2,1,2) =tens56;
tens5(:,:,1,2,2,1) =tens57;
tens5(:,:,1,2,2,2) =tens58;
tens5(:,:,2,1,1,1) =tens59;
tens5(:,:,2,1,1,2) =tens510;
tens5(:,:,2,1,2,1) =tens511;
tens5(:,:,2,1,2,2) =tens512;
```

```
tens5(:,:,2,2,1,1) =tens513;
tens5(:,:,2,2,1,2) =tens514;
tens5(:,:,2,2,2,1) =tens515;
tens5(:,:,2,2,2,2) =tens516;
%------------------------------------------------
function userfun1=userf1(t,kmrgd,beta1,beta2,CC,DD)
userfun1=beta2-2;
%------------------------------------------------
function userfun2=userf2(t,kmrgd,beta1,beta2,CC,DD)
userfun2=beta2-0.5;
%------------------------------------------------
```

A mapfile of M_{TN} without symbolic derivatives is given by:

```
%------------------------------------------------
%The mapfile of a truncated normal form map
% without symbolic derivatives
%------------------------------------------------
function out = Tnfmap1
out{1} = @init;
out{2} = @fun_eval;
out{3} = [];
out{4} = [];
out{5} = [];
out{6} = [];
out{7} = [];
out{8} = [];
out{9} = [];
out{10}= @userf1;
out{11}= @userf2;
%------------------------------------------------
function dydt = fun_eval(t,kmrgd,beta1,beta2,CC,DD)
dydt=[-kmrgd(1)+kmrgd(2);;
beta1*kmrgd(1)+(-1+beta2)*kmrgd(2)+CC*kmrgd(1)^3+
DD*kmrgd(1)^2*kmrgd(2);;];

%------------------------------------------------
function [tspan,y0,options] = init
handles = feval(Tnfmap1);
```

```
y0=[0,0];
options = odeset('Jacobian',[],'JacobianP',[],
'Hessians',[],'HessiansP',[]);
tspan = [0 10];

%------------------------------------------------
function jac = jacobian(t,kmrgd,beta1,beta2,CC,DD)
%------------------------------------------------
function jacp = jacobianp(t,kmrgd,beta1,beta2,CC,DD)
%------------------------------------------------
function hess = hessians(t,kmrgd,beta1,beta2,CC,DD)
%------------------------------------------------
function hessp = hessiansp(t,kmrgd,beta1,beta2,CC,DD)
%------------------------------------------------
function tens3  = der3(t,kmrgd,beta1,beta2,CC,DD)
%------------------------------------------------
function tens4  = der4(t,kmrgd,beta1,beta2,CC,DD)
%------------------------------------------------
function tens5  = der5(t,kmrgd,beta1,beta2,CC,DD)
%------------------------------------------------
function userfun1=userf1(t,kmrgd,beta1,beta2,CC,DD)
userfun1=beta2-2;
%------------------------------------------------
function userfun2=userf2(t,kmrgd,beta1,beta2,CC,DD)
userfun2=beta2-0.5;
%------------------------------------------------
```

Chapter 2

Local bifurcation analysis

We consider (1.3) at some fixed parameter values and assume that it has a fixed point x_0. If the Jacobian matrix A of (1.3) at x_0 has no eigenvalue λ with $|\lambda| = 1$, then x_0 is called a hyperbolic fixed point. In this case, the dynamics near x_0 is topologically equivalent to that of the linear map $x \longmapsto Ax$ (Grobman - Hartman Theorem). If eigenvalues with $|\lambda| = 1$ are present then x_0 is called a nonhyperbolic fixed point. In this case, the Center Manifold Theorem [101, 50] guarantuees the existence of stable, unstable and center manifolds near the fixed point. The center manifold is an invariant manifold of the a map which is tangent at the fixed point to the eigenspace of the neutrally stable eigenvalues. We determine the reduced dynamics on the center manifold, study its stability and then conclude about the stability of the original system. This theory combined with the normal form approach of Poincaré was used extensively to study parameterized dynamical systems exhibiting bifurcations [49, 62]. On the stable and unstable manifolds, the local dynamics is still determined by the linear part of the map. In contrast, the dynamics in the center manifold depends on both linear and nonlinear terms. Not all nonlinear terms are equally important, since some of them can be eliminated by an appropriate smooth coordinate transformation that puts the map restricted to its center manifold into a normal form.

Assuming sufficient smoothness of g, we write the Taylor expansion of g about (x_0, α_0)

$$\begin{aligned} g(x_0+x,\alpha_0+\alpha) &= x_0 + Ax + \tfrac{1}{2}B(x,x) + \tfrac{1}{6}C(x,x,x) \\ &\quad + \tfrac{1}{24}D(x,x,x,x) + \tfrac{1}{120}E(x,x,x,x,x) \\ &\quad + J_1\alpha + \tfrac{1}{2}J_2(\alpha,\alpha) \\ &\quad + A_1(x,\alpha) + \tfrac{1}{2}B_1(x,x,\alpha) \\ &\quad + \tfrac{1}{6}C_1(x,x,x,\alpha) + \tfrac{1}{24}D_1(x,x,x,x,\alpha) \\ &\quad + \tfrac{1}{2}A_2(x,\alpha,\alpha) + \tfrac{1}{4}B_2(x,x,\alpha,\alpha) + \tfrac{1}{12}C_2(x,x,x,\alpha,\alpha) \\ &\quad + \dots, \end{aligned} \tag{2.1}$$

where all functions are multilinear forms of their arguments and the dots denote higher order terms in x and α. In particular, $A = g_x(x_0,\alpha)$ and the components of the multilinear functions B and C are given by

$$B_i(u,v) = \sum_{j,k=1}^{n} \frac{\partial^2 g_i(x_0,\alpha_0)}{\partial x_j \partial x_k} u_j v_k, \quad C_i(u,v,w) = \sum_{j,k,l=1}^{n} \frac{\partial^3 g_i(x_0,\alpha_0)}{\partial x_j \partial x_k \partial x_l} u_j v_k w_l, \tag{2.2}$$

for $i = 1, 2, \dots, n$. From now on, I_n is the unit $n \times n$ matrix and $\|x\| = \sqrt{\langle x, x\rangle}$, where $\langle u, v\rangle = \bar{u}^T v$ is the standard scalar product in $\mathbb{C}^n$ (or $\mathbb{R}^n$).

This chapter starts with the bifurcation analysis of fixed points of maps and their implementation in CL_MATCONTM. We proceed with the computation of the secondary branch of fixed points emanating from a branch point followed by a detailed study of the normal form coefficients of codim 1 bifurcation points. The chapter ends with a bifurcation study of a truncated normal form map.

Parts of this chapter were published or accepted in [40].

2.1 Bifurcation analysis of codim 1 bifurcations of maps

There are three generic codim 1 bifurcations that can be detected along a curve of fixed points of the K-th iterate, namely NS, LP and NS. Also, there can be branch points BP. We consider them in this order. To detect these singularities, we define 4 test functions.

$$\phi_1(x,\alpha) = \det(A \odot A - I_m), \tag{2.3}$$

$$\phi_2(x,\alpha) = \det(A + I_n), \tag{2.4}$$

$$\phi_3(x,\alpha) = v_{n+1}, \tag{2.5}$$

$$\phi_4(x,\alpha) = \det \begin{pmatrix} F_X \\ v^T \end{pmatrix}. \tag{2.6}$$

Here v is the tangent vector along the curve, $\odot$ is the bialternate matrix product where $m = \frac{n(n-1)}{2}$ (cf. [38], §4.4.4), $F(X) = g(x,\alpha) - x$ and A is the Jacobian matrix of g.

The following codimension 1 bifurcations and branch points can be detected and located as regular zeroes of the above test functions:

- NS: $\phi_1 = 0$.
- PD: $\phi_2 = 0$.
- LP: $\phi_3 = 0, \phi_4 \neq 0$.
- BP: $\phi_4 = 0$.

We notice that ϕ_1 is also zero if there is a pair of real multipliers with product 1. Such points are called neutral saddles. We have to take care of these when processing the NS points. The singularity matrix is :

$$S = \begin{pmatrix} 0 & - & - & 0 \\ - & 0 & - & - \\ - & - & 0 & 1 \\ - & - & - & 0 \end{pmatrix}. \tag{2.7}$$

2.1.1 Fixed point initializations

Numerical continuation of a curve of fixed points (1.4), starts from an initial point along with its tangent vector on the fixed point curve. These data are provided by the routines, initializers, that correspond to each continuation curve. The fixed point initializers are *init_FP_FP.m, init_PD_FP2.m* and *init_BP_FP.m*. To start from a known fixed point of the K-th iterate one first gives the following curve initializer statement:

$$[x0, v0] = init\_FP\_FP(@mapfile, x, p, ap, K).$$

Here *mapfile* is the mapfile to be used, *x* is a vector containing the starting values of the state variables, *p* is the vector containing the starting values of the parameters and *ap* is the index of the active parameter. This routine stores its output partly in the global structure *fpmds*. The output of *init_FP_FP* contains a vector x_0 with the state variables and the active parameter and an empty vector v_0.
To explain the meaning of *fpmds* we run the fixed point initializer using the *mapfile* that is defined in §1.3.7 where symbolic derivatives are used. The global structure *fpmds* is set using:

$$[x0, v0] = init\_FP\_FP(@Tnfmap, [0; 0], [-1; 0; 1; 1], 2, 1)$$

Some important fields of *fpmds* are given by:

```
           P0: [4x1 double]
 ActiveParams: 2
      mapfile: @Tnfmap
         func: @fun_eval
     Jacobian: @jacobian
    JacobianP: @jacobianp
     Hessians: @hessians
    HessiansP: @hessiansp
         Der3: @der3
         Der4: @der4
         Der5: @der5
  Niterations: 1
       nphase: 2
```

To start the continuation of $2K$-cycles from a period-doubling point detected during a fixed point of K-th iterate continuation one first gives the following curve initializer statement:

$$[x0, v0] = init\_PD\_FP2(@mapfile, xnew, p, ap, s(j), h, K)$$

Here *mapfile* is the mapfile to be used, *xnew* is a vector containing the starting values of the state variables, *p* is the vector containing the starting values of the parameters and *ap* must be the index of the active parameter. In the most natural situation where *x* is the vector returned by the previous fixed point curve continuation one starts to build *xnew* by the statement *xnew=x(1:nphase,s(j).index)*.
s(j) is the special point structure of the detected period doubling point on the fixed point of K-th iterate curve continuation and *nphase* is the number of state variables.

Next, the statement $p(ap\_old) = x(end, s(j).index);$ replaces the old value of the free parameter in the previous run by the parameter value at the PD point.
The output of *init_PD_FP2* contains a vector x_0 with the state variables and the active parameter and a tangent vector v_0. If x_{PD} is the PD point on the original branch and q is the right eigenvector of the multiplier -1 in x_{PD} then $x_0 = x_{PD} + hq$ and $v_0 = q$; the scalar h is called the amplitude. The routine *init_PD_FP2* stores its output partly in the global structure *fpmds*.

2.1.2 Output of a fixed point continuation

The fixed point curve is continued by calling:

$$[x, v, s, h, f] = cont(@fixedpointmap, x0, v0, opt)$$

This call returns : *x* and *v*: points and their tangent vectors along the fixed point curve, respectively.
The array *s* contains information about the computed singular points, including zeros of user functions, with the following fields:

s.index index of the point in *x*.

s.label label of the singularity, may be *00*, NS, PD, LP, BP, *99* or the label of a user function.The strings *00* and *99* indicate the first and the last point on the fixed point curve, respectively.

s.data extra information.
For the first and last points this is only an empty tangent vector. For zeroes of user functions an empty tangent vector is given, plus the values of

all active user functions and test functions. For bifurcation points see the respective cases in §2.3.1, §2.3.2, §2.3.3, §2.2.

s.msg a string containing a message for this particular singularity. For the first and last points these are the strings 'This is the first point of the curve' and 'This is the last point of the curve', respectively. For zeroes of user functions the name of the user function is given. For bifurcation points see the respective cases.

h and *f* were described in §1.3.2

2.2 Branch switching at a branch point

In this section we consider the approximation of a new cycle curve that emanates from a branch point BP for (1.4). The same algorithm is used to switch at a PD point for the period-K cycle to the period-$2K$ cycle, since it corresponds to a branch point for $f^{(2K)}(x,\alpha) - x = 0$. The method is similar to that for branch points of equilibria and is presented here only for completeness; it is also used in CONTENT.

A solution $X_0 = X(s_0)$ of

$$F(X) = g(x,\alpha) - x = 0, \tag{2.8}$$

is called a *simple singular point* if $F_X(X_0)$ has rank $n-1$. For system (2.8), we have $F_X^0 = [g_x(x_0,\alpha_0) - I_n, g_\alpha(x_0,\alpha_0)]$, and $X_0 = (x_0,\alpha_0)$ is a simple singular point if and only if, either

$$\dim N(g_x(x_0,\alpha_0) - I_n) = 1, \;\; g_\alpha(x_0,\alpha_0) \in R(g_x(x_0,\alpha_0) - I_n),$$

or

$$\dim N(g_x(x_0,\alpha_0) - I_n) = 2, \;\; g_\alpha(x_0,\alpha_0) \notin R(g_x(x_0,\alpha_0) - I_n).$$

The first case is a codimension 2 situation, the second case has codimension 4, so in practice we only expect the first case.

Suppose we have a solution branch $X(s)$ and let $X_{s_0} = (x_0,\alpha_0)$ be a simple singular point. Then $N(F_X^0)$ is two-dimensional and can be written as

$\operatorname{span}\{\phi_1, \phi_2\}$ where $\phi_1, \phi_2 \in \mathbb{R}^{n+1}$ are linearly independent. Also, $N([F_X^0]^T)$ is one-dimensional and is spanned by a vector $\psi \in \mathbb{R}^n$. Let F_{YY}^0 be the bilinear form in the Taylor expansion of F about X_0. If $Y(s)$ is any solution branch of (2.8) with $Y(s_0) = X_0$, then $Y_s(s_0)$ can be written as $Y_s(s_0) = \alpha\phi_1 + \beta\phi_2$ for some $\alpha, \beta \in \mathbb{R}$. Differentiating the identity $F(Y(s)) = 0$ twice and computing the scalar product with ψ at s_0, we get

$$\langle \psi, F_{YY}^0(\alpha\phi_1 + \beta\phi_2)(\alpha\phi_1 + \beta\phi_2)\rangle = 0,$$

or, equivalently,

$$c_{11}\alpha^2 + 2c_{12}\alpha\beta + c_{22}\beta^2 = 0, \tag{2.9}$$

where $c_{jk} = \langle \psi, F_{YY}^0\phi_j\phi_k\rangle$ for $j, k = 1, 2$.

Equation (2.9) is called the *algebraic bifurcation equation* (*ABE*). The case $c_{12}^2 - c_{11}c_{22} < 0$ is impossible, since at least one branch goes through X_0. Thus, generically, $c_{12}^2 - c_{11}c_{22} > 0$, and equation (2.9) has two real nontrivial, independent solution pairs, (α_1, β_1) and (α_2, β_2), which are unique up to scaling. In this case we have a *simple branch point*, where two distinct branches pass through X_0.

The above procedure allows one to compute the normalized tangent vectors $Y_{1s}(s_0), Y_{2s}(s_0)$ of the two branches that pass through X_0. Now if

$$|\langle Y_{1s}(s_0), X_s(s_0)\rangle| < |\langle Y_{2s}(s_0), X_s(s_0)\rangle|,$$

then we conclude that $Y_{1s}(s_0)$ is the tangent vector to the new branch; otherwise, $Y_{2s}(s_0)$ is the tangent vector.

2.3 Normal form coefficients of codim 1 bifurcation points

When a limit point, period doubling point or Neimark-Sacker point is detected on a curve of fixed points, then the processing of these points includes computation of the corresponding normal form coefficients. What follows is the normal form analysis of the codim 1 bifurcation points.

2.3.1 Limit point

At a fold point the matrix A of (1.3) (Jacobian of the K-th iterate) has a simple eigenvalue $\lambda_1 = 1$ and no other multipliers on the unit circle, while the restriction

of (1.3) to a one dimensional center manifold at the critical parameter value has the form

$$w \longmapsto w + aw^2 + \mathcal{O}(w^3), w \in \mathbb{R} \tag{2.10}$$

if $a \neq 0$, where for the coefficient a we have the expression:

$$a = \frac{1}{2} \langle p, B(q,q) \rangle , \tag{2.11}$$

where $Aq = q, \quad A^T p = p$, and $\langle q, q \rangle = 1$, $\langle p, q \rangle = 1$.
A generic unfolding of (2.10) is

$$w \longmapsto \alpha + w + aw^2 + \mathcal{O}(w^3), w \in \mathbb{R} \tag{2.12}$$

where α is the control parameter with critical value 0. When the control parameter crosses 0, two fixed points collide and disappear. So the fixed point curve has a turning point with respect to the control parameter.

2.3.2 Period doubling

At a PD point the matrix A has a simple eigenvalue $\lambda_1 = -1$ and no other multipliers on the unit circle. The restriction of (1.3) to a one dimensional center manifold at the critical parameter value can be transformed to the normal form

$$w \longmapsto -w + bw^3 + \mathcal{O}(w^4), w \in \mathbb{R} \tag{2.13}$$

if $b \neq 0$, where b is given by

$$b = \frac{1}{6} \left\langle p, C(q,q,q) + 3B(q, (I - A)^{-1} B(q,q)) \right\rangle , \tag{2.14}$$

where I is the unit $n \times n$ matrix, $Aq = -q, A^T p = -p$, $\langle q, q \rangle = 1$,$\langle p, q \rangle = 1$.
A generic unfolding of (2.13) is

$$w \longmapsto -w(1 + \alpha) + bw^3 + \mathcal{O}(w^4), w \in \mathbb{R} \tag{2.15}$$

where α is a control parameter. When the control parameter crosses the critical value 0, a cycle of period 2 bifurcates from the fixed point. If $b > 0$ then this

period two cycle is stable and found for those α - values where the fixed point of the map is unstable; this is called a supercritical PD. If $b < 0$ then the period two cycle is unstable and found for those α - values where the fixed point of the map is stable; this is called a subcritical PD point.

2.3.3 Neimark-Sacker

At a NS point the matrix A has simple critical multipliers $\lambda_{1,2} = e^{\pm i\theta_0}$ ($0 \leq \theta \leq \pi$) and no other multipliers on the unit circle. Assume that $e^{ik\theta_0} \neq 1, k = 1, 2, 3, 4$ (these special cases are the *strong resonances*). Then the restriction of (1.3) to the two dimensional center manifold at the critical parameter value can be transformed to the normal form

$$w \longmapsto we^{i\theta_0}(1 + d|w|^2) + \mathcal{O}(|w|^4), w \in \mathbb{C}$$

where w is now a complex variable and d is a complex number. If $c = Re(d) \neq 0$, then a unique closed invariant curve around the fixed point appears when the parameter crosses the critical value. One has the following expression for d:

$$d = \frac{1}{2}e^{-i\theta_0}\left\langle p, C(q, q, \bar{q}) + 2B(q, h_{11}) + B(\bar{q}, h_{20})\right\rangle, \tag{2.16}$$

where

$$h_{11} = (I_n - A)^{-1}B(q, \bar{q}), \;\; h_{20} = (e^{2i\theta_0}I_n - A)^{-1}B(q, q),$$

and $Aq = e^{i\theta_0}q, \; A^T p = e^{-i\theta_0}p$ and $\langle q, q\rangle = \langle p, q\rangle = 1$.
If $c < 0$ then a stable invariant curve branches off the NS point and is found for values of the control parameter for which the fixed point of the map is unstable. If $c > 0$ then an unstable invariant curve branches off the NS point and is found for values of the control parameter for which the fixed point of the map is stable.

2.4 Example: A truncated normal form map

2.4.1 The map and some analytical normal form coefficients

In this example we consider the two - dimensional map, introduced in [62], §9.9, (unfolding of an R2 point to which it reduces for $\beta_1 = \beta_2 = 0$)

$$M_{TN} : \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} \mapsto \begin{pmatrix} -1 & 1 \\ \beta_1 & -1+\beta_2 \end{pmatrix} \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} + \begin{pmatrix} 0 \\ C\xi_1^3 + D\xi_1^2\xi_2 \end{pmatrix}. \tag{2.17}$$

For all parameter values, this map has a trivial fixed point $(0,0)^T$. If (ξ_1, ξ_2) is a nontrivial fixed point then we have:

$$\xi_2 = 2\xi_1, \quad \xi_2 = \beta_1\xi_1 + (-1+\beta_1)\xi_2 + C\xi_1^3 + D\xi_1^2\xi_2. \tag{2.18}$$

It is easy to see that if

$$\frac{4-(\beta_1+2\beta_2)}{C+2D} > 0,$$

then nontrivial real fixed points $(\xi_1, 2\xi_1)$ exist and are given by

$$\xi_1 = \pm\sqrt{\frac{4-(\beta_1+2\beta_2)}{C+2D}}, \xi_2 = 2\xi_1. \tag{2.19}$$

If

$$\frac{4-(\beta_1+2\beta_2)}{C+2D} = 0,$$

then these points collide with the trivial fixed point. If this happens with β_1 or β_2 as a free parameter in a continuation of trivial fixed points, then clearly we have a pitchfork bifurcation of fixed points.
The characteristic equation of the Jacobian in the trivial fixed point is:

$$\lambda^2 + (2-\beta_2)\lambda + 1 - \beta_1 - \beta_2 = 0. \tag{2.20}$$

We first note that the product of the two multipliers is 1 if and only if $\beta_1 + \beta_2 = 0$. In particular, NS points can only be found if $\beta_1 + \beta_2 = 0$. In this case, $\Delta = (2-\beta_2)^2 - 4 = \beta_2(\beta_2 - 4)$. So we have true NS points if $\beta_2 \in]0,4[, \beta_1 = -\beta_2$; we have neutral saddles if $\beta_2 \notin [0,4], \beta_1 = -\beta_2$.

In particular we consider the following three special cases of the NS bifurcation:

$$\begin{array}{ll} (i) & \beta_1 = -1, \beta_2 = 1, (\theta = \frac{2\pi}{3}) \\ (ii) & \beta_1 = -2, \beta_2 = 2, (\theta = \frac{\pi}{2}) \\ (iii) & \beta_1 = -3, \beta_2 = 3, (\theta = \frac{\pi}{3}) \end{array} \tag{2.21}$$

We note that that cases (i) and (ii) are cases with a strong resonance.

Also, it is easy to see that (2.20) has a root -1 if and only if $\beta_1 = 0$. The other root then is $-1 + \beta_2$. We will also consider the case :

$$(iv)\beta_1 = 0, \beta_2 = 1. \tag{2.22}$$

One can obtain analytically the normal form coefficients. The results are as follows:

- in the case of (i), i.e. $\theta = \frac{2\pi}{3}$: $c = -\frac{1}{8}(6C + 4D)$
- in the case of (ii), i.e. $\theta = \frac{\pi}{4}$: $c = -\frac{1}{12}(6C + 6D)$
- in the case of (iii), i.e. $\theta = \frac{\pi}{3}$: $c = -\frac{1}{16}(6C + 8D)$
- in the case of (iv), i.e. $\theta = \pi$: $b = -C$

2.4.2 Numerical continuation of fixed points

Theoretically computed values of the normal form coefficients can now be checked numerically when continuing the fixed point curve. In the $mapfile$ (cf. §1.3.7) the order of state variables and parameters is (ξ_1, ξ_2) and (β_1, β_2, C, D), respectively. For illustration purposes we defined two user functions, namely $\beta_2 - 2$ with label $'B2\ '$ and $\beta_2 - 0.5$ with label $'B3\ '$.

First we continue the fixed point curve numerically to detect the NS point in case (i) in *Run 1*, where we use the mapfile that uses symbolic derivatives.

```
>  global opt cds fpmds
>> ap=2; p=[-1;0;1;1];
>> opt = contset;
>> [x0,v0]=init_FP_FP(@Tnfmap,[0;0], p, ap);
>> opt=contset;opt=contset(opt,'MaxNumPoints',50);
>> opt=contset(opt,'Singularities',1);
>> opt = contset(opt,'Multipliers',1);
>> [x1,v1,s1,h1,f1]=cont(@fixedPointmap,x0,[],opt);
first point found
tangent vector to first point found
label = B3  , x = ( 0.000000 0.000000 0.500000 )
```

```
label = NS  , x = ( 0.000000 0.000000 1.000000 )
normal form coefficient of NS = -1.250000e+000
label = B2  , x = ( 0.000000 0.000000 2.000000 )
label = BP  , x = ( 0.000000 0.000000 2.500000 )
elapsed time  = 0.7 secs
npoints curve = 50
```

Our theoretical outcome is confirmed by the numerical value for d obtained in *Run 1*. Indeed, in this case the theoretically obtained value of the normal form coefficient c is

$$c = -\frac{1}{8}(6C + 4D) = -1.25,$$

since $C = D = 1$.

By (2.19) the nontrivial fixed points collide with the trivial fixed point when

$$4 - (\beta_1 + 2\beta_2) = 0, \tag{2.23}$$

The fixed parameter in *Run 1* is $\beta_1 = -1$, this implies that in a BP $\beta_2 = 2.5$ in (2.23). This confirms the result in *Run 1* concerning the BP point.

The Jacobian is given by:

$$[(M_{TN})_x - I|(M_{TN})_{\beta_2}] = \begin{pmatrix} -2 & 1 & 0 \\ \beta_1 & -2 + \beta_2 & 0 \end{pmatrix}. \tag{2.24}$$

If $\beta_1 = -1$ and $\beta_2 = 2.5$, then this reduces to:

$$[(M_{TN})_x - I|(M_{TN})_{\beta_2}] = \begin{pmatrix} -2 & 1 & 0 \\ -1 & 0.5 & 0 \end{pmatrix}. \tag{2.25}$$

Clearly $[(M_{TN})_x - I|(M_{TN})_{\beta_2}]$ is rank deficient as expected.

Now we compute the new branch in the BP point of *Run 1*; we refer to this as *Run 2*:

```
>> global x1 v1 s1  opt cds  fpmds
>> opt = contset;
>> opt = contset(opt,'Multipliers',1);
>>>>> Branch switching at BP >>>>>>>
>> xx2=x1(1:2,s1(3).index);p1=p;
```

```
>> p1(fpmds.ActiveParams)=x1(3,s1(3).index);
>> opt=contset(opt,'backward',0);
>>opt=contset(opt,'MaxNumPoints',50);
>>[x2,v2]=init_BP_FP(@Tnfmap,xx2,p1,s1(3),0.01);
>>[x21,v21,s21,h21,f21]=cont(@fixedPointmap,xx2,[],opt);
first point found
tangent vector to first point found
label = PD, x = ( 0.377964 0.755929 2.285714 )
normal form coefficient of PD = 4.392157e+000
elapsed time  = 0.6 secs
npoints curve = 50
>> cpl(x21,v21,s21,[3 1]);
>>opt=contset(opt,'backward',1);;
>> [x22,v22,s22,h22,f22]=cont(@fixedPointmap,x2,[],opt);
first point found
tangent vector to first point found
label = BP, x = ( -0.000000 -0.000000 2.500000 )
label = PD, x = ( -0.377964 -0.755929 2.285714 )
normal form coefficient of PD = 4.392157e+000
elapsed time  = 0.9 secs
npoints curve = 50
>> cpl(x22,v22,s22,[3 1])
```

The branch in *Run 2* is a nontrivial one and we remark that for the singular points $\xi_2 = 2\xi_1$ holds. In fact the curve of nontrivial fixed points in (2.19) in (β_2, ξ_1) space is a parabola. A picture of the continued trivial fixed points of *Run 1* and nontrivial fixed points computed in *Run 2* is given in Figure 2.1.

In *Run 3* we continue a fixed point curve to detect the NS point in case (ii) :

```
>>global opt cds fpmds
>>ap=2; p=[-2;0;1;1];
>>opt = contset;
>>[x0,v0]=init_FP_FP(@Tnfmap,[0;0], p, ap,1);
>>opt=contset;opt=contset(opt,'MaxNumPoints',300);
>>opt=contset(opt,'Singularities',1);
>>opt=contset(opt,'Backward',0);
```

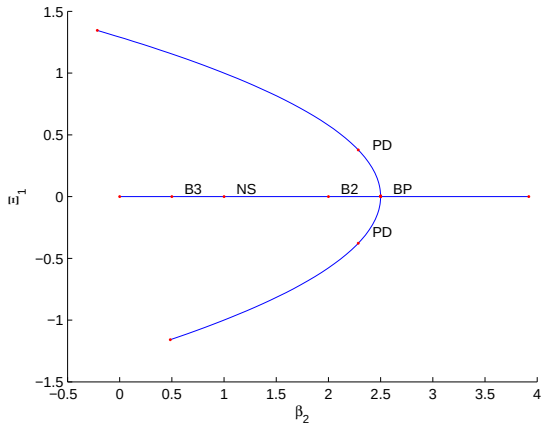


Figure 2.1: Continuation of trivial and nontrivial fixed point of M_{TN} in (β_2, ξ_1) space .

```
>>opt = contset(opt,'Multipliers',1);
>>[x3,v3,s3,h3,f3]=cont(@fixedPointmap,x0,[],opt);
first point found
tangent vector to first point found
label = NS  , x = ( 0.000000 0.000000 2.000000 )
normal form coefficient of NS = -1.000000e+000
label = BP  , x = ( 0.000000 0.000000 3.000000 )
elapsed time  = 1.1 secs
npoints curve = 300
>> cpl(x3,v3,s3,[3 1])
```

Again the numerically obtained value for the normal form coefficient d in *Run 3* confirms the theoretical result. Indeed, the theoretical value is

$$d = -\frac{1}{12}(6C + 6D) = -1,$$

since $C = D = 1$.
By (2.19) we have a BP point if $4 - (\beta_1 + 2\beta_2) = 0$. By subsituting $\beta_1 = -2$, we get $\beta_2 = 3$. This confirms the result in *Run 3* concerning the BP point.

Now we perform branch switching in this point in *Run 4*:

```
>>global x3 v3 s3 opt cds  fpmds
>>opt = contset;
>>xx2=x3(1:2,s3(3).index);p1=p;
>>p1(fpmds.ActiveParams)=x3(3,s3(3).index);
>>opt=contset(opt,'backward',0);
>>opt=contset(opt,'MaxNumPoints',300);
>>[x2,v2]=init_BP_FP(@Tnfmap,xx2,p1,s3(3),0.001);
>>opt = contset(opt,'Multipliers',1);
>>[x41,v41,s41,h41,f41]=cont(@fixedPointmap,xx2,[],opt);
first point found
tangent vector to first point found
label = PD, x = ( 0.534523 1.069045 2.571429 )
normal form coefficient of PD = 3.733333e+000
elapsed time  = 1.4 secs
npoints curve = 300
>>cpl(x41,v41,s41,[3 1 2]);
>>opt=contset(opt,'backward',1);;
>>[x42,v42,s42,h42,f42]=cont(@fixedPointmap,x2,[],opt);
first point found
tangent vector to first point found
label = BP, x = ( -0.000000 -0.000000 3.000000 )
label = PD, x = ( -0.534523 -1.069045 2.571429 )
normal form coefficient of PD = 3.733333e+000
elapsed time  = 1.5 secs
npoints curve = 300
>> cpl(x42,v42,s422,[3 1 2])
```

The BP point is the same as in *Run 3*, and the PD point on the new branch satisfies $\xi_2 = 2\xi_1$. A picture of the new branch computed in *Run 4* is given in Figure 2.2.

In *Run 5* we continue a fixed point curve to detect the NS point in case (iii) :

```
>>global opt cds fpmds
>>ap=2; p=[-3;0;1;1];
>>opt = contset;
>>[x0,v0]=init_FP_FP(@Tnfmap,[0;0], p, ap,1);
```

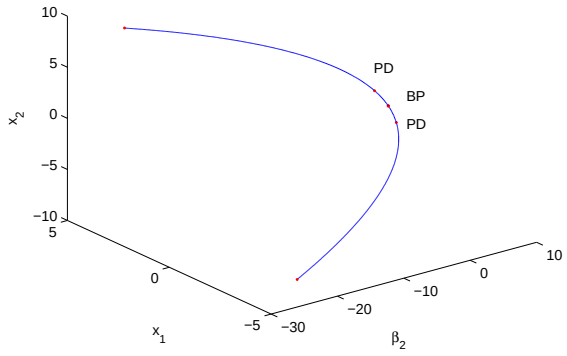


Figure 2.2: The fixed point curve of the second iterate in (β_2, x_1, x_2) space.

```
>>opt=contset;opt=contset(opt,'MaxNumPoints',300);
>>opt=contset(opt,'Singularities',1);
>>opt = contset(opt,'Multipliers',1);
>>[x5,v5,s5,h5,f5]=cont(@fixedPointmap,x0,[],opt);
first point found
tangent vector to first point found
label = NS, x = ( 0.000000 0.000000 3.000000 )
normal form coefficient of NS = -8.750000e-001
elapsed time  = 1.4 secs
npoints curve = 300
>> cpl(x5,v5,s5,[3 1])
```

Here also the numerically obtained value confirms the theoretical result $c = -\frac{1}{16}(6C + 8D) = -0.875$ where $C = D = 1$.

Since the normal form coefficient in *Run 5* is negative, the invariant curves nearby the NS point must be stable. In fact the characteristic polynomial (2.20) when $\beta_1 = -3$, is

$$\lambda^2 + (2 - \beta_2)\lambda + 4 - \beta_2 = 0. \tag{2.26}$$

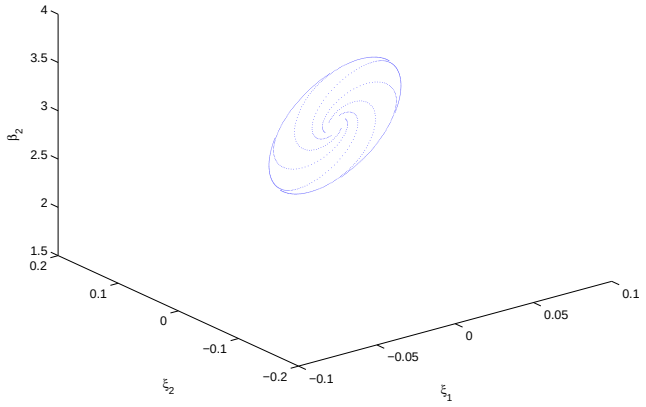


Figure 2.3: Stable invariant curve of M_{TN} started from $\xi_1 = \xi_2 = 0.01$ for $\beta_2 = 2.99$.

The multipliers for β_2 nearby 3 are

$$\lambda_{1,2} = -\frac{\beta_2 - 2}{2} \pm i\sqrt{3 - \frac{\beta_2^2}{4}}.$$

Also

$$|\lambda_{1,2}^2| = 4 - \beta_2.$$

So the fixed point of M_{TN} is stable for $\beta_2 > 3$ and unstable for $\beta_2 < 3$, i.e. the invariant curve is stable when $\beta_2 < 3$ and unstable when $\beta_2 > 3$. A picture of the stable invariant curve nearby the NS point is given in Figure 2.3. It was created by simulation of M_{TN} for the parameter values indicated in Figure 2.3.

The next *Run 6* will detect the PD point in case (iv):

```
>>global  opt cds fpmds
```

```
>>ap=1; p=[-1;1;1;1];
>>opt = contset;
>>[x0,v0]=init_FP_FP(@Tnfmap,[0;0], p, ap,1);
>>opt=contset;opt=contset(opt,'MaxNumPoints',300);
>>opt=contset(opt,'Singularities',1);
>>opt = contset(opt,'Multipliers',1);
>>[x6,v6,s6,h6,f6]=cont(@fixedPointmap,x0,[],opt);
first point found
tangent vector to first point found
label = NS, x = ( 0.000000 0.000000 -1.000000 )
normal form coefficient of NS = -1.250000e+000
label = PD, x = ( 0.000000 0.000000 0.000000 )
normal form coefficient of PD = -1
label = BP, x = ( 0.000000 0.000000 2.000000 )
elapsed time  = 1.3 secs
npoints curve = 300
```

Clearly the result of the continuation in *Run 6* is consistent with the theoretical statement for case (iv), that is $b = -2C = -2$ since $C = 1$.

By (2.19) we have a BP point when $4 - (\beta_1 + 2\beta_2) = 0$. Since $\beta_2 = 1$ in *Run 6*, the BP point must be found for $\beta_1 = 2$. This confirms the result in *Run 10* concerning the BP point.

Now we compute the curve of fixed points of the second iterate in the PD point of *Run 6*. We call this *Run 7*:

```
>>global x6 v6 s6  opt cds  fpmds
>>opt = contset;
>>>>>  switching at PD >>>>>>>
>>xx2=x6(1:2,s6(3).index);p1=p;
>>p1(fpmds.ActiveParams)=x6(3,s6(3).index);
>>opt=contset(opt,'backward',0);
>>opt=contset(opt,'MaxNumPoints',300);
>>[x2,v2]=init_PD_FP(@Tnfmap,xx2,p1,s6(3),0.01,1);
>>opt = contset(opt,'Multipliers',1);
>>[x7,v7,s7,h7,f7]=cont(@fixedPointmap,x2,[],opt);
first point found
tangent vector to first point found
label = BP, x = ( -0.000000 0.000000 -0.000000 )
```

```
label = NS, x = ( -0.577350 0.000000 -0.333333 )
Neutral saddle
label = BP, x = ( -0.707107 -0.000000 -0.500000 )
elapsed time  = 2.0 secs
npoints curve = 300
```

The first BP point in *Run 7* is the PD point in *Run 6*. The second BP point corresponds with $\beta_1 = -0.5$ and is clearly not a fixed point of M_{TN}. For the parameters values in *Run 7*, we have:

$$M_{TN} : \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} -1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ 0 \end{pmatrix}, \quad (2.27)$$

and

$$M_{TN} : \begin{pmatrix} \frac{\sqrt{2}}{2} \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} -1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{\sqrt{2}}{2} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ 0 \end{pmatrix}. \quad (2.28)$$

So indeed M_{TN}^2 maps the point $(-\frac{\sqrt{2}}{2}, 0)^T$ to itself.

We further remark that in *Run 10* the trivial fixed point is stable for negative values of β_1 close to 0 and unstable for positive values of β_1 close to zero. Also, the normal coefficient of the PD point is negative.

Further computations show that the multipliers of M_{TN} and M_{TN}^2 in the PD point $x = (0, 0, 0)$ are $(-1, 0)^T$ and $(1, 0)^T$ respectively.

A nearby point on the curve of fixed points of the second iterate is $x = (-0.0129710, 0.0000000, -0.0001683)$. For the same parameter value the fixed point of the map is stable, as could be expected from the sign of the normal form coefficient in the PD point in *Run 6*. The multipliers of M_{TN} and M_{TN}^2 in the same point are $(-1.0003363, 0.0005046)^T$ and $(1.0006728, 0.0000003)^T$ respectively. So the fixed points of the second iterate are unstable, as could also be expected from the sign of the normal form coefficient in the PD point in *Run 6*.

A survey of the runs of the example is given in the Table (2.1).

Run	Continuation	Detected bifurcation(s)
1	Fixed point	B3, NS, B2, BP
2	New branch of fixed points at the BP	PD, BP, PD
3	Fixed point	NS, BP
4	New branch of fixed points at the BP	PD, BP, PD
5	Fixed point	NS
6	Fixed point	NS, PD, BP
7	Fixed point of the second iterate	BP, NS, BP

Table 2.1: A survey of the runs of M_{TN}.

Chapter 3

Continuation of codim 1 bifurcations; branch switching

Once a codim 1 bifurcation has been located, it can be continued in two parameters. We give details on this continuation for curves of fold, period-doubling, and Neimark-Sacker bifurcations of period-K cycles that are computed by the mentioned Gauss-Newton continuation algorithm applied to *minimally extended defining systems*, cf. [38]. These systems were first implemented, together with the *standard extended defining systems*, in CONTENT [44]. We have adopted in CL_MATCONTM the most robust and efficient methods tested there.

Near a codim 2 bifurcation point there are often other branches of codim 1 bifurcation points and it is, therefore, highly desirable to be able to switch from one bifurcation curve to another one and then follow the new branch. The key problem here is to find a good approximation of the new curve from information on the old bifurcation curve. Branch switching is based on the predictor-corrector approach. The predictor calculates a good initial guess to an emanating solution curve. Then this initial guess serves as prediction to trace the new branch.

This chapter is outlined as follows. We start by studying the continuation of fold, flip and Neimar-Sacker points and its implementation in CL_MATCONTM.

Then we discuss branch switching followed by the initializer routines corresponding to the different branches. Finally, we present the detection graph indicating all codim 1 and codim 2 bifurcation points and their interconnections as well as switching graphs demonstrating possible switching at codim 1 and codim 2 bifurcation points.

Parts of this chapter were published in [42].

3.1 Continuation of fold and flip curves

The *limit point curve* and *period-doubling curve* are both defined by the following system

$$\left\{ \begin{array}{rcl} g(x,\alpha) - x & = & 0, \\ s(x,\alpha) & = & 0, \end{array} \right. \tag{3.1}$$

where $(x,\alpha) \in \mathbb{R}^{n+2}$, g is given by (1.3), while s is obtained by solving one of the algebraic systems

$$\left(\begin{array}{cc} g_x(x,\alpha) \mp I_n & w_{bor} \\ v_{bor}^T & 0 \end{array} \right) \left(\begin{array}{c} v \\ s \end{array} \right) = \left(\begin{array}{c} 0_n \\ 1 \end{array} \right), \tag{3.2}$$

where $w_{bor}, v_{bor} \in \mathbb{R}^n$ are chosen such that the matrix in (3.2) is nonsingular. One should take the "$-$" sign in (3.2) for the LP-curve and the "$+$" sign for the PD-curve. If v_{bor} is close to the nullvector of $g_x(x,\alpha) - I_n$ and $g_x(x,\alpha) + I_n$ on the LP- and PD-curve, respectively, and w_{bor} is close to the nullvector of $(g_x(x,\alpha) - I_n)^T$ and $(g_x(x,\alpha) + I_n)^T$ on the LP- and PD-curve, respectively, then the matrix in (3.2) is nonsingular at (x,α). In practical computations, v_{bor} and w_{bor} are the nullvecrors of $g_x \mp I$ and $(g_x \mp I)^T$, respectively, on the corresponding fold and flip curves.

The derivatives of s can be obtained easily from the derivatives of $g_x(x,\alpha)$:

$$s_z = -w^T (g_x)_z v, \tag{3.3}$$

where z is a state variable or an active parameter and w is obtained by solving

$$\left(\begin{array}{cc} g_x^T(x,\alpha) \mp I_n & v_{bor} \\ w_{bor}^T & 0 \end{array} \right) \left(\begin{array}{c} w \\ s \end{array} \right) = \left(\begin{array}{c} 0_n \\ 1 \end{array} \right). \tag{3.4}$$

We note that the quantities called s in (3.2) and (3.4) are the same since they are both equal to the bottom right element of the inverse of the square matrix in (3.2). The the resonance cases of 1:1 (R1) and 1:2 (R2) where g_x has a double eigenvalue $+1$ and -1 respectively, deserve special attention. We can make s in (3.1) dependent on a new parameter λ if we consider the bordered system

$$\begin{pmatrix} g_x(x,\alpha) \mp \lambda I_n & w_{bor} \\ v_{bor}^T & 0 \end{pmatrix} \begin{pmatrix} v \\ s \end{pmatrix} = \begin{pmatrix} 0_n \\ 1 \end{pmatrix}, \tag{3.5}$$

Then in the case of R1 the Jacobian matrix g_x has a double eigenvalue $\lambda = 1$ iff both $g(x,\alpha,\lambda)$ and $g_\lambda(x,\alpha,\lambda)$ vanish for $\lambda = 1$. Similarly, in the case of R2 the Jacobian matrix g_x has a double eigenvalue $\lambda = -1$ iff both $g(x,\alpha,\lambda)$ and $g_\lambda(x,\alpha,\lambda)$ vanish for $\lambda = -1$.

Derivatives of s with respect to λ can be computed by differentiating (3.5). Then

$$\begin{pmatrix} g_x(x,\alpha) \mp \lambda I_n & w_{bor} \\ v_{bor}^T & 0 \end{pmatrix} \begin{pmatrix} v_\lambda \\ g_\lambda \end{pmatrix} + \begin{pmatrix} \mp I_n & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} v \\ s \end{pmatrix} = \begin{pmatrix} 0_n \\ 0 \end{pmatrix}, \tag{3.6}$$

v_λ, s_λ can be found by solving the system

$$\begin{pmatrix} g_x(x,\alpha) - \lambda I_n & w_{bor} \\ v_{bor}^T & 0 \end{pmatrix} \begin{pmatrix} v_\lambda \\ s_\lambda \end{pmatrix} = \begin{pmatrix} v \\ 0 \end{pmatrix}, \tag{3.7}$$

Multiplying (3.7) from the left by (w^T, s) and setting $\lambda = 1$ gives

$$s_\lambda = w^T v. \tag{3.8}$$

We use (3.8) to define the test function for $R1$ on a LP-curve.

By multiplying (3.7) from the left by (w^T, s) and setting $\lambda = -1$ we obtain

$$g_\lambda = -w^T v. \tag{3.9}$$

We use (3.9) to define the test function for $R2$ on a PD-curve.

We remark that our formulas (3.8) and (3.9) are consistent with the well known result from linear algebra that a geometrically simple eigenvalue is algebraically double if and only if the associated left and right eigenvectors are orthogonal.

3.1.1 Bifurcations and test functions for LP curves

There are four generic codim 2 bifurcations that can be detected along the limit point curve:

- *1:1 resonance*. We will denote this bifurcation with R1
- *Fold+Flip* point, denoted as LPPD
- *Fold+Neimark-Sacker* point, denoted as LPNS
- *Cusp* point, denoted as CP

To detect these singularities, we first define 4 test functions:

- $\phi_1 = w^T v$.
- $\phi_2 = det(A(x,\alpha) + I_n)$.
- $\phi_3 = \det(A \odot A - I_m)$.
- $\phi_4 = \langle w, B(v,v)\rangle$.

In these expressions $A = g_x(x,\alpha)$, v and w are the vectors computed in (3.2) and (3.4), respectively. The singularity matrix is:

$$S = \begin{pmatrix} 0 & - & 0 & - \\ - & 0 & - & - \\ 1 & - & 0 & - \\ - & - & - & 0 \end{pmatrix}. \tag{3.10}$$

3.1.2 Bifurcations and test functions for PD curves

In discrete maps there are four generic codim 2 bifurcations that can be detected along the period doubling curve:

- *1:2 resonance* point, denoted as R2
- *Fold+Flip* point, denoted as LPPD

- *Flip+Neimark-Sacker* point, denoted as PDNS
- *Generalized flip* point, denoted as GPD

To detect these singularities, we define 4 test functions:

- $\phi_1 = w^T v$.
- $\phi_2 = det(A(x,\alpha) - I_n)$.
- $\phi_3 = \det(A \odot A - I_m)$.
- $\phi_4 = \langle w, C(v,v,v)\rangle + 3\langle w, B(v,(I_n - A^{-1}B(v,v))\rangle$.

In these expressions v and w are the vectors computed in (3.2) and (3.4), respectively. The singularity matrix is:

$$S = \begin{pmatrix} 0 & - & 0 & - \\ - & 0 & - & - \\ 1 & - & 0 & - \\ - & - & - & 0 \end{pmatrix}. \tag{3.11}$$

3.2 Continuation of NS curves

The *Neimark-Sacker* and *neutral-saddle curves* are defined by the following system

$$\begin{cases} g(x,\alpha) - x &= 0 \\ s_{i_1 j_1}(x,\alpha,\kappa) &= 0 \\ s_{i_2 j_2}(x,\alpha,\kappa) &= 0, \end{cases} \tag{3.12}$$

i.e. by $n+2$ equations for the $(n+3)$ unknowns $x \in \mathbb{R}^n$, $\alpha \in \mathbb{R}^2$, $\kappa \in \mathbb{R}$. Here $(i_1, j_1, i_2, j_2) \in \{1,2\}$ and $s_{i,j}$ are the components of S:

$$S = \begin{pmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{pmatrix},$$

which is obtained by solving

$$\begin{pmatrix} (g_x)^2(x,\alpha) - 2\kappa g_x + I_n & W_{bor} \\ V_{bor}^T & O \end{pmatrix} \begin{pmatrix} V \\ S \end{pmatrix} = \begin{pmatrix} 0_{n,2} \\ I_2 \end{pmatrix}, \tag{3.13}$$

where $V_{bor}, W_{bor} \in \mathbb{R}^{n\times 2}$ are chosen (and can be adapted) so that the matrix in (3.13) is nonsingular. Along the Neimark-Sacker curve, κ is the real part of the critical multipliers $\mathrm{e}^{\pm i\theta}$. The derivatives of s_{ij} can be obtained easily from the derivatives of $g_x(x,\alpha)$ as before.

3.2.1 Bifurcations and test functions for NS curves

The bifurcations that can be detected along the Neimark-Sacker curve are:

- *Chenciner* point, denoted as CH: $\phi_1 = 0$
- *Flip+Neimark-Sacker* point, denoted as PDNS: $\phi_2 = 0$; $\phi_6 \neq 0$
- *Fold+Neimark-Sacker* point, denoted as LPNS: $\phi_3 = 0$; $\phi_4 \neq 0$
- *1:1 resonance*. We will denote this bifurcation with R1 : $\phi_3 = \phi_4 = 0$
- *Double Neimark-Sacker* point, denoted as NSNS: $\phi_5 = 0$
- *1:2 resonance* point, denoted as R2: $\phi_2 = \phi_6 = 0$
- *1:3 resonance* point, denoted as R3: $\phi_7 = 0$
- *1:4 resonance* point, denoted as R4 : $\phi_8 = 0$

To detect these singularities, we define 8 test functions:

- $\phi_1 = Re(d)$. (see formula (2.16))
- $\phi_2 = \det(A + I_n)$.
- $\phi_3 = \det(A - I_n)$.
- $\phi_4 = k - 1$.
- $\phi_5 = \det(A|_{N^C} \odot A|_{N^C})$.
- $\phi_6 = k + 1$.
- $\phi_7 = k + \frac{1}{2}$.

- $\phi_8 = k$.

In these formulae $A = g_x(x, \alpha)$ and the vectors $p, q \in \mathbb{C}^n$ satisfy

$$Aq = e^{i\theta}q,\ A^T p = e^{-i\theta}p,\ \langle \mathrm{Re}\ q, \mathrm{Im}\ q\rangle = 0,\ \langle q, q\rangle = \langle p, q\rangle = 1.$$

The subspace N^C of $\mathbb{R}^n$ is the orthogonal complement of the critical two-dimensional left eigenspace associated with the pair of multipliers with unit product. $A|_{N^C} \odot A|_{N^C}$ is an $m \times m$ matrix where $2m = (n-2)(n-3)$.

In this case the singularity matrix is:

$$S = \begin{pmatrix} 0 & - & - & - & - & - & - & - \\ - & 0 & - & - & - & 1 & - & - \\ - & - & 0 & 1 & - & - & - & - \\ - & - & 0 & 0 & - & - & - & - \\ - & - & - & - & 0 & - & - & - \\ - & 0 & - & - & - & 0 & - & - \\ - & - & - & - & - & - & 0 & - \\ - & - & - & - & - & - & - & 0 \end{pmatrix}. \tag{3.14}$$

3.3 Bifurcation analysis of codim 2 bifurcations of maps

When two system parameters are allowed to vary, one may encounter the following eleven codim 2 bifurcations of period-K orbits in generic families of maps (1.3) where curves of codim 1 bifurcations intersect or meet tangentially [42]. The critical multipliers with modulus 1 are generally denoted by λ_1 and λ_2.

D_1 : $\lambda_1 = 1,\ a = 0$ (cusp, CP);
D_2 : $\lambda_1 = -1,\ b = 0$ (generalized flip, GPD);
D_3 : $\lambda_{1,2} = e^{\pm i\theta_0},\ c = 0$ (Chenciner bifurcation, CH);
D_4 : $\lambda_1 = \lambda_2 = 1$ (1:1 resonance, R1);
D_5 : $\lambda_1 = \lambda_2 = -1$ (1:2 resonance, R2);
D_6 : $\lambda_{1,2} = e^{\pm i\theta_0},\ \theta_0 = \frac{2\pi}{3}$ (1:3 resonance, R3);
D_7 : $\lambda_{1,2} = e^{\pm i\theta_0},\ \theta_0 = \frac{\pi}{2}$ (1:4 resonance, R4);
D_8 : $\lambda_1 = 1, \lambda_2 = -1$ (fold-flip LPPD);
D_9 : $\lambda_1 = 1, \lambda_{2,3} = e^{\pm i\theta_0}$ (fold-NS, LPNS);
D_{10} : $\lambda_1 = -1, \lambda_{2,3} = e^{\pm i\theta_0}$ (flip-NS, PDNS);
D_{11} : $\lambda_{1,2} = e^{\pm i\theta_0},\ \lambda_{3,4} = e^{\pm i\theta_1}$ (double NS, NSNS).

In 6 out of 11 cases, branches of local codim 1 bifurcations of higher period are rooted at codim 2 bifurcation points and for these cases we also incorporate the parameter-dependent part of the normal form and provide asymptotic expressions for the new curves. In Section 3.4 we specify how we switch to the continuation of those branches.

In the next 11 subsubsections we give the normal forms of the codim 2 bifurcations. The $\mathcal{O}$-symbol denotes higher order terms in phase-variables, the coefficients of which may also depend on parameters. But the qualitative picture is determined by the lowest order terms listed below. We refer to [62], Ch. 9, and [63, 64] for more details, including explicit expressions for all critical normal form coefficients. If a complex critical eigenvalue λ is involved, it is always assumed that $\lambda^\nu \neq 1$ for $\nu = 1, 2, 3, 4$. In some cases, we combine two real unfolding parameters (β_1, β_2) into one complex parameter $\beta = \beta_1 + i\beta_2 \in \mathbb{C}$.

3.3.1 Cusp (CP)

The critical smooth normal form on the center manifold at a *cusp bifurcation* is

$$w \mapsto w + a_2 w^3 + \mathcal{O}(|w|^4), \quad w \in \mathbb{R}, \tag{3.15}$$

where, generically, $a_2 \neq 0$. Under this condition, a generic two-parameter unfolding of this singularity has two fold curves in the parameter plane which form a cuspidal wedge. For nearby parameter values, the map g has up to three fixed

points that pairwise collide along the fold curves. In the direct product of the state and the parameter spaces, there is one smooth fold curve, so no branch switching is needed. CL_MATCONTM reports the value of a_2 at the bifurcation.

3.3.2 Generalized flip (GPD)

Near a *generalized flip* bifurcation the restriction of the map g to the parameter-dependent center manifold is smoothly equivalent to the normal form

$$w \mapsto -(1+\beta_1)w + \beta_2 w^3 + b_2(\beta)w^5 + \mathcal{O}(|w|^6), \quad w \in \mathbb{R}, \tag{3.16}$$

where, generically, the coefficient $b_2(0) \neq 0$, while the components of $\beta = (\beta_1, \beta_2)$ are smooth functions of α, which can serve as new unfolding parameters. The value of $b_2(0)$ is reported by CL_MATCONTM. The fixed point $w = 0$ of the map (3.16) exhibits a flip bifurcation for $\beta_1 = 0$. It is well-known that from the point $\beta = 0$, corresponding to the generalized flip bifurcation, a fold curve of double-period cycles emanates. The asymptotic expression for this curve in (3.16) is given by

$$(w, \beta_1, \beta_2) = (\varepsilon, -b_2\varepsilon^4 + \mathcal{O}(\varepsilon^5), -2b_2\varepsilon^2 + \mathcal{O}(\varepsilon^3)). \tag{3.17}$$

3.3.3 Chenciner (CH)

If $\mathrm{e}^{i\nu\theta_0} \neq 1$ for $\nu = 1, 2, \ldots, 6$, the critical smooth normal form on the center manifold at the *Chenciner bifurcation* can be written as

$$z \mapsto z\mathrm{e}^{i\theta_0}(1 + d_1|z|^2 + d_2|z|^4 + \mathcal{O}(|z|^6), \quad z \in \mathbb{C}, \tag{3.18}$$

where $c_1 = \Re(d_1) = 0$ but, generically, $c_2 = \Re(d_2) + \frac{1}{2}\Im(d_1)^2 \neq 0$ and is reported to the user. A generic two-parameter unfolding of this singularity has a complicated bifurcation set due to the "collision" and destruction of two closed invariant curves of different stability born via the sub- and super-critical Neimark-Sacker bifurcations, respectively. There are no cycle bifurcation curves rooted at this bifurcation.

3.3.4 1:1 resonance (R1)

The restriction of the map at a 1:1 *resonance* to the corresponding center manifold can be written in the form

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \mapsto \begin{pmatrix} w_1 + w_2 \\ w_2 + a_1 w_1^2 + b_1 w_1 w_2 \end{pmatrix} + \mathcal{O}(\|w\|^3), \quad w \in \mathbb{R}^2. \tag{3.19}$$

Generically, a Neimark-Sacker bifurcation curve of fixed points meets tangentially the fold bifurcation curve. CL_MATCONTM reports the sign of $a_1(b_1 - 2a_1)$ giving the first Lyapunov coefficient along the NS curve near the bifurcation. The local branch switching problem is trivial here, since both curves correspond to fixed points of g. The full bifurcation diagram near the codim 2 point is complicated and involves global bifurcations, e.g. tangencies of stable and unstable invariant manifolds of saddle fixed points of g and destruction of a closed invariant curve born via the Neimark-Sacker bifurcation.

3.3.5 1:2 resonance (R2)

Near a 1:2 *resonance* the restriction of the map g to the parameter-dependent center manifold is smoothly equivalent to the normal form

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \mapsto \begin{pmatrix} -w_1 + w_2 \\ \beta_1 w_1 + (-1 + \beta_2) w_2 + C_1(\beta) w_1^3 + D_1(\beta) w_1^2 w_2 \end{pmatrix} + \mathcal{O}(\|w\|^4), \quad w \in \mathbb{R}^2, \tag{3.20}$$

that depends on two unfolding parameters (β_1, β_2). If $C_1(0) < 0$, then there is a Neimark-Sacker curve of fixed points of g with double period that emanates from the flip bifurcation curve $\beta_2 = 0$ of fixed points. It has the following asymptotic expression

$$(w_1^2, w_2, \beta_1, \beta_2) = \left(-\frac{1}{C_1}, 0, 1, \left(2 + \frac{D_1}{C_1} \right) \right) \varepsilon + \mathcal{O}(\varepsilon^2). \tag{3.21}$$

There are also global bifurcations associated with the destruction of closed invariant curves. CL_MATCONTM reports the values of $4C_1(0)$ and $-2D_1(0) - 6C_1(0)$, (relevant for the flow approximation) to the user.

3.3.6 1:3 resonance (R3)

At a 1:3 *resonance*, the restriction of the map g to the parameter-dependent center manifold is smoothly equivalent to the normal form

$$z \mapsto (\mathrm{e}^{2i\pi/3} + \beta)z + B_1(\beta)\bar{z}^2 + C_1(\beta)z|z|^2 + \mathcal{O}(|z|^4), \quad z \in \mathbb{C}, \tag{3.22}$$

where $\beta = \beta_1 + i\beta_2 \in \mathbb{C}$. A generic unfolding of this singularity has a period-3 saddle cycle that does not bifurcate for nearby parameter values, although it merges with the primary fixed point as the parameters approach R3. Only global bifurcations related to the destruction of a closed invariant curve born via the primary Neimark-Sacker bifurcation occur in a neighborhood of this codim 2 point.

Note that the period-3 cycle becomes neutral near this bifurcation. Recall that a saddle cycle is called *neutral* if the corresponding fixed point has a pair of real eigenvalues with product 1. This singularity is important in analyzing global bifurcations of invariant manifolds of cycles. Moreover, the curve of neutral period-3 saddle cycles may turn into a true Neimark-Sacker bifurcation at R1 or R2. Therefore, we give here an asymptotic of this curve.

First we need a vector field for which the time-1 flow approximates the third iterate of the map, i.e.

$$\widetilde{g}(\eta, \widetilde{\beta}) = \widetilde{\beta}\eta + \bar{\eta}^2 + C_0(\beta)\eta^2\bar{\eta} + \mathcal{O}(|\eta|^4), \tag{3.23}$$

where

$$\widetilde{\beta} = 3\mathrm{e}^{-2i\pi/3}\beta, \; z = \frac{1}{|B_1(\beta)|}\mathrm{e}^{i\arg(B_1(\beta))/3}\eta,$$

and

$$C_0(\beta) = \frac{1}{3}\left(\frac{C_1(\beta)}{|B_1(\beta)|^2}\mathrm{e}^{-2i\pi/3} - 1\right).$$

We write $C_0 = a + ib$, where upon detection a is reported, so that for $\eta = \rho\mathrm{e}^{i\phi}$ the neutral saddle curve has the following asymptotic expression

$$(\rho, \phi, \beta_1, \beta_2) = \left(\varepsilon, s(\pi/6 - a\varepsilon/3), -2a\varepsilon^2, s\varepsilon - b\varepsilon^2\right) + \mathcal{O}(\varepsilon^3), \tag{3.24}$$

where $s = \pm 1$.

3.3.7 1:4 resonance (R4)

Near a 1:4 *resonance* the restriction of the map g to the parameter-dependent center manifold is smoothly equivalent to the normal form

$$z \mapsto (i+\beta)z + C_1(\beta)z^2\bar{z} + D_1(\beta)\bar{z}^3 + \mathcal{O}(|z|^4), \quad z \in \mathbb{C}, \tag{3.25}$$

where $\beta = \beta_1 + i\beta_2 \in \mathbb{C}$. For this bifurcation we do not only need this parameter-dependent normal form, but also an approximation of its 4th iterate by a unit-time shift along orbits of a vector field

$$\widetilde{g}(\eta, \widetilde{\beta}) = \widetilde{\beta}\eta + A_0(\beta)\eta^2\bar{\eta} + \bar{\eta}^3 + \mathcal{O}(|\eta|^4), \tag{3.26}$$

where $\eta \in \mathbb{C}$ and $\widetilde{\beta} = \tilde{\beta}_1 + i\tilde{\beta}_2, \tilde{\beta}_i \in \mathbb{R}$. Here we use

$$z = \frac{1}{\sqrt{|D_1(\beta)|}} \mathrm{e}^{i\arg(D_1(\beta))/4}\eta, \;\; A_0(\beta) = -i\frac{C_1(\beta)}{|D_1(\beta)|}.$$

Moreover, we have

$$\begin{pmatrix} \widetilde{\beta}_1 \\ \widetilde{\beta}_2 \end{pmatrix} = \begin{pmatrix} 0 & 4 \\ -4 & 0 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}. \tag{3.27}$$

There are three possible branch switches for this bifurcation. Denote the reported values by $a = \Re(A_0(0))$ and $b = \Im(A_0(0))$. If $\Delta \equiv a^2 + b^2 - 1 > 0$, then there are two half-lines $l_{1,2}$ of a limit-point curve of cycles with four times the original period. If

$$|b| > \frac{(1+a^2)}{\sqrt{1-a^2}},$$

then there is a curve n_1 along which a cycle of four times the primary period exhibits a Neimark-Sacker bifurcation. Using $\eta = re^{i\phi}$ we have the following

approximations

$$\begin{aligned} l_{1,2} : (r^2, \phi, \widetilde{\beta}_1, \widetilde{\beta}_2) =& \Bigg(\varepsilon, \frac{1}{4} \arctan \left(\frac{ab \pm \sqrt{\Delta}}{b^2 - 1} \right) + \mathcal{O}(\varepsilon), \\ & \frac{-a\Delta \mp b\sqrt{\Delta}}{a^2 + b^2} \varepsilon, \frac{-b\Delta \pm a\sqrt{\Delta}}{a^2 + b^2} \varepsilon \Bigg) + \mathcal{O}(\varepsilon^2) \\ n_1 : (r^2, \phi, \widetilde{\beta}_1, \widetilde{\beta}_2) =& \quad \big(\varepsilon + \mathcal{O}(\varepsilon^2), \operatorname{sign}(b) \arccos(a)/4 + \mathcal{O}(\varepsilon), \\ & -2a\varepsilon + \mathcal{O}(\varepsilon^2), -(b - \operatorname{sign}(b)\sqrt{1 - a^2})\varepsilon + \mathcal{O}(\varepsilon^2) \big) . \end{aligned} \tag{3.28}$$

Taking into account (3.27), we obtain expressions for the unfolding parameters β_1 and β_2. If, in the formula for n_1, we replace $\operatorname{sign}(b)$ by $-\operatorname{sign}(b)$, then this gives the asymptotic for a neutral saddle singularity of the period-4 cycle.

Generically, there are also global bifurcations near R4.

3.3.8 Fold–Neimark-Sacker (LPNS)

For a *fold – Neimark-Sacker* bifurcation, the critical normal form on the center manifold is given by

$$\begin{pmatrix} w \\ z \end{pmatrix} \mapsto \begin{pmatrix} w + sz\bar{z} + w^2 + cx^3 \\ e^{i\theta_0} z + awz + bzw^2 \end{pmatrix} + \mathcal{O}(\|(w,z)\|^4), \quad (w,z) \in \mathbb{R} \times \mathbb{C}. \tag{3.29}$$

The critical coefficients s, a, b, c are reported. Depending on their values, several bifurcation scenarios are possible in parameter-dependent unfoldings, which all involve global phenomena.

3.3.9 Fold–Flip (LPPD)

Near a *fold–flip* bifurcation, the restriction of the map g to the parameter-dependent center manifold is smoothly equivalent to the normal form

$$\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \mapsto$$
$$\begin{pmatrix} \beta_1 + (1+\beta_2)w_1 + a(\beta)w_1^2 + b(\beta)w_2^2 + c_1(\beta)w_1^3 + c_2(\beta)w_1w_2^2 \\ -w_2 + e(\beta)w_1w_2 + c_3(\beta)w_1^2w_2 + c_4(\beta)w_2^3 \end{pmatrix}$$
$$+ \mathcal{O}(\|w\|^4), \quad w \in \mathbb{R}^2. \tag{3.30}$$

3.4 Branch switching at codim 2 bifurcation points

Here we address the problem of branch switching at codim 2 bifurcation points of maps, when the emanating curve corresponds to a local bifurcation. This involves the generalized flip, 1:2 resonance, 1:3 resonance, 1:4 resonance, fold-flip and flip-Neimark-Sacker bifurcations only. To obtain appropriate initial continuation data for the original map, we combine parameter-dependent center-manifold reduction with asymptotic expressions for the new curves given in Section 3.3.

Although we know that in several cases also global bifurcations are involved, we will not try to switch to those branches as the continuation of these global bifurcations is out of the scope of this thesis.

3.4.1 Parameter-Dependent Center-Manifold Reduction

In all our cases, the map $g(x,\alpha) : \mathbb{R}^n \times \mathbb{R}^2 \to \mathbb{R}^n$, where g is defined by (1.3), satisfies $g(x_0,\alpha_0) = x_0$, and its Jacobian matrix $A = g_x(x_0,\alpha_0)$ has at most 3 multipliers on the unit circle. Furthermore we know a parameter-dependent smooth normal form $G(w,\beta)$ for the corresponding bifurcation, see Section 3.3. Then we assume a relation

$$\alpha - \alpha_0 = V(\beta) = v_{10}\beta_1 + v_{01}\beta_2 + \mathcal{O}(\|\beta\|^2), \tag{3.31}$$

between the original and the unfolding parameters. Note that V incorporates linear scalings. Occasionally, we interpret $\beta_1 = \bar{\beta}_2$ as one parameter $\beta \in \mathbb{C}$; in such cases: $v_{01} = \bar{v}_{10} \in \mathbb{C}^2$.

To find a parameter-dependent center-manifold as the graph of $x = x_0 + H(w, \beta)$ we make a Taylor expansion of the *homological equation*

$$g(x_0 + H(w, \beta), \alpha_0 + V(\beta)) = H(G(w, \beta), \beta), \tag{3.32}$$

in w and β at $(w, \beta) = (0, 0)$, where we expand g as in (3.34) and write

$$H(w, \beta) = \sum_{|\mu|+|\nu|\geq 1} h_{\mu,\nu} w^\mu \beta^\nu, \tag{3.33}$$

and μ, ν are multi-indices. All coefficients must vanish and this leads to a solution for H and V. Below we will focus only on the parameter-dependent computations and assume full knowledge of the critical center-manifold and the critical normal form coefficients, see [62, 63, 64]. The solvability conditions imposed coincide with the transversality of the original family to the bifurcation manifold. A similar technique was introduced in [14], §11, to switch at codim 2 bifurcations of equilibria in ODEs.

Initial data for the new curve is now provided by substituting for w and β the asymptotic expression in ε of Section 3.3 into H and V up to a certain order in ε, usually 2. ε can be adjusted to obtain a starting point near the emanating curve for which the continuation code converges to a point on the new curve.

It will be convenient to introduce some notation. Let p denote an eigenvector of A^T corresponding to the eigenvalue -1 of A. We will then write $\Gamma : \mathbb{R}^{n+2} \to \mathbb{R}^n$ for $\Gamma(q, v) = \langle p, A_1(q, v) + B(q, (I_n - A)^{-1} J_1 v)\rangle$ and $\gamma_i = \Gamma(q, e_i)$ for the evaluation of Γ on the standard basis vectors in $\mathbb{R}^2$. If $\gamma_i \neq 0$ for $i = 1, 2$ then $s_1 = \frac{1}{(\gamma_1^2+\gamma_2^2)}(\gamma_1, \gamma_2)^T$ and $s_2 = (-\gamma_2, \gamma_1)^T$ compose a new orthogonal basis in $\mathbb{R}^2$.

$$\begin{aligned} g(x_0+x,\alpha_0+\alpha) &= x_0 + Ax + \tfrac{1}{2}B(x,x) + \tfrac{1}{6}C(x,x,x) \\ &\quad + \tfrac{1}{24}D(x,x,x,x) + \tfrac{1}{120}E(x,x,x,x,x) \\ &\quad + J_1\alpha + \tfrac{1}{2}J_2(\alpha,\alpha) \\ &\quad + A_1(x,\alpha) + \tfrac{1}{2}B_1(x,x,\alpha) \\ &\quad + \tfrac{1}{6}C_1(x,x,x,\alpha) + \tfrac{1}{24}D_1(x,x,x,x,\alpha) \\ &\quad + \tfrac{1}{2}A_2(x,\alpha,\alpha) + \tfrac{1}{4}B_2(x,x,\alpha,\alpha) + \tfrac{1}{12}C_2(x,x,x,\alpha,\alpha) \\ &\quad + \dots, \end{aligned} \tag{3.34}$$

Generalized flip

The homological equation (3.32) provides the following systems to be solved

$$(A-I_n)[h_{010},\ h_{001}] = -J_1[v_{10},\ v_{01}], \tag{3.35}$$

$$\begin{aligned} (A+I_n)[h_{110},\ h_{101}] &= -[q,\ 0] - A_1(q,[v_{10},\ v_{01}]) \\ &\quad - B(q,[h_{010},\ h_{001}]), \end{aligned} \tag{3.36}$$

where $[a,\ b]$ is an $n\times 2$-matrix with columns $a,b\in\mathbb{R}^n$. The higher orders give

$$\begin{aligned} (A-I_n)h_{210} &= 2h_{200} - [B_1(q,q,v_{10}) + B(h_{200},h_{010}) + A_1(h_{200},v_{10}) \\ &\quad +\ 2B(q,h_{110}) + C(q,q,h_{010})]\,, \\ (A-I_n)h_{201} &= -[B_1(q,q,v_{01}) + B(h_{200},h_{001}) + A_1(h_{200},v_{01}) \\ &\quad +\ 2B(q,h_{101}) + C(q,q,h_{001})]\,, \\ (A+I_n)h_{310} &= -3h_{300} - [D(q,q,q,h_{010}) + 3C(q,q,h_{110}) + C_1(q,q,q,v_{10}) \\ &\quad + 3B(h_{110},h_{200}) + B(h_{300},h_{010}) + 3B_1(h_{200},q,v_{10}) \\ &\quad + A_1(h_{300},v_{10}) + 3C(h_{200},q,h_{010}) + 3B(h_{210},q)]\,, \\ (A+I_n)h_{301} &= 6q - [D(q,q,q,h_{001}) + 3C(q,q,h_{101}) + C_1(q,q,q,v_{01}) \\ &\quad + 3B(h_{101},h_{200}) + B(h_{300},h_{001}) + 3B_1(h_{200},q,v_{01}) \\ &\quad + A_1(h_{300},v_{01}) + 3C(h_{200},q,h_{001}) + 3B(h_{201},q)]\,. \end{aligned} \tag{3.37, 3.38}$$

The linear part of V involves 4 unknowns to be found from equations with singular left hand sides, i.e. Eqs. (3.36), (3.37), (3.38). For the solution we

start with substitution of $[h_{010},\ h_{001}] = (I_n - A)^{-1}J_1[v_{10},\ v_{01}]$ from (3.35) into (3.36). As the LHS of (3.36) is singular, the RHS must be orthogonal to the adjoint eigenvector p. Application of the Fredholm Alternative leads to

$$\begin{aligned}\langle p, (A + I_n)[h_{110},\ h_{101}]\rangle &= -\langle p, [q,\ 0] + A_1(q, [v_{10},\ v_{01}]) \\ &\quad + B(q, (I_n - A)^{-1}J_1[v_{10}, v_{01}])\rangle.\end{aligned}$$

We see that the operator $\Gamma(q, v)$ appears naturally and rewrite it as

$$[\gamma_1,\ \gamma_2][v_{10},\ v_{01}] = [-1,\ 0].$$

The general solution is given by

$$v_{10} = -s_1 + \delta_1 s_2, \quad v_{01} = \delta_2 s_2, \quad \delta_1, \delta_2 \in \mathbb{R}.$$

Since v_{10}, v_{01} appear linearly in these equations (via the multilinear functions), application of the Fredholm Alternative to (3.37), (3.38) results in a linear system for the constants δ_1, δ_2.

1:2 resonance

As before, we first list the necessary equations obtained from the homological equation

$$(A - I_n)[h_{0010},\ h_{0001}] = -J_1[v_{10},\ v_{01}], \tag{3.39}$$

$$(A + I_n)[h_{1010},\ h_{1001}] = [q_1,\ 0] - A_1(q_0, [v_{10},\ v_{01}]) - B(q_0, [h_{0010},\ h_{0001}]), \tag{3.40}$$

$$\begin{aligned}(A + I_n)[h_{0110},\ h_{0101}] &= [h_{1010},\ q_1 + h_{1001}] - A_1(q_1, [v_{10},\ v_{01}]) \\ &\quad - B(q_1, [h_{0010},\ h_{0001}]).\end{aligned} \tag{3.41}$$

As for the generalized flip, we substitute $[h_{0010},\ h_{0001}] = (I_n - A)^{-1}J_1[v_{10},\ v_{01}]$ into (3.40). As (3.40) is similar to (3.36), the solution for v_{10} and v_{01} is now

$$v_{10} = s_1 + \delta_1 s_2, \quad v_{01} = \delta_2 s_2, \quad \delta_1, \delta_2 \in \mathbb{R}.$$

We substitute the s_i into (3.41) and write

$$\begin{aligned} Q_1 = &\ \langle p_1, A_1(q_0, s_1) + B(q_0, (A - I_n)^{-1} J_1 s_1)\rangle, \quad Q_2 = \Gamma(q_1, s_1) \\ Q_3 = &\ \langle p_1, A_1(q_0, s_2) + B(q_0, (A - I_n)^{-1} J_1 s_2)\rangle, \quad Q_4 = \Gamma(q_1, s_2). \end{aligned}$$

A little algebra shows that

$$\delta_1 = -\left(\frac{Q_1 + Q_2}{Q_3 + Q_4}\right), \quad \delta_2 = \frac{1}{Q_3 + Q_4}.$$

1:3 resonance

We follow a slightly different procedure here. We want to find $V(\beta) = v\beta + \bar{v}\bar{\beta}$, where $\beta = \beta_1 + i\beta_2$. Then we treat β and $\bar{\beta}$ as independent variables which makes it slightly easier to find the solutions. As the final $V(\beta)$ should be real, it follows that $v = v_{10} = \bar{v}_{01}$.

Let $\lambda = \mathrm{e}^{2i\pi/3}$ and introduce $Aq = \lambda q,\ A^T p = \bar{\lambda} p,\ \langle p, q\rangle = 1$.
As before, the first linear systems resulting from (3.32) are given by

$$\begin{aligned} (A - I_n)[h_{0010},\ h_{0001}] &= -J_1[v_{10},\ v_{01}], \\ (A - \lambda I_n)[h_{1010},\ h_{1001}] &= [q,\ 0] - A_1(q, [v_{10},\ v_{01}]) - B(q, [h_{0010},\ h_{0001}]), \end{aligned}$$

and two complex conjugated systems for h_{0101} and h_{0110}. With the same approach we will now find complex γ_i and rewriting the system for $v = v_{10} = \bar{v}_{01}$ we have $(\gamma_1, \gamma_2)v = 1, (\gamma_1, \gamma_2)\bar{v} = 0$, with $v = (\bar{\gamma}_2, -\bar{\gamma}_1)/(\gamma_1\bar{\gamma}_2 - \gamma_2\bar{\gamma}_1)$ as solution.

1:4 resonance

Replacing $\lambda = i$ we can repeat the procedure for the case of 1:3 resonance.

Fold-Flip

Let $Aq_{1,2} = \pm q_{1,2},\ A^T p_{1,2} = \pm p_{1,2},\ \langle p_1, q_1\rangle = \langle p_2, q_2\rangle = 1$. The necessary systems to solve from the homological equation (3.32) are

$$(A - I_n)[h_{0010},\ h_{0001}] = [q_1,\ 0] - J_1[v_{10},\ v_{01}], \tag{3.42}$$

$$\begin{aligned}(A-I_n)[h_{1010},\ h_{1001}] &= [h_{2000},\ q_1] - A_1(q_1,[v_{10},\ v_{01}]) \qquad (3.43)\\ &\quad -B(q_1,[h_{0010},\ h_{0001}]),\\ (A+I_n)[h_{0110},\ h_{0101}] &= [h_{1100},\ 0] - A_1(q_2,[v_{10},\ v_{01}]) \qquad (3.44)\\ &\quad -B(q_2,[h_{0010},\ h_{0001}]).\end{aligned}$$

First remark that all matrices in the left-hand sides are singular. If we take $(\gamma_1,\gamma_2) = p_1^T J_1$, we can define the orthogonal vectors s_1 and s_2 as before and then $v_{10} = s_1 + \delta_1 s_2$ and $v_{01} = \delta_2 s_2$ solve system (3.42) for arbitrary δ_1, δ_2. Bordering the singular matrix $(A - I_n)$ one can solve for h_{0010} and h_{0001}. Any multiple of q_1 can be added to h_{0010} and h_{0001}, so we use $h_{0010} = (A-I_n)^{INV}(q_1 - J_1 v_{10}) + \delta_3 q_1$ and $h_{0010} = -(A-I_n)^{INV}(J_1 v_{01}) + \delta_4 q_1$. We will use this freedom to solve equations (3.43) and (3.44) simultaneously for all δ's. Note that h_{2000} and h_{1100} are also found using bordered systems chosen, but such that $\langle p_1, h_{2000}\rangle = \langle p_2, h_{1100}\rangle = 0$.

Then we obtain the following $4-$dimensional system

$$\begin{pmatrix} L & 0_{2\times 2} \\ 0_{2\times 2} & L \end{pmatrix} \begin{pmatrix} \delta_1 \\ \delta_3 \\ \delta_2 \\ \delta_4 \end{pmatrix} = \begin{pmatrix} -\langle p_1, A_1(q_1,s_1) + B(q_1,(A-I_n)^{INV}(q_1 - J_1 s_1))\rangle \\ -\langle p_2, A_1(q_2,s_1) + B(q_2,(A-I_n)^{INV}(q_1 - J_1 s_1))\rangle \\ 1 \\ 0 \end{pmatrix}, \quad (3.45)$$

where L is defined by

$$L = \begin{pmatrix} \langle p_1, A_1(q_1,s_2) + B(q_1,(I_n-A)^{INV} J_1 s_2)\rangle & \langle p_1, B(q_1,q_1)\rangle \\ \langle p_2, A_1(q_2,s_2) + B(q_2,(I_n-A)^{INV} J_1 s_2)\rangle & \langle p_2, B(q_1,q_2)\rangle \end{pmatrix}. \quad (3.46)$$

Notice that $2a(0) = \langle p_1, B(q_1,q_1)\rangle$ and that q_1 can be chosen such that $e(0) = \langle p_2, B(q_1,q_2)\rangle = 1$. The condition $\gamma_1\gamma_2 \det(L) \neq 0$ is equivalent with the transversality to the bifurcation manifold of the family $g(x,\alpha)$.

Flip-Neimark-Sacker

Introduce $Aq_1 = q_1,\ A^T p_1 = p_1,\ \langle p_1, q_1\rangle = 1$, and $Aq_2 = e^{i\theta_0} q_2,\ A^T p_2 = e^{-i\theta_0} p_2,\ \langle p_2, q_2\rangle = 1$. The linear systems obtained from the homological equation (3.32) are

$$\begin{aligned}
(A - I_n)[h_{00010},\ h_{00001}] &= -J_1[v_{10},\ v_{01}],\\
(A + I_n)[h_{10010},\ h_{10001}] &= [-q_1,\ 0] - A_1(q_1, [v_{10},\ v_{01}]) -\\
&\quad B(q_1, [h_{00010},\ h_{00001}]),\\
(A - e^{i\theta_0} I_n)[h_{01010},\ h_{01001}] &= [0,\ q_2 e^{i\theta_0}] - A_1(q_2, [v_{10},\ v_{01}])\\
&\quad -B(q_2, [h_{00010},\ h_{00001}]).
\end{aligned}$$

The same approach as for the generalized flip and 1:2-resonance cases is to substitute the formal solution of the first equation into the second and we write

$$v_{10} = -s_1 + \delta_1 s_2, \quad v_{01} = \delta_2 s_2,$$

where the constants δ_i are to be found from the last equation. We compute

$$Q_i = \langle p_2, A_1(q_2, s_i) + B(q_2, (I_n - A)^{-1} J_1 s_i)\rangle,$$

for $i = 1, 2$. To obtain the derivative of the modulus and not the argument of the complex multiplier, we proceed similar to [94], Appendix, but adapt to the case of maps. Then we find the following real solutions

$$\delta_1 = \frac{\Re(e^{-i\theta_0} Q_1)}{\Re(e^{-i\theta_0} Q_2)},\ \delta_2 = -\frac{1}{\Re(e^{-i\theta_0} Q_2)}. \tag{3.47}$$

A new branch predicted by (3.30) for a generic map g is a Neimark-Sacker of double period that exists if $be > 0$ and has the asymptotic expression

$$(x, y^2, \beta_1, \beta_2) = \left(-\frac{c_4}{e}, 1, -b, -\frac{2b + ec_2 - 2(a+e)c_4}{e}\right)\varepsilon + \mathcal{O}(\varepsilon^2). \tag{3.48}$$

CL_MATCONTM reports the coefficients $\frac{a}{e}$ and be and the sign of the Lyapunov coefficient if applicable. As for the majority of the considered cases, there are also global bifurcations near this codim 2 point.

3.4.2 Flip – Neimark-Sacker (PDNS)

Near a *flip – Neimark-Sacker* bifurcation, the restriction of the map g to the parameter-dependent center manifold is smoothly equivalent to the parameter-dependent normal form

$$\begin{pmatrix} w \\ z \end{pmatrix} \mapsto \begin{pmatrix} -w(1+\beta_1+c_1(\beta)w^2+c_2(\beta)|z|^2) \\ ze^{i\theta(\beta)}(1+\beta_2+c_3(\beta)w^2+c_4(\beta)|z|^2) \end{pmatrix} + \mathcal{O}(\|(w,z)\|^4), \quad (w,z)\in\mathbb{R}\times\mathbb{C}, \tag{3.49}$$

where $\Re(c_i(0))$ are reported and $\theta(0)=\theta_0$. Besides global bifurcations, a Neimark-Sacker bifurcation curve of double period for g is rooted at $\beta=0$; it is always present. The asymptotic expression of this curve is given by

$$(w^2, z, \beta_1, \beta_2) = (1, 0, -c_1, -\mathrm{sign}(c_1)\Re(c_3)\,\varepsilon + \mathcal{O}(\varepsilon^2). \tag{3.50}$$

3.4.3 Double Neimark-Sacker (NSNS)

For a *double Neimark-Sacker bifurcation*, provided $l\theta_0 \neq j\theta_1$ for integer l and j with $l+j\leq 4$, the critical normal form on the center manifold is

$$\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \mapsto \begin{pmatrix} z_1(e^{i\theta_0}+c_1|z_1|^2+c_2|z_2|^2) \\ z_2(e^{i\theta_1}+c_3|z_1|^2+c_4|z_2|^2) \end{pmatrix} + \mathcal{O}(\|z\|^4), \quad z\in\mathbb{C}^2. \tag{3.51}$$

Depending on the values of $\Re(c_i)$, which are reported, several bifurcation scenarios are possible in parameter-dependent unfoldings, which all involve global phenomena. To analyse some of them, one has to take into account fourth- and fifth-order terms.

3.5 Initializations of branch switching

For each switch to a codim 1 curve an initializer m-function is constructed, the syntax is as follows :

```
[x0,v0] = init_GPD_LP2(@mapfile, eps, x, p, ap, n);
[x0,v0] = init_R2_NS2(@mapfile, eps, x, p, ap, n);
[x0,v0] = init_R3_NS3(@mapfile, eps, x, p, ap, n);
[x0,v0] = init_R4_LP41(@mapfile, eps, x, p, ap, n);
[x0,v0] = init_R4_LP42(@mapfile, eps, x, p, ap, n);
[x0,v0] = init_R4_NS4(@mapfile, eps, x, p, ap, n);
[x0,v0] = init_LPPD_NS2(@mapfile, eps, x, p, ap, n);
[x0,v0] = init_PDNS_NS2(@mapfile, eps, x, p, ap, n);
```

The arguments are

- mapfile An m-file containing the specifications of the map.

- eps The (positive) amplitude of the initial step.

- x The coordinates of the bifurcating fixed point.

- p The parameters at which the codim 2 bifurcation occurs.

- ap The active parameters which are used.

- n The number of iterates of the bifurcating fixed point.

In some cases it depends on the critical normal form coefficients whether branch switching is possible. If it is not, we stop. If it is, then we specify a new coordinate $\tilde{x}$ and parameter $\tilde{p}$ and return $x0 = (\tilde{x}, \tilde{p})$. So, for example, for the generalized flip bifurcation $\tilde{x} = x + \epsilon q$ and $\tilde{p} = p - 2c_2 v_{01}\epsilon^2 + (-c_2 v_{10} + 2c_2^2 v_{02})\epsilon^4$, where only the parameters specified by ap are updated. If the amplitude ϵ is small, it is enough to use only the leading terms of the asymptotic expressions. For the definition of v_{02} see [69], pp. 109-110.

3.6 Detection and switching graphs

In the detection graph Figure 3.1, we present all codim 1 and codim 2 bifurcation points and their interconnections that can be detected on curves of fixed points and codim 1 bifurcation curves. The graph demonstrates the CL_MATCONTM

continuation strategy. For example, it shows three arrows which connect FP with LP, PD and NS, respectively. They mean that when tracing a fixed point curve FP, the bifurcations LP, PD and NS may be detected and located. Each bifurcation point found may be used to start tracing the corresponding codim 1 curve. All bifurcations in the graph are ordered in accordance with the number of control parameters needed for their continuation.

In the switching graphs Figure 3.2 and Figure 3.3, possible switchings at codim 1 and codim 2 bifurcation points are indicated graphically.

The arrows emanating from a bifurcation point indicate the possible branches that emanate from the bifurcation point. The solid arrows mean that the indicating bifurcation curve is generically present. The dashed arrows mean that the presence of the new bifurcation curve is subjected to inequality conditions on the normal form coefficients. The notation $\times 2$, $\times 3$ and $\times 4$ refers to the period of the emanating bifurcation curves. For example we consider the $R4$ point in Figure 3.3. There are four possible branches emanating form this bifurcation point. A NS curve of the original period is always present (indicated by a solid arrow) and a NS curve of the quadruple period depends on the normal form coefficient b, see §3.3.7 (indicated by a dashed arrow). This NS curve exists if b satisfies

$$|b| > \frac{(1+a^2)}{\sqrt{1-a^2}},$$

There are also two half-line limitpoint curves of cycles with four times the original period emanating from the R4 point . These curves are present if the normal form coefficients a and b , see §3.3.7, satisfy $\Delta \equiv a^2 + b^2 - 1 > 0$.

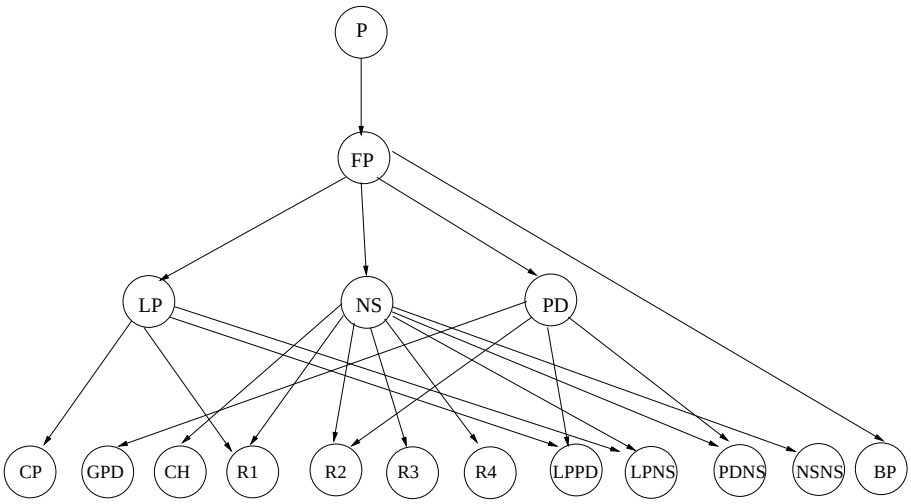


Figure 3.1: Detection graph.

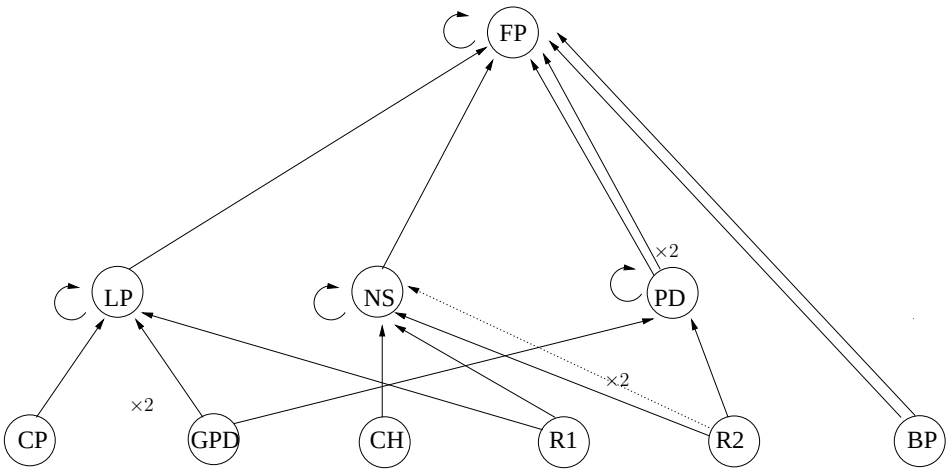


Figure 3.2: Switching graph 1: dashed lines indicate switching subject to constraints and $\times 2$ indicates curve of double period.

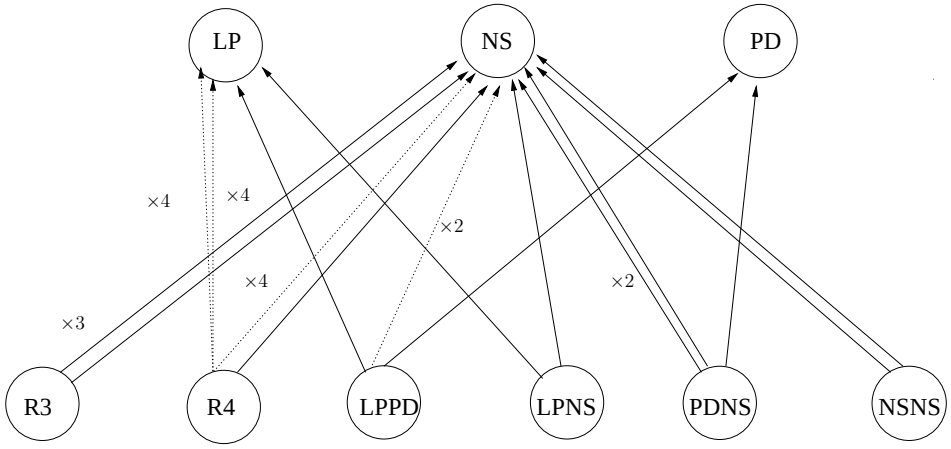


Figure 3.3: Switching graph 2: dashed lines indicate switching subject to constraints, $\times 2$ and $\times 4$ indicate curves of double and quadruple periods, respectively.

Chapter 4

Algorithmic and numerical details

This chapter starts with computation of a Neimark-Sacker curve, defined by (3.12), which includes a detailed description of the continuation variables, defining system, initialization and adaptation corresponding to a NS-curve. We proceed with the computation of the derivatives of g, in (3.12), tensor-vector and vector-tensor-vector products, which are not only needed for the continuation, but also for the computation of the critical normal form coefficients at codim 1 and 2 bifurcation points and for branch switching. We introduce the recursive formulas for derivatives of g w.r.t a state variable and a parameter. Then we present recursive formulas for derivatives of the defining systems for the continuation of codim 1 bifurcation curves. We continue with the numerical computation of finite difference directional derivatives that are used in the computations of the normal form coefficients of codim 1 and codim 2 bifurcations. We describe the implementation of automatic differentiation to compute the multilinear forms that appear in the normal form coefficients. At the end we perform a comparison of the speed and accuracy of the three differentiation strategies.

Parts of this chapter were accepted for publication or submitted in [56, 42].

4.1 Computation of a Neimark-Sacker curve

As an example, we now discuss the implementation of a NS curve, defined by (3.12), starting from a fixed point x. LP and PD curves are implemented in a similar way.

- *Continuation variables*
  The continuation variables are stored in a $(n+3)$-vector, containing:
  - An n-vector with the coordinates of the fixed point x.
  - A scalar κ, that is the real part of the critical multipliers $\mathrm{e}^{\pm i\theta}$.
  - Two active parameters ap.

- *Defining system*
  The defining system consists of $n+2$ equations contaning the fixed point constraint $g(x,\alpha)-x=0$, and the two equations $s_{i_1 j_1}(x,\alpha,\kappa)=0$ and $s_{i_2 j_2}(x,\alpha,\kappa)=0$ defined by (3.12) in the mimimally extended system.

- *Initialization*
  To implement the computation of a NS curve in CL_MATCONTM, we need to initialize the NS curve. We set the parameter vector α, the fixed point x. We also set a global structure *nsmds* containing the following fields:
  - Dimension of the state space (*nsmds.nphase*).
  - The iteration number of the map K (*nsmds.Niterations*)
  - Mapfile where the map is defined (*nsmds.func*)
  - Vector of starting values of parameters and index of the active parameters (*nsmds.*P_0 and *nsmds.ActiveParams*, respectively)
  - The bordering $2\times n$-matrices V_{bor} and W_{bor} in (3.13) (*nsmds.borders.v* and *nsmds.borders.w*, respectively).
  - The 2-vectors *nsmds.index1* and *nsmds.index2* for keeping the auxiliary indexes (i_1, j_1) and (i_2, j_2) , respectively.

Before continuation of a NS curve, the bordering matrices V_{bor} and W_{bor} in (3.13) must be initialized in the file *init_NSm_NSm* such that the matrix

$$M = \begin{pmatrix} (g_x)^2(x,\alpha) - 2\kappa g_x + I_n & W_{bor} \\ V_{bor}^T & O \end{pmatrix}, \tag{4.1}$$

in (3.13) is nonsingular. To this end, suppose $A = g_x(x,\alpha)$ is the Jacobian matrix at a NS point where the eigenvector q corresponds to the multiplier $e^{i\theta}$. We perform the decomposition $[Q,R,E] = qr(\Re(q),\Im(q))$ (QR decomposition with column pivoting). The bordered matrix V_{bor} is given by $V_{bor} = Q(:,1:2)$. The bordered matrix W_{bor} can be computed similarly. Suppose p is an eigenvector that corresponding to the multiplier $e^{i\theta}$ of A^T. We use $[Q,R,E] = qr(\Re(p),\Im(p))$. Then we have $W_{bor} = Q(:,1:2)$.

From (3.12) and (3.13), we get four equations $s_{ij} = 0 \ \ ((i,j) \in \{1,2\})$. We only need two of these equations. We explain the computation of these indicies during the intitialization. Let $A = (jac - I_n \ \ jacp \ \ 0)^T$ where jac and $jacp$ are the Jacobian matrices w.r.t state variables and parameters respectively and I_n is an $n \times n$ identity matrix. To select the two indices we start with the QR factorization $[Q_1,R_1] = qr(A)$. Extending the equality $A^TQ_1 = R_1^T$ by adding rows that correspond to the four components of S, defined by (3.2), we obtain the decomposition

$$\begin{pmatrix} jac - I_n & jacp & 0 \\ (s_{11})_x & (s_{11})_\alpha & (s_{11})_\kappa \\ (s_{12})_x & (s_{12})_\alpha & (s_{12})_\kappa \\ (s_{21})_x & (s_{21})_\alpha & (s_{21})_\kappa \\ (s_{22})_x & (s_{22})_\alpha & (s_{22})_\kappa \end{pmatrix} Q_1 = \left(\begin{array}{cccc|ccc} * & 0 & \dots & 0 & 0 & 0 & 0 \\ * & * & \dots & 0 & 0 & 0 & 0 \\ & & \dots & & & \dots & \\ * & * & \dots & * & 0 & 0 & 0 \\ \hline * & * & \dots & * & J_{res}^T & & \end{array}\right), \tag{4.2}$$

where J_{res} is a 3×4 matrix with rank 2. We want to choose two among four rows of the right-hand-side of (4.2) to make the right-hand-side as well conditioned as possible. We perform the decomposition $[Q,R,E] = qr(J_{res})$. So

$$J_{res}^T E_2 = (q_1 \ \ q_2 \ \ q_3) \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{pmatrix}. \tag{4.3}$$

The righmost matrix in this expression is R, hence its first two diagonal elements are nonzero. From this it follows that the first two columns of $J_{res}^T E_2$ are linearly independent. This means that the columns of J_{res}^T we need to use (equivalently, which s_{ij} we need to choose), are those where the first or second columns of E_2 contains an entry equal to 1.

- *Adaptation*
  It is necessary to adapt the auxiliary variables used in (3.12) while generating a NS curve. The bordering matrices V_{bor} and W_{bor} require updating, since during the continuation they must make sure that the matrix in (3.13) is nonsingular. The border matrix V_{bor} is adapted by replacing it by the normalized and orthogonalized vector V in (3.13). Similarly, the border matrix W_{bor} is adapted by replacing it by the normalized and orthogonalized vector W obtained from solving a system transpose to that in (3.13). Accordingly, during the continuation of a NS curve the new indexes i_1, j_1, i_2 and j_2 must be adapted in the same way as in the initializer.

4.2 Recursive formulas for derivatives of iterates of maps

4.2.1 Derivatives with respect to phase variables

The iteration of (1.2) gives rise to a sequence of points

$$\{x = x_1, x_2, x_3, \ldots, x_{K+1}\},$$

where $x_{J+1} = f^{(J)}(x_1, \alpha)$ for $J = 1, 2, \ldots, K$. Suppose that symbolic derivatives of f up to order 5 can be computed at each point. We write

$$A(x_J)_{i,j} = \frac{\partial f_i}{\partial x_j}(x_J) B(x_J)_{i,j,k} = \frac{\partial^2 f_i}{\partial x_j \partial x_k}(x_J), C(x_J)_{i,j,k,l} = \frac{\partial^3 f_i}{\partial x_j \partial x_k \partial x_l}(x_J),$$

and similarly for $D(x_J)$ and $E(x_J)$.

We want to find recursive formulas for the derivatives of the composition (1.2), i.e. the coefficients of the multilinear functions in (3.34) that we now denote by

$A^{(J)}$, $B^{(J)}$, and $C^{(J)}$ to indicate the iterate explicitly:

$$(A^{(J)})_{i,j} = \frac{\partial (f^{(J)}(x_1))_i}{\partial x_j},$$
$$(B^{(J)})_{i,j,k} = \frac{\partial^2 (f^{(J)}(x_1))_i}{\partial x_j \partial x_k},$$
$$(C^{(J)})_{i,j,k,l} = \frac{\partial^3 (f^{(J)}(x_1)_i}{\partial x_j \partial x_k \partial x_l},$$

and $D^{(J)}$ and $E^{(J)}$ are analogously defined. What follows is a straightforward application of the Chain Rule.

For $J = 1$ we have $A^{(1)} = A(x_1), B^{(1)} = B(x_1)$ and $C^{(1)} = C(x_1)$ and these are known. Now,

$$A^{(J)}_{i,j} = \sum_k \frac{\partial f_i}{\partial x_k}(f^{(J-1)}(x_1)) \frac{\partial (f^{(J-1)}(x_1))_k}{\partial x_j} = \sum_l A(x_J)_{i,k} A^{(J-1)}_{k,j}$$
$$= (A(x_J)A(x_{J-1}) \ \dots \ A(x_1))_{i,j}. \quad (4.4)$$

We see that

$$(F(x,\alpha))_x = A(x^K)A(x^{K-1}) \cdots A(x^1) - I_n, \quad (4.5)$$

where $F(x,\alpha) = f^{(K)}(x,\alpha) - x$.

For the second order derivatives we first write $B^{(J)}$ in coordinates

$$\begin{aligned} B^{(J)}_{i,j,k} &= \frac{\partial}{\partial x_j}\frac{\partial}{\partial x_k} f_i(f^{(J-1)}(x)) \\ &= \sum_{l,m} \frac{\partial^2 f_i}{\partial x_l \partial x_m}(x_J) \frac{\partial (f^{(J-1)})_m}{\partial x_j} \frac{\partial (f^{(J-1)})_l}{\partial x_k} + \sum_l \frac{\partial f_i}{\partial x_l}(x_J) \frac{\partial^2 (f^{(J-1)})_l}{\partial x_j \partial x_k}. \end{aligned}$$

For any two vectors q_1 and q_2, we can multiply the previous expression by $(q_1)_j (q_2)_k$ and sum over (j,k) to obtain

$$B^{(J)}(q_1,q_2) = B(x_J)(A^{(J-1)}q_1, A^{(J-1)}q_2) + A(x_J)B^{(J-1)}(q_1,q_2). \quad (4.6)$$

As $A(x_J)$ and $B(x_J)$ are known, (4.6) allows to compute the multilinear form $B^{(K)}(q_1, q_2)$ recursively.

Let $q_i, i = 1, 2, 3, 4, 5$, be given vectors. Multilinear forms with higher order derivatives can be computed with

$$\begin{aligned} C^{(J)}(q_1, q_2, q_3) = \quad & C(x_J)(A^{(J-1)}q_1, A^{(J-1)}q_2, A^{(J-1)}q_3) \\ & +B(x_J)(B^{(J-1)}(q_1, q_2), A^{(J-1)}q_3)^* \\ & +A(x_J)(C^{(J-1)}(q_1, q_2, q_3)), \end{aligned} \tag{4.7}$$

where * means that all combinatorially different terms have to be included, i.e.,

$$\begin{aligned} B(x_J)(B^{(J-1)}(q_1, q_2), A^{(J-1)}q_3)^* \quad = \quad & B(x_J)(B^{(J-1)}(q_1, q_2), A^{(J-1)}q_3) \\ & +B(x_J)(B^{(J-1)}(q_1, q_3), A^{(J-1)}q_2) \\ & +B(x_J)(B^{(J-1)}(q_2, q_3), A^{(J-1)}q_1). \end{aligned}$$

For $D^{(J)}$ we get

$$\begin{aligned} D^{(J)}(q_1, q_2, q_3, q_4) = \quad & D(x_J)(A^{(J-1)}q_1, A^{(J-1)}q_2, A^{(J-1)}q_3, A^{(J-1)}q_4) \\ & +C(x_J)(B^{(J-1)}(q_1, q_2), A^{(J-1)}q_3, A^{(J-1)}q_4)^* \\ & +B(x_J)(B^{(J-1)}(q_1, q_2), B^{(J-1)}(q_3, q_4))^* \\ & +B(x_J)(C^{(J-1)}(q_1, q_2, q_3)), A^{(J-1)}q_4)^* \\ & +A(x_J)D^{(J-1)}(q_1, q_2, q_3, q_4). \end{aligned} \tag{4.8}$$

Finally, for $E^{(J)}$ we have

$$\begin{aligned} & E^{(J)}(q_1, q_2, q_3, q_4, q_5) = \\ & E(x_J)(A^{(J-1)}q_1, A^{(J-1)}q_2, A^{(J-1)}q_3, A^{(J-1)}q_4, A^{(J-1)}q_5) \\ & +D(x_J)(B^{(J-1)}(q_1, q_2), A^{(J-1)}q_3, A^{(J-1)}q_4, A^{(J-1)}q_5)^* \\ & +C(x_J)(B^{(J-1)}(q_1, q_2), B^{(J-1)}(q_3, q_4), A^{(J-1)}q_5)^* \\ & +C(x_J)(C^{(J-1)}(q_1, q_2, q_3), A^{(J-1)}q_4, A^{(J-1)}q_5)^* \\ & +B(x_J)(C^{(J-1)}(q_1, q_2, q_3), B^{(J-1)}(q_4, q_5))^* \\ & +B(x_J)(D^{(J-1)}(q_1, q_2, q_3, q_4))(A^{(J-1)}q_5)^* \\ & +A(x_J)(E^{(J-1)}(q_1, q_2, q_3, q_4, q_5)). \end{aligned} \tag{4.9}$$

The multilinear forms $A^{(K)}(q_1)$, $B^{(K)}(q_1, q_2)$, $C^{(K)}(q_1, q_2, q_3)$, $D^{(K)}(q_1, q_2, q_3, q_4)$ and $E^{(K)}(q_1, q_2, q_3, q_4, q_5)$ are used in the computations of

normal form coefficients for codim 1 and codim 2 bifurcations of period-J cycles and also in branch switching.

4.2.2 Derivatives with respect to parameters

In the continuation of codimension 1 bifurcation curves (§3.1, §3.2), we need derivatives of the form $\frac{\partial f^{(J)}}{\partial \alpha_k}$ and $\frac{\partial^2 f^{(J)}}{\partial \alpha_k \partial x}$ where α_k is a parameter. If enough symbolic derivatives of f are available, then CL_MATCONTM computes these expressions symbolically. The idea is as follows. Taking the derivative of (1.5) with respect to α_k, gives

$$\frac{\partial (f^{(J)}(x_1,\alpha))}{\partial \alpha_k} = \frac{\partial f}{\partial \alpha_k}(x_J,\alpha) + \frac{\partial f}{\partial x}(x_J,\alpha)\frac{\partial (f^{(J-1)}(x_1,\alpha))}{\partial \alpha_k}, \qquad (4.10)$$

which is recursively computable. Also mixed derivatives, which are necessary for continuation and branch switching, can be found recursively:

$$\frac{\partial^2 (f^{(J)}(x_1,\alpha))}{\partial \alpha_k \partial x} = \frac{\partial^2 f}{\partial \alpha_k \partial x}(x_J,\alpha) + \frac{\partial^2 f}{\partial x^2}(x_J,\alpha)\frac{\partial (f^{(J-1)}(x_1,\alpha))}{\partial \alpha_k}. \qquad (4.11)$$

In fact, the recursion is not applied to (4.11) itself, but to its product with a fixed vector.

This is sufficient for all continuations of fixed points and their codimension 1 bifurcations. It is also sufficient for all cases of branch switching from codimension 2 points, except for the case of generalized flip. For this case, we fall back to a finite difference approximation. Since it is only used in the prediction step for which high accuracy is not needed, this seems acceptable.

4.2.3 Recursive formulas for derivatives of the defining systems for continuation

For the continuation of fixed points and cycles we need the derivatives of (1.5) which can be computed from (4.5) and (4.10). Now, we consider the derivatives of s (as defined in (3.1)) with respect to z, a state variable or parameter. The flip and Neimark-Sacker cases can be handled in a similar way. Let M be the matrix

in (3.2). By taking derivatives of (3.2) with respect to z we obtain

$$M \begin{bmatrix} v_z \\ s_z \end{bmatrix} + \begin{bmatrix} A_z^{(K)} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v \\ s \end{bmatrix} = 0. \tag{4.12}$$

Using (3.4) we obtain

$$s_z = -w^T (A^{(K)})_z v. \tag{4.13}$$

If z represents one of the state variables, then

$$s_{x_i} = -\langle w, B^{(K)}(e_i, v)\rangle, \tag{4.14}$$

as computed in section 4.2. When z is a parameter α_k we can write

$$s_{\alpha_k} = \sum_{J=1}^{K} C_J, \tag{4.15}$$

where

$$C_J = -w^T f_x(x_K) \cdots (f_x(x_J))_{\alpha_k} f_x(x_{J-1}) \cdots f_x(x_1) v, \tag{4.16}$$

where $J = 1, \ldots, K$. In this expression

$$(f_x(x_J))_{\alpha_k} = [f_x(f^{(J)}(x^1, \alpha))]_{\alpha_k} = f_{x\alpha}(x_J, \alpha) + B(x_J) T_J, \tag{4.17}$$

where T_J is a vector, that can be recursively defined by

$$T_J = f_{\alpha_k}(x_{J-1}, \alpha) + A(x_{J-1}) T_{J-1}, \quad T_1 = 0. \tag{4.18}$$

Summarizing, for the computation of s_α we need to compute $f_x, f_{\alpha_k}, f_{xx}, f_{x\alpha_k}$ in all iteration points $x_1, \ldots, x_K$, and given these compute T_J for $J = 1, \ldots K$. Then

$$C_J = -w^T A(x_K) \cdots (f_{x\alpha_k}(x_J) + B(x_J) T_J) A(x_{J-1}) \cdots A(x_1) v, \tag{4.19}$$

and s_{α_k} is computed via (4.15).

4.3 Computing the vector-Hessian-vector and Hessian-vector products

To define the Jacobian system of fold, flip and Neimark-Sacker continuations we need to compute terms of the form vector-Hessian-vector and Hessian-vector where Hessian is a $n \times n \times n$ tensor and vector is a n-vector. These can be computed symbolically by using the recursive formulas derived in Section 4.2.3. Computation of vector-Hessian-vector by using (4.14) and (4.15) are implemented in the files *lpvecthessvect* and *lpvecthesspvect*. These files are used in fold continuation and contained in the folder *LimitPointMap*. The same computations are implemented in *pdvecthessvect* and *pdvecthesspvect* for flip continuation and *nsvecthessvect* and *nsvecthesspvect* for Neimark-Sacker continuation. Furthermore, in switching of some codim 2 bifurcation points, we need to compute expressions of the form Hessian-vector symbolically, where Hessian is a $n \times n \times n$ tensor w.r.t state variables and parameters. This computation can be done by using the recursive formula (4.15) and is implemented in *lphesspvect*, *pdhesspvect* and *nshesspvect* which are contained in the folders *LimitPointMap*, *PeriodDoublingMap*, *NeimarkSackerMap*, respectively.

4.4 Finite difference approximation of directional derivatives

For a general discussion of directional derivatives we refer to [62], $\S 10.3.4$, where it is shown that all computations can be reduced to computing expressions of the form

$$f_x q,\ \ f_{xx} qq,\ \ f_{xxx} qqq,\ \ f_{xxxx} qqqq\ \ ,\ \ f_{xxxxx} qqqqq.$$

Here we only discuss how we choose the increment h in the directional derivatives for a given function $f(x)$. In fact, we want to choose h to minimize the combination of truncation and roundoff error in the computation of the multilinear functions A, B, C, D and E.

To compute $f_x q$ we start from

$$f(x+hq) = f(x) + hf_x q + \frac{h^2}{2} f_{xx} qq + \frac{h^3}{6} f_{xxx} qqq + O(h^4), \tag{4.20}$$

where q is a unit vector. Similarly,

$$f(x-hq) = f(x) - hf_x q + \frac{h^2}{2} f_{xx} qq - \frac{h^3}{6} f_{xxx} qqq + O(h^4). \tag{4.21}$$

Subtracting these two expressions we obtain

$$f(x+hq) - f(x-hq) = 2hf_x q + \frac{h^3}{3} f_{xxx} qqq + O(h^5). \tag{4.22}$$

Dividing (4.22) by $2h$ and rearranging, we find the following expression for the first order derviative:

$$f_x q = \frac{f(x+hq) - f(x-hq)}{2h} - \frac{h^2}{6} f_{xxx} qqq + O(h^4). \tag{4.23}$$

An unavoidable consequence of using numerical differentiation formulas like (4.23) is roundoff error. By taking into account this error and ignoring the $O(h^4)$ term, the approximation formula (4.23) can be written as

$$f_x q = (\frac{f(x+hq) - f(x-hq)}{2h})_{fl} + e_t(h) + e_r(h), \tag{4.24}$$

where $e_r(h)$ is roundoff error, $e_t(h)$ is the truncation error and

$$(\frac{f(x+hq) - f(x-hq)}{2h})_{fl},$$

is the floating point approximation. The total error $E(h)$ is given by

$$E(h) = e_t(h) + e_r(h). \tag{4.25}$$

We can bound the norm of $E(h)$ by

$$||E(h)|| \leq \frac{e_r}{2h} + \frac{h^2 M}{6}, \tag{4.26}$$

where e_r is a bound for the roundoff error made in the subtraction

$$f(x+hq) - f(x-hq),$$

and

$$|f_{xxx}qqq| \leq M.$$

The roundoff error e_r is of order $K\epsilon_m$ where K is the magnitude of f and ϵ_m is machine precision. So we can rewrite (4.26)

$$||E(h)|| \leq \frac{CK\epsilon_m}{2h} + \frac{h^2M}{6}, \tag{4.27}$$

where C is a modest constant. To minimize the error with respect to h, we require

$$\frac{d}{dh}(\frac{CK\epsilon_m}{2h} + \frac{h^2M}{6}) = 0. \tag{4.28}$$

Calculating this derivative gives

$$-\frac{CK\epsilon_m}{2h^2} + \frac{2hM}{6} = 0. \tag{4.29}$$

If we assume that f and $f_{xxx}qqq$ have almost similar size then the optimal choice h_{1min} resulting in a minimum error, is of order

$$h_{1min} \approx (\epsilon_m)^{\frac{1}{3}}. \tag{4.30}$$

When we use double precision, i.e. 16-digit accuracy in the representation of a given number $\epsilon_m \approx 10^{-16}$ then $h_{1min} \approx 10^{-5}$.

A similar procedure can be carried out for evaluating the optimal step size for the second order derivative. We have

$$f_{xx}qq = \frac{f(x+hq) - 2f(x) + f(x-hq)}{h^2}) - \frac{h^2}{12}f_{xxxx}qqqq + O(h^4). \tag{4.31}$$

By taking into account the roundoff error and bounding the absolute value of the total error $E(h)$, we have

$$||E(h)|| \leq \frac{CK\epsilon_m}{h^2} + \frac{h^2}{12}M, \tag{4.32}$$

where K and ϵ_m are defined as before, C is a small number and $|f_{xxxxqqqq}| < M$. Assuming that f and $f_{xxxxqqqq}$ have almost similar size, the optimal choice of $h_{2min} = h_2$ is given by

$$h_{2min} \approx (\epsilon_m)^{\frac{1}{4}}. \tag{4.33}$$

This process can be continued to find the optimal step-size for the third order derivative, we have

$$\begin{aligned} f_{xxxqqq} = & \tfrac{f(x+3hq)-3f(x+hq)+3f(x-hq)-f(x-3hq)}{8h^3} \\ & -\tfrac{237}{960}h^2 f_{xxxxxqqqqq} + O(h^4). \end{aligned} \tag{4.34}$$

By taking into account the roundoff error and bounding the absolute value of total error $E(h)$, we have

$$||E(h)|| \leq \frac{CK\epsilon_m}{8h^3} + \frac{237h^2}{960}M, \tag{4.35}$$

where C, K and ϵ_m are defined as before, and $|f_{xxxxxqqqqq}| < M$.
Assuming that f and $f_{xxxxxqqqqq}$ have almost similar size, then the optimal choice of h_{3min} is given by

$$h_{3min} \approx (\epsilon_m)^{\frac{1}{5}}. \tag{4.36}$$

Similarly, for the fourth and fifth order derivatives we have

$$h_{4min} \approx (\epsilon_m)^{\frac{1}{6}}. \tag{4.37}$$

and

$$h_{5min} = (\epsilon_m)^{\frac{1}{7}}. \tag{4.38}$$

In CL_MATCONTM the default values of the increment *h* for the first, second, third, fourth and fifth order derivatives are defined as follows:

- $Increment = (\epsilon_m)^{\frac{1}{3}}$.
- $hessIncrement = (\epsilon_m)^{\frac{1}{4}}$.

- $tens3Increment = (\epsilon_m)^{\frac{1}{5}}$.
- $tens4Increment = (\epsilon_m)^{\frac{1}{6}}$.
- $tens5Increment = (\epsilon_m)^{\frac{1}{7}}$.

where ϵ_m is the machine precision; we use $\epsilon_m = 10^{-15}$. However, the $Increment$ can be adjusted by the user by setting $cds.options.Increment$. The increments of the higher-order derivatives are then adapted accordingly.

4.5 Automatic differentiation

We now discuss our experience in using automatic differentiation techniques as an aid in the numerical continuation and bifurcation of cycles. In particular, we consider the computation of the multilinear forms in the Taylor expansion up to the fifth order of an iterated map. For our application, due to the needed high accuracy of differentiation, one is forced to turn to automatic differentiation or symbolic approaches. Methods based on finite differences are inaccurate for such computations.

In this section we first give a brief background on Automatic Differentiation (AD) followed by a discussion of the techniques to compute the derivatives of a given function. Then, we present a detailed description of the usage of AD in the computation of multilinear forms that arise in the computation of the normal form coefficients of codim 1 and codim 2 bifurcation of cycles. At the end, we present some numerical results and present a comparison of time complexity using AD and SD (Symbolic Differentiation) in our application.

4.5.1 Automatic differentiation background

In the standard reference for the subject, Griewank [45] states that Algorithmic, or Automatic Differentiation (AD) is concerned with the accurate and efficient evaluation of derivatives for functions defined by computer programs. AD uses the systematic application of the chain rule for differentiation applied to the floating point representation of a variables value and its derivatives. Unlike in the finite-difference approximation, no discretization or cancellation errors occur, and the

resulting derivative values are accurate to within floating-point round-off. Since only floating point values are used (unlike differentiation within symbolic algebra packages such as Mathematica or Maple), good efficiency may be obtained. Additionally, AD permits the use of control structures (loops, branches, and subfunctions) common to modern computer languages but not easily amenable to symbolic differentiation. To compute derivatives symbolically using computer algebra software, an enormous expression growth normally occurs due to a repeated evaluation of common sub-expressions. On the other hand, with finite difference approximations, the accuracy of the derivatives is restricted because of cancellation and truncation errors, particularly, for high-order derivatives.

4.5.2 Forward mode and implementation of AD

The technique of automatic differentiation is conceptually based on first transforming a given computer code into a straight-line code. That is, after this preprocessing step, the code is a finite sequence of elementary operations without loops, conditionals, branching, or subroutines.

Take as an example a function $f : \mathbb{R}^5 \mapsto \mathbb{R}^5$ mapping a five-dimensional vector x to a five-dimensional vector y. For the sake of simplicity, assume that the corresponding straight-line code takes an array $x(1:5)$ as input and produces an output array $y(1:5)$ making use of an eight-dimensional array $t(1:8)$ to store intermediate values. The straight-line code is given in Figure 4.1, where an independent variable $x(i)$ is initialized with a value c_i and the symbol $\odot$ is used to denote any binary elementary function available in a given programming language, for instance multiplication or addition (example taken from [18]).

Notice that we assume that the straight-line code is given in single-assignment form, i.e., each intermediate variable has only one assignment. This assumption may lead to tremendous growth of the number of intermediate variables but preserves uniqueness of the left-hand sides.

A straight-line code is commonly represented by a directed acyclic graph as follows. A node is associated with every statement of the straight-line code or, equivalently, with every left-hand side variable. There is an edge from node i to j whenever variable i is an input to variable j. This computational graph represents the data dependence when evaluating the code for a given set of inputs.

	$t(1) \longleftarrow x(1) \odot x(2)$	
	$t(2) \longleftarrow t(1) \odot x(3)$	
$x(1) \longleftarrow c_1$	$t(3) \longleftarrow x(2) \odot x(4)$	$y(1) \longleftarrow t(2) \odot t(2)$
$x(2) \longleftarrow c_2$	$t(4) \longleftarrow x(2) \odot x(5)$	$y(2) \longleftarrow t(2) \odot t(7)$
$x(3) \longleftarrow c_3$	$t(5) \longleftarrow t(3) \odot t(4)$	$y(3) \longleftarrow y(2) \odot t(7)$
$x(4) \longleftarrow c_4$	$t(6) \longleftarrow t(2) \odot t(5)$	$y(4) \longleftarrow t(6) \odot t(8)$
$x(5) \longleftarrow c_5$	$t(7) \longleftarrow t(2) \odot t(6)$	$y(5) \longleftarrow t(5) \odot t(8)$
	$t(8) \longleftarrow t(5) \odot t(7)$	

Table 4.1: The straight-line code of a simple function with $n = 5$ independent scalar variables $x(1), x(2), \ldots, x(5)$ and $m = 5$ dependent scalar variables $y(1), y(2), \ldots, y(5)$.

Think of the computational graph as a visualization of the data transmitted along its edges in the direction indicated by its arrows. The roots of the graph represent the independent variables and the leaves represent the dependent variables. An example of the computational graph associated with the straight-line code of Table 4.1 is depicted in Figure 4.1. From a practical point of view, the number of nodes roughly corresponds to the number of floating point operations $\odot$ used to evaluate the function. This is the reason why computational graphs in automatic differentiation are typically quite large. In AD, the edges of the computational graph are suitably weighted by partial derivatives. From this initial scenario, the Jacobian may be calculated by eliminating the intermediate vertices of the computational graph one at a time according to certain rules manipulating the edge weights. Eventually, the graph will be bipartite with edges exclusively connecting independent and dependent variables. A final weight of an edge connecting the independent variable i with the dependent variable j corresponds to the partial derivative of variable j with respect to variable i.

Two standard elimination sequences are known. The Forward Mode eliminates the intermediate vertices starting from the vertices next to the independent variables, escorting the data flow of the computation of the function along the direction of the edges of the computational graph, and reaching the intermediate vertices next to the dependent variables at last.

The Reverse Mode will not be used in this thesis, so we do not further describe it.

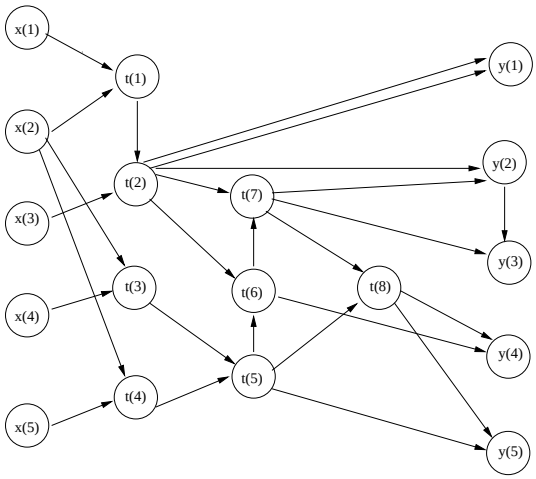


Figure 4.1: The computational graph associated with the code given in Table 4.1.

AD is implemented in one of two ways: operator overloading or source transformation, Griewank [45], Ch 5. The operator overloading approach takes advantage of the facility to define new classes (or types) within modern computer languages such as Fortran 95, C++ or MATLAB. Objects of the new AD class are defined to have a component which stores their value and components to store derivative information. Arithmetic and intrinsic functions are extended to the AD class making use of operator and function overloading. In typed languages, such as Fortran or C++, all that remains is for the user to redefine the classes of all relevant objects within the function and all subfunctions to that of the AD class, initialize appropriate values and derivatives, invoke the function, and then extract the values of the derivatives. Representative examples of such implementations are the packages ADO1, [87] and ADOL-C, [46]. In MATLAB there are a variety of intrinsic classes with associated functions and operations. We use the object-oriented programming features of MATLAB to introduce new classes.

We use the known formulae for differentiating elementary functions, together with the chain rule, to build up the needed derivatives of an arbitrary f (f is

expressed by a computer program). We assume f is a vector function $y = f(x)$ over the reals with n real *inputs*, or independent variables, $x = (x_1, \ldots, x_n)$ and m real *outputs* $y = (y_1, \ldots, y_m)$. The code for f may contain branches and loops. However, each evaluation of f at given inputs x can be written as a *code list*, which is a finite sequence of assignments of the simple form

$$v_i = e_i(\text{previously defined } v_j\text{'s, or constants}), \quad i = 1, 2, \ldots, p+m, \tag{4.39}$$

where each e_i is one of the elementary functions. The v_i are called *variables*. In (4.39) it is convenient to use $v_{1-n}, \ldots, v_0$ as aliases for the inputs $x_1, \ldots, x_n$ and $v_{p+1}, \ldots, v_{p+m}$ as aliases for the outputs $y_1, \ldots, y_m$, following the notation of [45]. The remaining variables $v_1, \ldots, v_p$ are called *intermediate*.

The *forward mode* of AD is the simplest, and is appropriate to our application. Each v_i is represented by an object $\mathtt{v}_i$ of a data type that holds not just the value but some needed set of derivatives. For our purposes:

- There is (at any one point) just one variable being treated as independent: we call it t. Thus each variable v_i is regarded as a function of t.

- The data structure holds *Taylor coefficients* (TCs), that is the coefficients of the truncated Taylor series of v_i, up to some order p, expanded about some point $t = a$. Changing to a new independent variable $s = t - a$:
  v_i holds $(v_{i,0}, v_{i,1}, \ldots v_{i,p})$ where

$$v(a + s) = v_{i,0} + v_{i,1}s + \cdots + v_{i,p}s^p + O(s^{p+1}).$$

We call the data type `adtayl`. It is helpful to think of an `adtayl` object as representing an *infinite* power series which is only known up to the order p term. The Taylor coefficients are of course just scaled derivatives, and are simpler to manipulate than are derivatives.

When evaluating a code list, each elementary operation on real arguments is replaced by the corresponding operation on `adtayl` arguments regarded as power series known up to a certain order. Consider first the four basic arithmetic operations. Agree that `a` holds $(a_0, a_1 \ldots)$, and so on for other named variables.

Let a be defined to order p, and b to order q. Then c = a + b and d = a × b are defined to order $r = \min(p, q)$ by

$$c_i = a_i + b_i,$$
$$d_i = a_0 b_i + a_1 b_{i-1} + \cdots + a_i b_0,$$

and similarly for a − b, and for a ÷ b provided $b_0 \neq 0$.

For example, suppose $y = (2 + t)(3 + t^2)$ and we wish to obtain the power series of y up to order $p = 2$, expanded about the point $t = 1$. We initialise the process by creating the object representing the independent variable expanded to order 2 in terms of $s = t - 1$:

$$\texttt{t} = (t_0, t_1, t_2) = (1, 1, 0) \text{ representing } 1 + 1s + 0s^2 \ ,$$

We create objects c2 and c3 representing the constant functions 2 and 3 respectively. The whole computation is shown in the following table.

Computation	Holds	Represents
t=indep	$(1, 1, 0)$	$t = 1 + s$
c2=const	$(2, 0, 0)$	2
c3=const	$(3, 0, 0)$	3
v1=c2 + t	$(3, 1, 0)$	$2 + t = 3 + s$
v2=t * t	$(1, 2, 1)$	$t^2 = 1 + 2s + s^2$
v3=c3 + v2	$(4, 2, 1)$	$3 + t^2 = 4 + 2s + s^2$
outputy=v1 * v3	$(12, 10, 5)$	$(2 + t)(3 + t^2) = 12 + 10s + 5s^2 + O(s^3)$

For applying the standard functions $\exp, \cos, \ldots$ to power series there are various formulas in the literature. We have aimed to choose ones that can be made reasonably fast in MATLAB, especially when the argument is a vector of power series, not just a single one. This is not the place for details but we give a few examples.

First, multiplication of power series is a convolution, which can be realised by the very fast built-in filter function of MATLAB.

Second, the method for exp(a) exploits the fact that if $c(t) = \exp(a(t))$ then $c'(t) = a'(t)c(t)$, which is converted to the integral form

$$c(t) = c_0 + \int_0^t a'(s)c(s)\delta s,$$

where $c_0 = e^{a_0}$. In terms of the coefficients this reduces to a triangular linear system, which is fast in MATLAB.

Third, for $\sin(\mathtt{a})$ and $\cos(\mathtt{a})$, it is convenient to compute both simultaneously as the real and imaginary parts of $\exp(i\,\mathtt{a})$, and to record both results, along with the argument `a`, in persistent storage. Each time a new `a` is given, it is checked against the recorded one. If they are equal, the result can be returned at once. Since $\cos$ and $\sin$ at the same argument often occur together in applications, this reduces the cost for these functions by nearly half.

Regarding previous work to provide AD facilities in MATLAB, Rich and Hill [93] provided a limited facility for MATLAB that enabled AD of simple arithmetic expressions defined by a character string. Such strings, together with necessary values of variables were passed to an external routine, written in turbo-C, for differentiation. However, the first significant work was that of Coleman and Verma [21, 22], and Verma [22] who produced an operator-overloading AD package named ADMAT that provides facilities for forward and reverse mode AD for both first and second derivatives and run-time Jacobian sparsity detection. The MAD package, see [100], facilitates the evaluation of first derivatives of multidimensional functions that are defined by computer codes written in MATLAB. The underlying algorithm is the well-known forward mode of automatic differentiation implemented via operator overloading on variables. However, none of these tools handles Taylor series.

4.6 The Taylor series class

We use the object-oriented programming features of MATLAB to introduce the new class `adtayl` in the bifurcation software CL_MATCONTM. A MATLAB class consists of a set of functions that create and manipulate objects of this class. Here the manipulations that concern us are extending the arithmetic operations of MATLAB to those that calculate both an object's value and an associated directional derivative. Whenever MATLAB encounters objects of these classes, for example when two such objects are matrix-multiplied, it will not use the standard times function designed for objects of class double, but instead will use the times function defined in our new class.

4.6.1 The MATLAB `adtayl` class

The `adtayl` data type was implemented as a MATLAB class of the same name. An `adtayl` object `x` has one field `tc`. Here `x` can be a scalar, a vector or a matrix. In the scalar case, `tc` is a row vector of length $(p+1)$ holding the TCs $x_0, x_1, ...x_p$ of a variable $x = x(t)$ around a point $t = a$. MATLAB arrays are numbered from 1, so x_r is in position $r+1$ of `tc` for each r. In general, `tc` holds an $m \times n \times (p+1)$ array with the obvious meaning, with the TCs always along the third dimension. Thus $m = n = 1$ for the scalar case, and $m = 1$ or $n = 1$ for a row vector or column vector respectively.

One cannot create a general series (1) directly. `adtayl` creates the TS of the independent variable t, that is

$$t = a + 1s + 0s^2 + ... + 0s^p,$$

and all other functions must be calculated from that.

One can create constant-functions, after creating the independent variable. Suppose `cval` holds the numeric value c. Then

$$\texttt{c} = \texttt{adtayl}(\texttt{cval});$$

sets `c` to the `adtayl` object representing $c+0s+...+0s^p$, where p comes from the current (most recently created) independent variable. `cval` can be a scalar or a one or two dimensional array, creating an `adtayl` object of the same shape. Such named constants are not needed often, as real constants in arithmetic expressions are converted automatically. For instance the calculation in the example given above can be done by the MATLAB statements

```
>>  t = adtayl(1,2);
>>  (2+t)*(3+t*t)
```

and the result will be displayed in the command window:

```
 Coefficients of orders 0 to 2 are:
    12    10     5
>>
```

All the operations are element-wise, so multiplication and division are the MATLAB operations `.*` and `./` (there is no `.\`). Matrix operations are not supported at all: the matrix multiplication operator `*` only works for scalar objects, being just a case of `.*`, and similarly for matrix division, power etc.

The following MATLAB standard functions are implemented for the `adtayl` class:

> `sqrt`, `exp`, `log`, `log10`, `sin`, `cos`, `tan`, `cotan`, `sec`, `csec`, `asin`, `acos`, `atan`, `acotan`, `asec`, `acsec`, `sinh`, `cosh`, `tanh`, `cotanh`, `sech`, `csech`, `asinh`, `acosh`, `atanh`, `acotanh`, `asech`, `acsech`.

They do not all handle vector/matrix arguments efficiently, and in some cases do not do so at all. For the applications to date, only scalar values were needed. However, the following MATLAB housekeeping functions are implemented for vector/matrix arguments.

- `display` prints an object in the command window.

- The class supports standard MATLAB array subscripting for referencing (`subsref`) and assigning (`subsasgn`) elements, or sections, of arrays, and assembling arrays using MATLAB's square bracket notation (`horzcat`, `vertcat`). E.g. `[t y; 1+t y*y]` creates a 2 by 2 matrix of Taylor series. Higher-dimensional arrays are not supported, and subscripting applies to the m and n directions only.

- Functions `size`, `numel`, and `end` are overloaded to give the correct behaviour of array accesses.

- By design, one cannot access the TC dimension with the above functions. There is an `order` function that returns the order p of an object; and a `tcs` function that extracts its TCs as an $[m, n, p+1]$ array. If m or n is 1, the singleton dimension is "squeezed" out to give a normal 2D array, which is easier to manipulate and display.

4.6.2 The `adtayl` constructor function

Below, we show the `adtayl` constructor function. The call `adtayl` (a,p) calculates Taylor series coefficients up to order p. This function takes two inputs a and p and returns an object of the `adtayl` class. The input a is a constant scalar or array where the Taylor series is computed around a, while p is a constant integer or constant string 'const' where p indicates the order of the Taylor series.

```
function obj =  adtayl(a, p)

global ADTAYLORDER
switch nargin
  case 0
    %Not documented: system can use it for anonymous initialising.
    if isempty(ADTAYLORDER)
      error('Invalid use of ADTAYL constructor: no order p was set')
    else
      tc = zeros(1,1,ADTAYLORDER+1);
    end
  case 1
    if isa(a,'adtayl') %Not documented: just copies an ADTAYL object
      obj = a;
      return
    elseif ~(isnumeric(a) && ndims(a)==2) || isempty(ADTAYLORDER)
      error('Wrong use of ADTAYL constructor with one argument')
    else %Set a constant
      [m,n] = size(a); % A is now a constant scalar or array
      p = ADTAYLORDER;
      tc = reshape([a(:); zeros(m*n*p,1)], [m,n,p+1]);
    end
  case 2
    if ~(isnumeric(p) && isnumeric(a) ...
        && isscalar(a) && isscalar(p) && p==fix(p) && p>=0)
      error('Invalid input A or P to construct independent variable')
    else
      % Store chosen Taylor order as a global variable:
      ADTAYLORDER = p;
      if p==0
        tc = a;
```

```
      else
        tc = [a, 1, zeros(1,p-1)];
      end
      tc = reshape(tc, [1 1 p+1]);
    end
  otherwise
    error('Wrong number of arguments to ADTAYL')
end

obj = class(struct('tc',tc),'adtayl');
```

4.7 Computing multilinear forms

In this section, we describe how to use the `adtayl` class to calculate the multilinear forms that arise in the normal form coefficients of codim 1 and codim 2 bifurcation of cycles. The sign and size of these coefficients determine the bifurcation scenario near a local bifurcation point. We consider the multilinear forms $B(q_1, q_2)$, $C(q_1, q_2, q_3)$, $D(q_1, q_2, q_3, q_4)$ and $E(q_1, q_2, q_3, q_4, q_5)$ of order 2, 3 , 4 and 5, respectively, as they are defined in section 2.1. These multilinear forms can be computed by using the `adtayl` class and computing directional derivatives that are stored as an array of class *double*.

4.7.1 Computing the forms by AD

These multilinear forms can be computed by using the `adtayl` class and computing directional derivatives that are stored as an array of class `double`. We first define a function that iterates the map f a desired number of times. Here the argument `func` is (the function-handle of) the map f.

```
function y1 = Tmap(func,x0,h,par,taylorder,J)
  s = adtayl(0,taylorder); %Base point & Taylor order
  y1= x0 + s*h;
  for i=1:J
    y1 = func(0, y1, par{:});
  end
```

We now give the code for `multilinear1AD` and `multilinear2AD`,

which compute $A^{(J)}(q_1)$ and $B^{(J)}(q_1, q_2)$ respectively, using `Tmap`. In their input lists, `q1` and `q2` are the n-vectors q_1, q_2, where n depends on the map. `x0` and `par` are the vector of state variables and of parameter values respectively, at the bifurcation point. `J` is the iteration number for the map. Similar code can be used for the higher-order multilinear forms.

```
function ytayl1 = multilinear1AD(func,q1,x0,par,J)
  taylorder = 1;
  y1 = Tmap(func,x0,q1,par,taylorder,J);
  ytayl1 = tcs(y1);

function ytayl2 = multilinear2AD(func,q1,q2,x0,par,J)
  taylorder = 2;
  if q1==q2
    y1 = Tmap(func,x0,q1,par,taylorder,J);
  else
    y11 = Tmap(func,x0,q1+q2,par,taylorder,J);
    y12 = Tmap(func,x0,q1-q2,par,taylorder,J);
    y1 = 1/4.0*(y11-y12);
  end
  ytayl2 = tcs(y1);
```

At the end, $A^{(J)}(q_1)$ is the last column of `ytayl1`, that is `ytayl1(:,end)`; and $B^{(J)}(q_1, q_2)$ is twice the last column of `ytayl2`, namely `2*ytayl2(:,end)`.

In the definition of `ytayl2` we used the polarization identity

$$B(u, v) = \frac{1}{4}(B(u + v, u + v) - B(u - v, u - v)).$$

where B is any bilinear form. Similar identities exist for the higher order forms, see [62], §10.3.4.

It only remains to provide code for a specific map f. The "Cod Stock"model in Test case 2 in §4.8 may be coded as follows:

```
function y = CodStockFunc(t,x,F,P,beta1,beta2,beta3,mu1,mu2)
  x1 = x(1); x2 = x(2);
  y = [ F*exp(-beta1*x2)*x2 + (1-mu1)*exp(-beta2*x2)*x1;
        P*exp(-beta3*x2)*x1 + (1-mu2)*x2 ];
```

To evaluate $A^{(J)}(q_1)$ at specific q_1 and q_2 one can type the following at the MATLAB command line. Note `q1` and `x0` are ordinary column vectors, while `par` is a MATLAB cell-array, which is transformed into part of the list of arguments to the map; it holds $F, P, \beta_1, \beta_2, \beta_3, \mu_1, \mu_2$ in that order. `@CodStockFunc` is the function-handle for the `CodStockFunc` function.

```
>> par={399.5681,0.5, 1,1,1, 0.5,0.444715}
>> x0=[26.0; 3.0]
>> q1=[1;2]
>> q2=[3;4]
>> J=15
>> ytayl1=multilinear1AD(@CodStockFunc,q1,x0,par,J)
>> ytayl2=multilinear2AD(@CodStockFunc,q1,q2,x0,par,J)
>> A=ytayl1(:,end)
>> B=2*ytayl2(:,end)
```

When the above was run it produced these results:

```
ytayl1 =
   2.3896e+01   7.7278e+01
   3.0477e+00  -3.1623e-01
ytayl2 =
            0   7.7688e+01   1.9340e+03
            0  -3.1796e-01   7.4153e-01
A =
   7.7278e+01
  -3.1623e-01
B =
   3.8680e+03
   1.4831e+00
>>
```

4.7.2 Comparison with symbolic derivatives

If the symbolic toolbox of MATLAB is available then we can compute derivatives of (1.2), using recursive formulas, see §4.2. Here follows a brief description of the recursive formulas, since we need to refer to them. The iteration of (1.2) gives rise to a sequence of points

$$\{x^1, x^2, x^3, \ldots, x^{K+1}\},$$

where $x^{J+1} = f^{(J)}(x^1, \alpha)$ for $J = 1, 2, \ldots, K$. Suppose that symbolic derivatives of f up to order 5 can be computed at each point. Suppose that symbolic derivatives of f up to order 5 can be computed at each point. A drawback of using these recursive formulas is the nonlinear growth rate of the time complexity when the iteration number J increases. To make it clear, we use e_1, e_2, e_3, e_4 and e_5 to indicate the complexity of computation of the multilinear functions, i.e.

$$\begin{aligned} e_1 &= e(Aq), \\ e_2 &= e(B(q_1, q_2)), \\ e_3 &= e(C(q_1, q_2, q_3)), \\ e_4 &= e(D(q_1, q_2, q_3, q_4)), \\ e_5 &= e(E(q_1, q_2, q_3, q_4, q_5)). \end{aligned}$$

Then complexity for the multilinear forms up to the fifth order using the recursive formulas (4.4), (4.6), (4.7), (4.8) and (4.9), respectively, is given by:

$$\begin{array}{lcl} e(A^{(J)}q) & = & Je_1 \\ e(B^{(J)}(q_1, q_2)) & = & (J-1)(J+1)e_1 + Je_2 = J^2e_1 + Je_2 + \text{l.o.t}, \\ e(C^{(J)}(q_1, q_2, q_3)) & = & J^3e_1 + J^2e_2 + Je_3 + \text{ l.o.t}, \\ e(D^{(J)}(q_1, q_2, q_3, q_4)) & = & J^4e_1 + J^3e_2 + J^2e_3 + Je_4 + \text{ l.o.t}, \\ e(E^{(J)}(q_1, q_2, q_3, q_4, q_5)) & = & J^5e_1 + J^4e_2 + J^3e_3 + J^2e_4 + Je_5 + \text{ l.o.t}, \end{array}$$

where l.o.t stands for 'lower order terms'.

4.8 Test cases

In this section, we present test cases to compare the accuracy and speed of AD and SD in the computation of the normal form coefficients of bifurcations of cycles in CL_MATCONTM.

4.8.1 Test case 1

We consider a 2-dimensional difference equation with 3 parameters:

$$M_K : \begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} Sx - y - (\epsilon y^2 + x^2) \\ Lx - (y^2 + x^2)/5 \end{pmatrix} \tag{4.40}$$

J	bifurcation point
3	$(0.375974, -0.627941, 1.335343)$
6	$(0.349755, -0.948447, 1.436871)$
12	$(0.336832, -0.881886, 1.463676)$
24	$(0.330787, -0.857379, 1.469354)$
48	$(0.329173, -0.852556, 1.470561)$
96	$(0.3391975, -0.925268, 1.470819)$
192	$(0.339223, -0.925734, 1.470874)$

Table 4.2: PD bifurcation points of iterates of M_K. The left column contains the iteration numbers.

where ϵ, L and S are parameters (unpublished PhD thesis of A. Yu. Kuznetsova, Saratov university).

$M_K^{(3)}$ has a fixed point at $(x^*, y^*) = (0.37588802742303, -0.62783638474655)$ when the parameter values are given by $(\epsilon, L, S) = (1, 1.3353, -0.799600)$. Continuation of fixed points of the third iterate, with ϵ free and keeping L, S fixed, leads to a supercritical flip bifurcation point when $\epsilon = 1.335343$. The map $M_K^{(3)}$ has a cascade of flip points that can be computed by switching to the new branches of double period at the PD points. We compute the PD points of the order 3, 6, 12, 24, 48, 96, 192 and then compute their successive NFCs. The coordinates of the PD points (x, y, ϵ) are given in Table 4.2. The computed values and elapsed time are given in Table 4.3 and depicted in Figure 4.2.

In the case of AD, the elapsed time grows linearly with the iteration number whereas in the case of SD the elapsed time grows much faster. However, for the iterates of low order 3, 6, 12, SD is still faster than AD, with crossover at around iteration 24.

We continue the flip bifurcation curves of M_K starting from the computed PD point with L and S as bifurcation parameters, keeping ϵ fixed, and detect codimension-2 bifurcation points (LPPD). Their coordinates (x, y, L, S) are given in Table 4.4. The computed NFCs $a(0)$ and $b(0)$ and elapsed time are given in Table 4.5 and depicted in Figure 4.2. It is clear from Figure 4.2 that the growth of

J	SD	t	AD	t	FD	t
3	5.662603e+3	0.041	5.6662603e+3	0.074	-3.590146e+4	0.041
6	8.753606e+1	0.047	8.753606e+1	0.086	8.753396e+1	0.045
12	5.807277e+2	0.080	5.807277e+2	0.135	5.800071e+2	0.046
24	2.080773e+4	0.225	2.080773e+4	0.222	8.347385e+3	0.054
48	6.023199e+5	1.13	6.023199e+5	0.401	-6.915118e+4	0.072
96	4.881335e+6	8.00	4.881335e+6	0.764	-5.112090e+5	0.107
192	1.969764e+8	66.6	1.969764e+8	2.90	-15.55230e+7	0.269

Table 4.3: Computed NFC b and elapsed time in the computation for a cascade of PD points.

J	bifurcation point
3	$(-0.089643, -0.664722, 1.249136, -1.474424)$
6	$(0.297387, -0.923383, 1.468821, -0.746244)$
12	$(0.310548, -0.873006, 1.504478, -0.731940)$
24	$(0.316667, -0.855444, 1.510727, -0.729198)$
48	$(0.317656, -0.850743, 1.512561, -0.728413)$
96	$(0.296337, -0.907914, 1.512765, -0.728324)$

Table 4.4: LPPD bifurcation points (x, y, L, S) of iterates of M_K. The left column contains the iteration numbers.

elapsed time in the computation of NFCs of the LPPD points is apparently linear for AD and far more rapid for SD, again with crossover at around iteration 24.

We remark that the finite difference approximations to $a(0)$ and $b(0)$ are rather good; this is probably due to the fact that $a(0)$ and $b(0)$ depend only on second and lower order derivatives of $f^{(J)}$, while b in Table 4.3 depends on third order derivatives.

4.8.2 Test case 2

The roots of the present map (4.41) can be found in [19, 24, 26]. It is two-dimensional with seven parameters described in [104], as follows.

$$M_{CS} : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} Fe^{-\beta_1 x_2} x_2 + (1-\mu_1) e^{-\beta_2 x_2} x_1 \\ Pe^{-\beta_3 x_2} x_1 + (1-\mu_2) x_2 \end{pmatrix}, \tag{4.41}$$

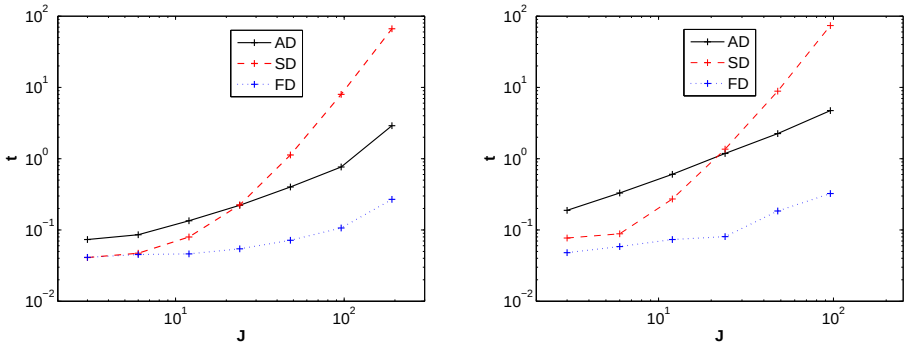


Figure 4.2: Elapsed times, in seconds, as a function of the iteration number, for normal form computations using SD, AD, and FD. Left: for the PD points. Right: for the LPPD points.

where x_1 and x_2 are the immature and mature parts of the cod stock (at some time t) respectively and $F, P, \beta_1, \beta_2, \beta_3, \mu_1$ and μ_2 are dynamics parameters. Overall dynamical behavior of M_{CS} is studied in §5.2.

$M_{CS}^{(3)}$ has a fixed point at $X^* = (x_1^*, x_2^*) = (26.16934, 3.04173)$ for $F = 399.5861, \mu_1 = 0.5, \mu_2 = 0.444715, \beta_1 = \beta_2 = \beta_3 = 1$ and $P = 0.5$. We continue the fixed point of $M_{CS}^{(3)}$ starting from X^* with free parameter μ_2 and find a cascade of period doubling points that can be computed by switching to new branches of period 6, 12, 24, etc. We compute the NFC of the PD points of iterates 3, 6, 12, 24, 48, 96 and compare the speed of SD and AD. The results are given in Figure 4.3. As in test case 1, the elapsed time grows apparently linearly for AD and much faster for SD, with crossover at around iteration 24.

J	SD	t	AD	t	FD	t
3	2.715094e+0 1.173625e+0	0.077	2.715094e+0 1.173625e+0	0.186	2.715093e+0 1.173625e+0	0.048
6	3.814972e+0 1.275108e+1	0.088	3.814972e+0 1.275108e+1	0.330	3.814978e+0 1.275108e+1	0.059
12	1.783747e+0 –7.907552e+1	0.272	1.783747e+0 –7.907552e+1	0.604	1.783765e+0 –7.907370e+1	0.073
24	7.404406e–1 2.184766e+3	1.36	7.404406e–1 2.184766e+3	1.18	7.404697e–1 2.185171e+3	1.081
48	3.603919e–1 –8.539430e+4	8.86	3.603919e–1 –8.539430e+4	2.25	3.601789e–1 –8.542193e+4	0.184
96	1.337659e–1 8.791200e+5	73.7	1.337659e–1 8.791200e+5	4.73	1.340109e–1 8.802758e+5	0.325

Table 4.5: Computed coefficients and elapsed time in the computation of normal form coefficients of iterates of M_K at LPPD bifurcation points.

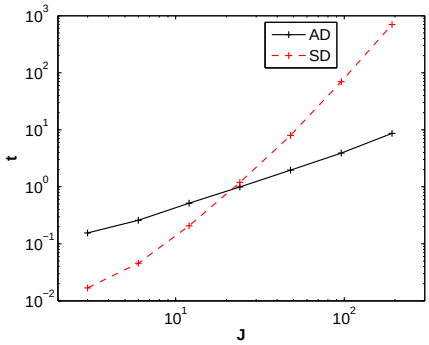


Figure 4.3: Elapsed time in the computation of normal forms of PD points, using SD and AD for the map M_{CS}.

Chapter 5

Applications in biology

Population dynamics is the study of marginal and long-term changes in the numbers, individual weights and age composition of individuals in one or several populations, and biological and environmental processes influencing those changes. Applications of discrete dynamical systems experienced enormous growth in this area, see e.g. [71]. Software tools play an important role in the numerical study of these systems.

In this chapter we study two population models, namely a Leslie-Gower competition model for the interaction of two different species of the flour beetle Tribolium and an age-structured cod stock model. By using CL_MATCONTM we study the dynamical behavior of these systems. In particular, we compute stability domains of the low-order iterations of corresponding maps.

Results from this chapter were published or accepted for publication in [41, 42].

5.1 A Leslie-Gower competition model in biology

5.1.1 Introduction

The classic principle of competitive exclusion requires, for the coexistence of two species, that competitive interference be low. In competition models this

requirement generally means that those coefficients which measure the intensity of the inter-specific competition, the competition between two or more different species for some limited resource, be sufficiently small (usually in relation to the coefficients measuring intraspecific, same species, competition). Put another way, large values of inter-specific competition coefficients imply that one of the species necessarily goes extinct. This form of the principle finds its most forceful and straightforward expression in the famous Lotka-Volterra system of differential equations [11]. In the early development of competition theory, controlled laboratory experiments played a significant role in establishing the competitive exclusion principle. Among these were the famous experiments performed by G. F. Gause [36] and by T. Park [81, 82, 83, 84]. Both worked within the framework of the dynamic scenarios of Lotka-Volterra theory.

The Leslie-Gower model is a discrete time analog of the competition Lotka-Volterra model and is known to possess the same dynamic scenarios as that famous model. The Leslie-Gower model played a significant role in the history of competition theory in its application to classic laboratory experiments of two competing species of flour beetles. While these experiments generally supported what became the Competitive Exclusion Principle, Park observed an anomalous coexistence case. Recent literature has discussed Parks coexistence case by means of non-Lotka-Volterra model with life cycle stages. We study this dynamics by means of a model involving only two species each with two life cycle stages, i.e., juvenile and adult classes.

5.1.2 The model and its fixed points

The roots of the present model can be found in [65], [66] and [31]. Roughly speaking, it was found in biological experiments that two species of flour beetles can coexist under strong inter-specific competition. This was rather unexpected at the time and several models were built to explain this phenomenon. One of the ideas in [31] and [102] is to use an age-structured competition model. For general background we refer to [16].

The model that we use is a four-dimensional map M_{LG} (5.1) with 14 parameters described in [102]. It is a Leslie-Gower competition model for the interaction between the juveniles (j) and adults (a) of one species of the flour beetle Tribolium

and the juveniles (y) and adults (z) of another species for the same food.

$$M_{LG}:\begin{pmatrix} j \\ a \\ y \\ z \end{pmatrix} \longmapsto \begin{pmatrix} j_+ \\ a_+ \\ y_+ \\ z_+ \end{pmatrix} = \begin{pmatrix} \dfrac{b_1 a_t}{1 + c_{jj} j_t + c_{ja} a_t + c_{jy} y_t + c_{jz} z_t} \\ (1-\mu_j) j_t + (1-\mu_a) a_t \\ \dfrac{b_2 z_t}{1 + c_{yj} j_t + c_{ya} a_t + c_{yy} y_t + c_{yz} z_t} \\ (1-\mu_y) y_t + (1-\mu_z) z_t \end{pmatrix}. \tag{5.1}$$

Each species has its own juvenile recruitment rate $b_1 > 0$, $b_2 > 0$, juvenile death rate μ_j and μ_y, and adult death rate μ_a and μ_z. For biological reasons we have

$$0 < \mu_j, \mu_a, \mu_y, \mu_z < 1. \tag{5.2}$$

The other coefficients $c_{jj}, c_{ja}, c_{jy}, c_{jz}$ and $c_{yj}, c_{ya}, c_{yy}, c_{yz}$ describe the competition. They are all strictly positive. By assumption, competition does not affect the adults of either species. In the present study, as in [102], we will study the influence of the coefficients c_{yj} and c_{jy} on the behavior of M_{LG} in a case where all other parameters are fixed. In other words, we study the influence of the competition between juveniles if all other parameters are fixed.

We first look for the fixed points (j^*, a^*, y^*, z^*) of the map (5.1), i.e. solutions of:

$$\begin{pmatrix} j^* \\ a^* \\ y^* \\ z^* \end{pmatrix} = \begin{pmatrix} \dfrac{b_1 a^*}{1 + c_{jj} j^* + c_{ja} a^* + c_{jy} y^* + c_{jz} z^*} \\ (1-\mu_j) j^* + (1-\mu_a) a^* \\ \dfrac{b_2 z^*}{1 + c_{yj} j^* + c_{ya} a^* + c_{yy} y^* + c_{yz} z^*} \\ (1-\mu_y) y^* + (1-\mu_z) z^* \end{pmatrix}, \tag{5.3}$$

From the second and last equations of this system we obtain the relations:

$$\begin{aligned} a^* &= \tfrac{1-\mu_j}{\mu_a} j^*, \\ z^* &= \tfrac{1-\mu_y}{\mu_z} y^*. \end{aligned} \tag{5.4}$$

As a consequence, j^*, a^* are either both zero or both nonzero. Similarly, y^*, z^* are both zero or both nonzero. The trivial vector $(0,0,0,0)^T$ is always a solution of (5.3) but is of little interest.

We first consider 'horizontal' fixed points, i.e. fixed points of the form $(j^*, a^*, 0, 0)$. From (5.3) it follows easily that they must satisfy

$$j^* = \frac{b_1(1-\mu_j) - \mu_a}{\mu_a c_{jj} + c_{ja}(1-\mu_j)}, \tag{5.5}$$

$$a^* = \frac{1-\mu_j}{\mu_a} j^* = \frac{b_1(1-\mu_j)^2 - \mu_a(1-\mu_j)}{\mu_a(\mu_a c_{jj} + c_{ja}(1-\mu_j))}, \tag{5.6}$$

with $(j^*, a^*) > 0$ (i.e. j^*, a^* are biologically meaningful) iff

$$b_1 \frac{1-\mu_j}{\mu_a} > 1. \tag{5.7}$$

Similarly, there are unique 'vertical' fixed points of the form $(0, 0, y^*, z^*)$ given by

$$y^* = \frac{b_2(1-\mu_y) - \mu_z}{\mu_z c_{yy} + c_{yz}(1-\mu_y)}, \tag{5.8}$$

$$z^* = \frac{1-\mu_y}{\mu_z} j^* = \frac{b_2(1-\mu_y)^2 - \mu_z(1-\mu_y)}{\mu_z(\mu_z c_{yy} + c_{yz}(1-\mu_y))}. \tag{5.9}$$

These are biologically meaningful if $(y^*, z^*) > 0$, i.e. iff

$$b_2 \frac{1-\mu_y}{\mu_z} > 1. \tag{5.10}$$

The general form of the Jacobian matrix of our model is:

$$\begin{pmatrix} \frac{-b_1 c_{jj} a^*}{\beta_1^2} & \frac{b_1\beta_1 - b_1 a^* c_{ja}}{\beta_1^2} & \frac{-b_1 c_{jy} a^*}{\beta_1^2} & \frac{-b_1 c_{jz} a^*}{\beta_1^2} \\ 1-\mu_j & 1-\mu_a & 0 & 0 \\ \frac{-b_1 c_{yj} z^*}{\beta_2^2} & \frac{-b_2 c_{ya} z^*}{\beta_2^2} & \frac{-b_1 c_{yy} z^*}{\beta_2^2} & \frac{b_2\beta_2 - b_2 z^* c_{yz}}{\beta_2^2} \\ 0 & 0 & 1-\mu_y & 1-\mu_z \end{pmatrix}, \tag{5.11}$$

where

$$\beta_1 = 1 + c_{jj}j^* + c_{ja}a^* + c_{jy}y^* + c_{jz}z^*, \tag{5.12}$$

$$\beta_2 = 1 + c_{yj}j^* + c_{ya}a^* + c_{yy}y^* + c_{yz}z^*. \tag{5.13}$$

We now study the stability of the 'axis', i.e. horizontal or vertical fixed points. First we consider the horizontal fixed points. So we consider the Jacobian matrix evaluated at $(j^*, a^*, 0, 0)$:

$$\begin{pmatrix} \frac{-b_1 c_{jj} a^*}{\beta_1^2} & \frac{b_1\beta_1 - b_1 a^* c_{ja}}{\beta_1^2} & \frac{-b_1 c_{jy} a^*}{\beta_1^2} & \frac{-b_1 c_{jz} a^*}{\beta_1^2} \\ 1-\mu_j & 1-\mu_a & 0 & 0 \\ 0 & 0 & 0 & \frac{b_2}{\beta^2} \\ 0 & 0 & 1-\mu_y & 1-\mu_z \end{pmatrix}. \tag{5.14}$$

Because of the 2×2 block of zeros in the lower left corner, the eigenvalues of this 4×4 matrix are the eigenvalues of the 2×2 block in the upper left corner, and those of the 2×2 block in the lower right corner.

The eigenvalues of the upper left block determine whether or not the horizontal fixed point is stable in the absence of competition by the second species. The coefficients related to the second species or to the competition between the two species do not appear in the entries of this block.

The eigenvalues of the lower right block determine if a fixed point that is stable within the axis will remain stable in the presence of an invading small number of the second species.

The characteristic polynomial of the lower right block is

$$\lambda^2 - (1-\mu_z)\lambda - (1-\mu_y)\frac{b_2}{\beta_2}. \tag{5.15}$$

We first establish the conditions under which the roots λ_1 and λ_2 of (5.15) are inside the unit circle. By (5.2), it is necessary and sufficient that:

$$\frac{1}{2}[(1-\mu_z) + \sqrt{(1-\mu_z)^2 + 4(1-\mu_y)\frac{b_2}{\beta_2}}] < 1. \tag{5.16}$$

This is equivalent to:

$$\sqrt{(1-\mu_z)^2 + 4(1-\mu_y)\frac{b_2}{\beta_2}} < 1 + \mu_z, \tag{5.17}$$

and also to:

$$\frac{(1-\mu_y)b_2}{\mu_z\beta_2} < 1. \tag{5.18}$$

By substituting the value of β_2 in (5.13) into (5.18), and evaluating at $(j^*, a^*, 0, 0)$, we have :

$$\frac{b_2(1-\mu_y)}{\mu_z(1 + c_{yj}j^* + c_{ya}a^*)} < 1. \tag{5.19}$$

So if the eigenvalues of the upper left 2 by 2 block in (5.14) are inside the unit circle, then (5.19) is a necessary and sufficient condition for the stability of the horizontal fixed points.

A similar analysis shows that a vertical axis fixed point is stable if

$$\frac{b_1(1-\mu_j)}{\mu_a(1 + c_{jy}y^* + c_{jz}z^*)} < 1. \tag{5.20}$$

We now consider the coexistence fixed points, i.e. solutions of (5.3) which have no zero components. By substituting (5.4) in (5.3), we have a system of two linear equations for j^* and y^*:

$$\begin{aligned} j^*(c_{jj} + c_{ja}\tfrac{1-\mu_j}{\mu_a}) + y^*(c_{jy} + c_{jz}\tfrac{1-\mu_y}{\mu_z}) &= b_1\tfrac{1-\mu_j}{\mu_a} - 1, \\ j^*(c_{yj} + c_{ya}\tfrac{1-\mu_j}{\mu_a}) + y^*(c_{yy} + c_{yz}\tfrac{1-\mu_y}{\mu_z}) &= b_2\tfrac{1-\mu_y}{\mu_z} - 1. \end{aligned} \tag{5.21}$$

For convenience we denote the coefficients of this system as follows:

$$\alpha = \frac{1-\mu_j}{\mu_a}, \beta = \frac{1-\mu_y}{\mu_z},$$

$$\epsilon = c_{jj} + c_{ja}\alpha, \gamma = c_{jy} + c_{jz}\beta,$$

$$\delta = c_{yj} + c_{ya}\alpha, \eta = c_{yy} + c_{yz}\beta.$$

Then the system (5.21) can be rewritten as:

$$\begin{array}{l} j^*\epsilon + y^*\gamma = b_1\alpha - 1, \\ j^*\delta + y^*\eta = b_2\beta - 1. \end{array} \tag{5.22}$$

It has a unique solution if :

$$H \equiv \det \begin{pmatrix} \epsilon & \gamma \\ \delta & \eta \end{pmatrix} \neq 0, \tag{5.23}$$

which is then given by:

$$j^* = \frac{\gamma(b_2\beta - 1) - (b_1\alpha - 1)\eta}{\delta\gamma - \epsilon\eta}, \tag{5.24}$$

$$y^* = \frac{-\epsilon(b_2\beta - 1) + (b_1\alpha - 1)\delta}{\delta\gamma - \epsilon\eta}. \tag{5.25}$$

By substituting j^* and y^* in (5.4), we find a^* and z^*:

$$a^* = \alpha(\frac{\gamma(b_2\beta - 1) - (b_1\alpha - 1)\eta}{\delta\gamma - \epsilon\eta}), \tag{5.26}$$

$$z^* = \beta(\frac{-\epsilon(b_2\beta - 1) + (b_1\alpha - 1)\delta}{\delta\gamma - \epsilon\eta}). \tag{5.27}$$

The Jacobian matrix evaluated in (j^*, a^*, y^*, z^*) is:

$$\begin{pmatrix} \frac{b_1 c_{jj} a^*}{\beta_1^2} & \frac{-b_1\beta_1 - b_1 a^* c_{ja}}{\beta_1^2} & \frac{-b_1 c_{jy} a^*}{\beta_1^2} & \frac{-b_1 c_{jz} a^*}{\beta_1^2} \\ 1 - \mu_j & 1 - \mu_a & 0 & 0 \\ \frac{-b_1 c_{yj} z^*}{\beta_2^2} & \frac{-b_2 c_{ya} z^*}{\beta_2^2} & \frac{-b_1 c_{yy} z^*}{\beta_2^2} & \frac{-b_2\beta_2 - b_2 z^* c_{yz}}{\beta_2^2} \\ 0 & 0 & 1 - \mu_y & 1 - \mu_z \end{pmatrix}. \tag{5.28}$$

The resulting eigenvalue equation is quartic, but we can compute the eigenvalues numerically in CL_MATCONTM if the actual values of state variables and parameters are known. In this way we will be able to determine the stability of fixed points numerically. We will now study the overall dynamics of the model for the

Table 5.1: Parameter values for the Leslie–Gower model.

$b_1 = 20$	$c_{jj} = 0.36$	$b_2 = 18$	$c_{ja} = 0.55$	$c_{jz} = 0.23$	$\mu_j = 0.23$
$\mu_a = 0.72$	$c_{ya} = 0.08$	$c_{yy} = 0.18$	$c_{yz} = 0.26$	$\mu_y = 0.29$	$\mu_z = 0.98$

fixed model parameter values in Table 5.1. We perform a numerical continuation of the fixed points of M_{LG} under variation of the control parameters c_{yj} and c_{jy}. First we consider the horizontal fixed point

$$F_H = (21.50285631, 22.99611022, 0, 0),$$

which remains unchanged since c_{jy} and c_{yj} do not appear in (5.5) and (5.6). Since

$$b_1 \frac{1-\mu_j}{\mu_a} = 20 \times (1 - 0.23)/0.72 = 20.2778 > 1, \tag{5.29}$$

by (5.7), the horizontal fixed point is biologically meaningfull. Also, the free parameters do not enter in the entries of the upper left block matrix of (5.14). We first show that all the eigenvalues of this block are inside the unit circle. The 2 by 2 upper left block is:

$$\begin{pmatrix} \frac{-b_1 c_{jj} a^*}{\beta_1^2} & \frac{b_1\beta_1 - b_1 a^* c_{ja}}{\beta_1^2} \\ 1-\mu_j & 1-\mu_a \end{pmatrix}. \tag{5.30}$$

Evaluation of this matrix in F_H and substitution of the values of the model parameters gives:

$$\begin{pmatrix} -0.3619 & 0.3821 \\ 0.77 & 0.28 \end{pmatrix}. \tag{5.31}$$

The eigenvalues of this matrix are -0.6712 and 0.5893, with absolute values less than 1. That means that in a continuation of the horizontal fixed points with either c_{yj} or c_{jy} free, the eigenvalues of the upper left block of (5.14) are inside the unit circle. Now we consider the stability condition (5.19). For the given model parameters, the horizontal fixed point is stable if :

$$\frac{12.78}{21.0728 \times c_{yj} + 2.7829} < 1. \tag{5.32}$$

This is equivalent to

$$c_{yj} > c_{yj0},$$

where $c_{yj0} \approx 0.474477674$. Hence F_H is unstable if $0 \leq c_{yj} < c_{yj0}$ and stable if $c_{yj0} < c_{yj} \leq 1$. It is biologically plausible that the horizontal fixed point is stable only if the juveniles of the first species suppress the juveniles of the second species to a sufficient degree.

We will study the overall dynamics of the model for the parameter values specified in Table 5.1. The parameters c_{jy} and c_{yj} will vary.

5.1.3 Numerical continuation of the horizontal and vertical fixed points

First we consider the horizontal and the vertical fixed points and their stability. For all values of c_{jy} and c_{yj}, the fixed point obtained from (5.5) and (5.6)

$$F_H = (21.50285631, 22.99611022, 0, 0),$$

remains unchanged since c_{jy} and c_{yj} do not appear in (5.5) and (5.6). For the given model parameters, the horizontal fixed point is stable if $c_{yj} > c_{yj0}$, where $c_{yj0} = 0.474408$ and loses stability at a branch point

```
label = BP,
 x = ( 21.502856 22.996110 0.000000 0.000000 0.474408 )
```

It is biologically plausible that the horizontal fixed point is stable only if the juveniles of the first species suppress the juveniles of the second species to a sufficient degree.

Now we consider the vertical fixed point

$$F_V = (0, 0, 32.68698060, 23.68138391)^T.$$

F_V is stable if $c_{jy} > c_{jy0}$, where $c_{jy0} = 0.4571312026$. It loses stability via the branch point

```
label = BP,
x = ( 0.000000 0.000000 32.686981 23.681384 0.457129 )
```

5.1.4 Numerical continuation of the coexistence fixed points

Now we consider the coexistence fixed point (j^*, a^*, y^*, z^*), starting the continuation from

$$F_C = (16.42912, 17.570032, 28.871217, 20.916902),$$

where $c_{yj} = c_{jy} = 0$. This fixed point bifurcates into vertical and horizontal fixed points respectively, when one of c_{jy} and c_{yj} is varied and the other variable is fixed at 0. In the model this means that one species drives another to extinction. Continuation of the coexistence fixed points, where c_{jy} is the free parameter leads to BP and PD bifurcations at $c_{jy} = c_{yj0}$ and $c_{jy} = 0.170849$, respectively. The coexistence fixed points bifurcate into vertical fixed points at the BP. The coexistence fixed point is stable before the BP and unstable afterwards, this reconfirms the above analytical results. If we continue the coexistence fixed points with the free parameter c_{yj}, it bifurcates into the horizontal fixed point at another BP. The coexistence fixed point is stable before this BP and unstable afterwards.

The solutions to the equation $H = 0$, where H is given by (5.23), are the parameter values for which the existence and uniqueness of the coexistence fixed point are not guaranteed. In the present context, where only c_{yj} and c_{jy} vary, this leads to the condition

$$c_{yj}c_{jy} + ac_{yj} + bc_{jy} - c = 0, \tag{5.33}$$

where $a = 0.1666326531,\ b = 0.0855555552$, and $c = 0.3350275226$, which defines a hyperbola in $(c_{yj},\ c_{jy})$ space. It is not hard to prove that the point (c_{yj0}, c_{jy0}) lies on the hyperbola.

Now we return to the stability of the coexistence fixed points. The coexistence fixed point is unstable outside the rectangle

$$S_1 = \{(c_{yj}, c_{jy}) : 0 \le c_{yj} \le c_{yj0},\ \ 0 \le c_{jy} \le c_{jy0}\}\,.$$

Figure 5.1 shows the hyperbola $H = 0$ and the rectangle S_1.

Inside S_1 the stability properties of the coexistence fixed point are more complicated. By numerical continuation we find that there is an interior region in which the coexistence fixed points are unstable. This region is bounded by the

PD curve, where the stability changes. The projection of the PD curve on the (c_{yj}, c_{jy})-plane goes twice through the point (c_{yj0}, c_{jy0}). Indeed, the PD curve has two fold-flip points, where $(c_{yj}, c_{jy}) = (c_{yj0}, c_{jy0})$. Moreover, there are two gereralized period-doubling points GPD on the PD curve:
$(c_{yj}, c_{jy}) = (0.210138, 0.383143)$ and $(c_{yj}, c_{jy}) = (0.454279, 0.297779)$. These points along with the corresponding normal form coefficients are:

```
label = GPD ,
 x = ( 19.463734 20.815382 4.163299 3.016267
 0.297779 0.454279 )
Normal form coefficient of GPD = 5.271115e-006
label = LPPD,
 x = ( 20.354989 21.768530 1.744899 1.264162
 0.457129 0.474408 )
Normal form coefficient for LPPD :
[a/e , be]= -1.329134e-009, -5.060725e-005,

label = GPD ,
x = ( 4.771088 5.102414 28.857159 20.906717
 0.383143 0.210138 )
Normal form coefficient of GPD = 4.494110e-008
label = LPPD,
x = ( 1.297714 1.387833 30.714296 22.252194
 0.457129 0.474408 )
Normal form coefficient for LPPD :
[a/e , be]= -1.076502e-008, -2.619423e-006,
```

The branches of fold curves of the second iterate can be computed by switching at the GPD points. These curves emanate tangentially to the PD curve and form the stability boundary of the 2-cycles that are born when crossing the PD-curve.

More precisely, the region where there are stable fixed points of the second iterate is bounded by the two fold curves of the second iterate and the lower left part of the PD curve. From the applications point of view, this is the most interesting region because it shows that indeed the two species can coexist even when the competition is strong. We note that if both c_{yj} and c_{jy} are larger than 0.5 then the

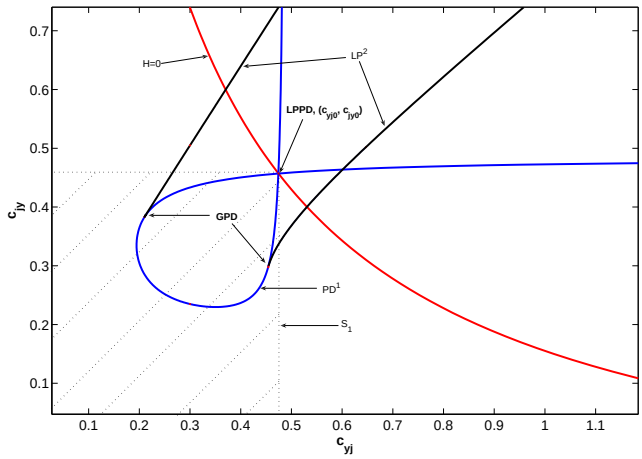


Figure 5.1: The flip curve PD^1, the fold curve of the second iterate LP^2, the hyperbola $H = 0$ and the rectangle S_1 in (c_{yj}, c_{jy})-plane.

horizontal fixed points, the vertical fixed points and the fixed points of the second iterate are all stable. The PD curve and the fold curves of the second iterate are given in Figure 5.1.

It can be shown analytically that there is a straight line of coexistence fixed points for the fixed parameter values $(c_{yj}, c_{jy}) = (c_{yj0}, c_{jy0})$ which bifurcates to the horizontal and vertical fixed points where $c_{jy} = c_{jy0}$ and $c_{yj} = c_{yj0}$ respectively. This straight line can be found numerically by switching to the fold curve in the fold-flip (LPPD) points of the flip curve since technically it is a curve of (generalized) fold points of the original Leslie-Gower map.

5.2 An age-structured cod stock model

We study the long-term dynamics of a two-dimensional stage-structured population model for the Barents Sea cod stock with nonlinear cannibalism terms introduced by A. Wikan and A. Eide (2004). The model is represented by a two-

dimensional system of difference equations for two stages of population. Following Wikan and Eide we consider three special cases of the original model with different ranges of cannibalism pressures on the new born, immature and the oldest part of immature. Using CL_MATCONTM, we discuss mathematical features of the model, that were not considered heretofore. This includes the continuation of curves of codimension 1 bifurcations of fixed points, and normal form analysis of codim 1 and codim 2 bifurcations. In this way we can compute the stability domains of the map and its iterates. We concentrate in particular on the third and fourth iterates of the map and their relation to the 1:3 and 1:4 resonant Neimark-Sacker points.

5.2.1 Introduction

In [104] Wikan and Eide discuss the highly oscillatory year to year behavior of fish population biomasses of commercial interest. This is well documented in [70] where data for several North Atlantic fish stocks are presented. Among them, the Barents Sea Cod stock is known as a heavily fluctuating stock biomass, see [37]. Four principal reasons may serve to explain these fluctuations, see [71]:

- Environmental changes: variation in temperature, salinity, current system etc.
- Ecosystem dynamics: multispecies dynamics, change in prey and predator biomasses.
- Changes in fishing pattern: open access dynamics, fisheries regulations.
- Cod stock dynamics: including recruitment and cannibalism dynamics.

There is no established understanding of which of the above factors is the most dominant as they probably all contribute to the observed fluctuations. The aim of

the study in [104] is to concentrate on the last factor. Intraspecific predation or cannibalism is a well known behavioral trait found in a variety of animal populations, see [33] and [85]. This biological phenomenon is also expected to play a crucial role in the population dynamics of cod stocks, see [67].

We will consider a discrete nonlinear stage structured model with seven parameters taken from [104]. In previous studies [104, 105] only a one-parameter bifurcation analysis is performed and the analysis of the supercritical nature of the found bifurcation points was possible only in very special situations. We extend this by numerical means to a two-parameter analysis with computation of all relevant normal form coefficients, which leads to several new results.

5.2.2 The model, its fixed points and their stability properties

The roots of the present model can be found in [47, 24, 26]. The model that we use is a two-dimensional difference equation (5.34) with seven parameters described in [104] as follows

$$M_{CS} : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} Fe^{-\beta_1 x_2} x_2 + (1-\mu_1) e^{-\beta_2 x_2} x_1 \\ Pe^{-\beta_3 x_2} x_1 + (1-\mu_2) x_2 \end{pmatrix}, \tag{5.34}$$

where $x_{1,t}$ and $x_{2,t}$ are the immature and mature parts of the population at time t respectively. F is the fecundity (that is the number of newborns per adult), and P, $0 < P \leq 1$, is the fraction of the immature population that survive and enter the mature stage one time later. μ_2 may be interpreted as natural death rate. μ_1 combines natural death and maturation, so $\mu_1 \geq P$. The corresponding parameters $\beta_i, i = 1, 2, 3$ will be referred to as cannibalism parameters. Thus, F is reduced by the factor $e^{-\beta_1 x_2}$ due to cannibalism practised by the mature part of the population. In a similar way the remaining part of the immature population $(1 - \mu_1)$ is reduced by the factor $e^{-\beta_2 x_2}$, and finally the survival from immature stage to the mature stage is reduced by the factor $e^{-\beta_3 x_2}$ due to cannibalism practised by individuals in the mature stage. In the model (5.34) we do not consider cannibalism within the stages.

The basic analytical results are given in [104]. We summarize them briefly. The

general form of the Jacobian of (5.34) is:

$$\begin{pmatrix} (1-\mu_1)e^{-\beta_2 x_{2,t}} & -F.\beta_1 e^{-\beta_1 x_{2,t}} x_{2,t} + Fe^{-\beta_1 x_{2,t}} - (1-\mu_1).\beta_2 e^{-\beta_1 x_{2,t}} x_{1,t} \\ P.e^{-\beta_3 x_{2,t}} & -P.\beta_3 e^{-\beta_3 x_{2,t}} x_{1,t} + (1-\mu_2) \end{pmatrix}. \tag{5.35}$$

Clearly, the vector $(x_1^*, x_2^*) = (0,0)$ is a trivial fixed point of (5.34). Evaluation of (5.35) at the trivial fixed point gives,

$$\begin{pmatrix} 1-\mu_1 & F \\ P & 1-\mu_2 \end{pmatrix}. \tag{5.36}$$

The characteristic polynomial of (5.36) is given by,

$$\lambda^2 + a\lambda + b = 0, \tag{5.37}$$

where $a = \mu_1 + \mu_2 - 2$ and $b = (1-\mu_1)(1-\mu_2) - FP$. The roots of (5.37) , eigenvalues of (5.36), are given by:

$$\frac{1}{2}\left(2 - (\mu_1+\mu_2) \pm \sqrt{(\mu_1+\mu_2)^2 + 4\mu_2\left(\frac{F.P + (1-\mu_1)\mu_2}{\mu_2} - 1\right)}\right). \tag{5.38}$$

We denote the inherent net productive number, R, by

$$R = \frac{F.P + (1-\mu_1)\mu_2}{\mu_2}, \tag{5.39}$$

It is clear that the trivial fixed point $(0,0)$ becomes unstable where $R > 1$, i.e., $F.P > \mu_1\mu_2$. In the rest of this study we assume $R > 1$. A nontrivial fixed point (x_1^*, x_2^*) of the model must satisfy:

$$\begin{aligned} x_1^* &= F.e^{-\beta_1 x_2^*} x_2^* + (1-\mu_1)e^{-\beta_2 x_2^*} x_1^*, \\ x_2^* &= P.e^{-\beta_3 x_2^*} x_1^* + (1-\mu_2)x_2^*. \end{aligned} \tag{5.40}$$

By the second equation of (5.40), we have

$$x_1^* = \frac{\mu_2}{P} e^{\beta_3 x_2^*} x_2^*. \tag{5.41}$$

Moreover, by substituting (5.41) into the first equation of (5.40) we find that x_2^* must satisfy the nonlinear equation $g(x_2^*) = 0$, where

$$g(x_2^*) = \frac{F.P}{\mu_2} e^{-(\beta_1+\beta_3)x_2^*} + (1-\mu_1)e^{-\beta_2 x_2^*} - 1, \tag{5.42}$$

Evidently x_2^* is uniquely determined from (5.42) since $g'(x_2^*) < 0$, $g(0) = R-1 > 0$ and $g(x_2^*) < 0$ when x_2^* is sufficiently large. The characteristic polynomial of (5.35) evaluated at (x_1^*, x_2^*) is given by

$$P(\lambda) = \lambda^2 + a_1\lambda + a_2, \tag{5.43}$$

where

$$\begin{aligned} a_1 &= (1+\beta_3 x_2^*)\mu_2 - 1 - (1-\mu_1)e^{-\beta_2 x_2^*}, \\ a_2 &= (1-\mu_1)e^{-\beta_2 x_2^*}(1-(\beta_1-\beta_2+\beta_3)\mu_2 x_2^*) - (1-\beta_1 x_2^*)\mu_2. \end{aligned} \tag{5.44}$$

The nontrivial fixed point (x_1^*, x_2^*) is stable if the roots of (5.43) are both in $]-1, 1[$. We use the stability conditions in the form of the Jury criteria, see [71], §A2.1, i.e., $P(1) > 0, P(-1) > 0$ and $a_2 < 1$.
Stability condition $P(1) > 0$ holds when

$$1 + a_1 + a_2 > 0,$$

i.e.,

$$\left(\beta_1 + \beta_3 - (1-\mu_1)(\beta_1-\beta_2+\beta_3)e^{-\beta_2 x_2^*}\right)\mu_2 x_2^* > 0. \tag{5.45}$$

It is clear that $P(1) = 0$ is a criterion to detect a fold bifurcation (LP), where a multiplier +1 crosses the unit circle. Moreover, it should be noticed that the left hand side of (5.45) is always positive for any nontrivial fixed point (x_1^*, x_2^*), so there can be no transition from stability to instability through a fold bifurcation.
The second stability condition $P(-1) > 0$ gives

$$1 - a_1 + a_2 > 0,$$

i.e.,

$$(1-\mu_1)e^{-\beta_2 x_2^*}(2-(\beta_1-\beta_2+\beta_3)\mu_2 x_2^*) + 2(1-\mu_2) + (\beta_1-\beta_3)\mu_2 x_2^* > 0. \tag{5.46}$$

Evidently $P(-1) = 0$ is a criterion to detect a flip bifurcation(PD).

Finally, if $\lambda_{1,2} = e^{\pm i\theta}$, then the stability condition $\lambda_1\lambda_2 < 1$ or equivalently $a_2 < 1$ leads to

$$1 + (1 - \beta_1 x_2^*)\mu_2 - (1 - \mu_1)e^{-\beta_2 x_2^*}(1 - (\beta_1 - \beta_2 + \beta_3)\mu_2 x_2^*) > 0. \quad (5.47)$$

It is clear that $a_2 = 1$ is a criterion to detect a Neimark-Sacker bifurcation, where a conjugate pair of complex multipliers crosses the unit circle.

5.2.3 Numerical bifurcation analysis of the model

Following [104] we consider three special parameter ranges of (5.34). In each case, we first discuss analytically the stability of the reduced model by using the stability conditions (5.46) and (5.47) derived in the previous section. Then, we use CL_MATCONTM for bifurcation analysis of the map. We note that all normal form coefficients in our computations are small in absolute value; this is caused by the exponentials in the definition of the map and does not indicate that the sign of the coefficients cannot be trusted.

Case 1

We consider the case where the cannibalism pressures on the newborns, immature population and those on the threshold of entering the mature stage are equal, i.e., $\beta_1 = \beta_2 = \beta_3 = \beta$. Thus, the model (5.34) is rewritten as

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} F.e^{-\beta x_2}x_2 + (1 - \mu_1)e^{-\beta x_2}x_1 \\ P.e^{-\beta x_2}x_1 + (1 - \mu_2)x_2 \end{pmatrix}. \quad (5.48)$$

The nontrivial solution of this model can be expressed by

$$(x_1^*, x_2^*) = (\frac{\mu_2}{\beta.P}K.\ln K, \frac{1}{\beta}\ln K), \quad (5.49)$$

where

$$K = \frac{1}{2}(1 - \mu_1) + \sqrt{\frac{F.P}{\mu_2} + \frac{(1 - \mu_1)^2}{4}} = \frac{1 - \mu_1}{2} + \frac{\sqrt{(1 - \mu_1)^2 + 4(R - 1)}}{2}. \quad (5.50)$$

We note that

$$K \geq \frac{1}{2}(1-\mu_1) + \sqrt{\mu_1 + \frac{(1-\mu_1)^2}{4}} \geq \frac{1}{2}(1-\mu_1) + \frac{1}{2}(1+\mu_1) = 1. \quad (5.51)$$

The characteristic polynomial (5.43) can be reduced to

$$P_1(\lambda) = \lambda^2 + b_1\lambda + b_2 = 0, \quad (5.52)$$

where

$$\begin{aligned} b_1 &= \mu_2(1+\beta x_2^*) - 1 - \frac{1-\mu_1}{K}, \\ b_2 &= \frac{1-\mu_1}{K}(1-\beta\mu_2 x_2^*) - (1-\beta x_2^*)\mu_2. \end{aligned} \quad (5.53)$$

Accordingly, the stability conditions (5.46) and (5.47) become

$$\frac{1-\mu_1}{K}(2-\mu_2 \ln K) + 2(1-\mu_2) > 0, \quad (5.54)$$

$$(\mu_2 \ln K - 1)(\frac{1-\mu_1}{K} - 1) + \mu_2 > 0. \quad (5.55)$$

In the stability conditions (5.54) and (5.55), the interaction parameter β dropped out. Denoting the left hand side of (5.54) and (5.55) by B and C respectively, it is clear that B is positive if $K \leq e^{\frac{2}{\mu_2}}$ and C is positive if $K \leq e^{\frac{1}{\mu_2}}$. Thus, on the common domain $K \leq e^{\frac{1}{\mu_2}}$ both (5.54) and (5.55) hold, hence in this part of parameter space (x_1^*, x_2^*) is a stable fixed point. In [104] it is shown by qualitative arguments that loss of stability is possible through either a NS or PD bifurcation but that the latter is possible only in a small parameter range. We will make this more precise by numerical computations.

We do a numerical bifurcation analysis of (5.48) by starting from the model parameters $\beta = 1, P = 0.5, \mu_1 = 0.5, F = 120$ and $\mu_2 = 0.9$. For the above parameter set the nontrivial fixed point $(x_1^*, x_2^*) = (32.2814, 2.1305)$ is computed from (5.49) and is stable. We do a numerical continuation of fixed points back and forth where F is the free parameter, we refer to this as *Run 1*. We obtain the following CL_MATCONTM output:

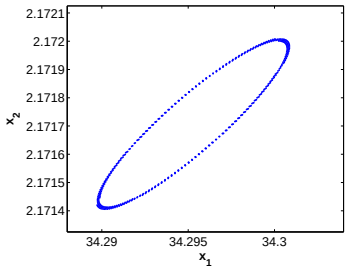


Figure 5.2: The invariant curve for $\beta_1 = \beta_2 = \beta_3 = 1, \mu_1 = P = 0.5, \mu_2 = 0.9$,$F = 130.65$.

```
label = BP,x = ( 0.00000 0.00000 0.90000 )
label = NS,x = ( 34.287724 2.171557 130.609334 )
normal form coefficient of NS = -5.721873e-004
```

Two bifurcation points are detected along the fixed point curve, a branch point (BP) and a supercritical Neimark-Sacker point (supercriticality follows from the fact that the normal form coefficient of the NS point is negative). The nontrivial fixed point is stable only for $0.9 < F < 130.609334$. The dynamics beyond the upper threshold is a stable invariant curve which surrounds the unstable fixed point. Such a curve is shown in Figure 5.2.
The new branch of fixed points that was encountered in *Run 1* for $F = 0.9$ is computed in *Run 2* and gives the following CL_MATCONTM output:

```
label = BP, x = ( -0.00000 -0.00000 0.90000 )
label = PD, x = ( 0.000000 0.000000 3.300000 )
normal form coefficient of PD = 8.753732e-002
```

Clearly, the new branch is the trivial branch of fixed points. The trivial fixed point is stable before the BP point and unstable afterwards where the reproductive number R in (5.39) becomes larger than 1.
Now we compute the Neimark-Sacker bifurcation curve forth and back, by starting from the NS point of *Run 1*, with two free parameters F and μ_2. We call this *Run 3*:

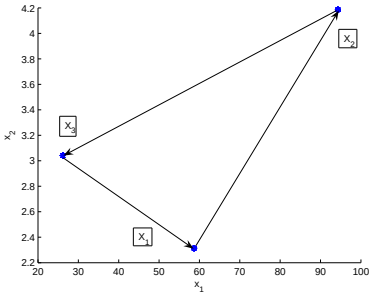


Figure 5.3: An exact 3-cycle close to the R_3 point, where $F = 399.5861$ and $\mu_2 = 0.444715$.

```
label = R3,
x = ( 61.127825 3.001853 399.586101 0.505977 -0.500000 )
Normal form coefficient of R3 :
 Re(c_1) = -1.080111e-000
label = R2,
x = ( 32.714248 2.069443 117.303643 0.997942 -1.000000 )
Normal form coefficient of R2 :
[c , d] = -1.518117e-004, -3.075159e-003
```

In *Run 3*, we find a resonance 1:3 point. Since its normal form coefficient is negative, the bifurcation picture near the R_3 point is qualitatively the same as presented in [62], Fig. 9.12. In particular, there is a region near the R_3 point where a stable invariant closed curve coexists with an unstable equilibrium. For parameter values close to the R_3 point, the map has a saddle cycle of period three. An exact 3-cycle near the R_3 point is $C_3 = \{X_1, X_2, X_3\}$ where

$$X_1 = (58.66425, 2.31385), X_2 = (94.32305, 4.18521), X_3 = (26.16934, 3.04173).$$

This cycle and the parameter values are given in Figure 5.3. The multipliers of the fixed point of the third iterate in X_1 are $\lambda_1 = 1.03980469$ and $\lambda_2 = 0.356852$, thus confirming the saddle character.

If we continue the fixed point of the third iterate of the map starting from X_3 for decreasing values of μ_2 then it gains stability at a fold point for $\mu_2 = 0.444666$. This stable 3-cycle loses its stability again at a PD point for $\mu_2 = 0.499060$ after which a series of successive period doubling bifurcations occur such that new orbits of period $3.2^k, k = 1, 2, ...,$ are created. A 6-cycle is given by $C_6 = \{X_1, X_2, X_3, X_4, X_5, X_6\}$ where

$$X_1 = (49.79841, 1.68883), X_2 = (129.26567, 5.43071)$$

$$X_3 = (9.78778, 2.95507), \quad X_4 = (61.74516, 1.70878)$$

$$X_5 = (1.29237, 6.43133), \quad X_6 = (4.24233, 3.26836).$$

This cycle is depicted in Figure 5.4. A 12-cycle with parameter values is given in Figure 5.5.

We note that for $\mu_2 \in [0.444666, 0.499060]$ we have bistability of a fixed point of the map and a fixed point of the third iterate.

We now consider the map near the detected R_2 point computed in *Run 3*. Since the normal form coefficients c and d are both negative, we are precisely in the situation of [62], Fig. 9.10 (case $s = -1$). For a region of parameter values close to the R_2 point the map has an unstable 2-cycle that coexists with a stable closed invariant curve. Crossing a bifurcation curve, the 2-cycle simultaneously undergoes a NS bifurcation. By branch switching in the R_2 point, we compute the NS branch of the second iterate, which corresponds to $H^{(2)}$ in [62], Fig. 9.10. Further, from the R_2 point a flip curve originates. Computing the flip curve, reveals that a flip bifurcation exists in a small vicinity of the parameter $\mu_2 = 0.997942$. This is consistent with the analysis in [104] of the reduced model in Case 1. A figure of the Neimark-Sacker curve in *Run 3*, the flip curve through the R_2 point and the branch of NS points of the second iterate is given in Figure 5.6. A magnified picture of these curves is given in Figure 5.7. This Figure can be compared (qualitatively, of course) with [62], Fig. 9.10.
We now continue the fixed point $(x_1^*, x_2^*) = (15.360; 13.183289)$ along the straight line $F = 114$ with $P = \mu_1 = 0.5, \mu_2 = 0.1$ and $\beta = 1$ and varying μ_2. We note that the fixed point is initially stable, and call this *Run 4*:

```
label = PD, x = ( 32.053569 2.055439 0.998335 )
normal form coefficient of PD = 2.952412e-003
```

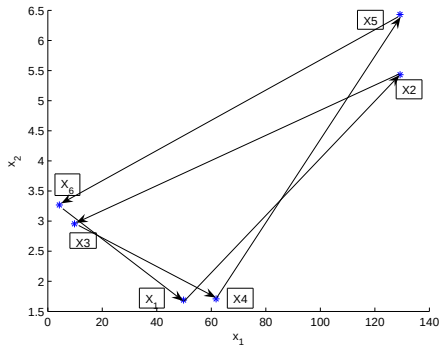


Figure 5.4: An exact 6-cycle for $F = 399.5861, P = \mu_1 = 0.5, \mu_2 = 0.508$ and $\beta = 1$.

The flip points in Figure 5.6 below the R_2 point have a positive normal form coefficient. Hence a supercritical stable 2-cycle is born when crossing the flip curve, which coexists with an unstable fixed point of the map. A point on this stable 2-cycle for $F = 114$ and $\mu_2 = 0.999831$ is given by

$$C_2 = \{X_1, X_2\} = \{(40.608284, 2.598127); (23.550936, 1.511325)\} .$$

The 2-cycle exists for $0.998335 < \mu_2 < 1$.

Case 2

Now we turn to the case where the cannibalism pressure on the newborn is dominating and decreases as age increases, i.e., $\beta_1 > \beta_2 > \beta_3$. Clearly, the left hand sides of (5.46) and (5.47) are positive for small values of x_2^*, i.e., in this part of parameter space the fixed point (x_1^*, x_2^*) is stable. A qualitative reasoning in [104] leads to the conclusion that both a NS and a PD bifurcation are possible, but the latter only in a small parameter region.
For the numerical stability analysis of the fixed point we consider the parameter set $\mu_1 = \mu_2 = P = 0.5, F = 55$ and $\beta_i = 4 - i, i = 1, 2, 3$. For these parameters the fixed point $(x_1^*, x_2^*) = (2.8213, 1.01868)$ is numerically computed from (5.41) and (5.42). We note that it is an unstable fixed point. Now, in *Run 5*, we continue fixed points where F is the free parameter.

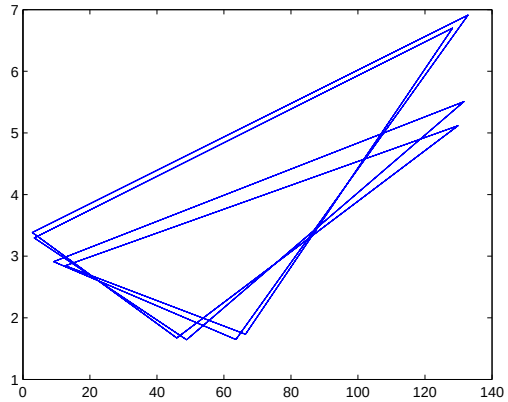


Figure 5.5: An exact 12-cycle for $F = 399.5861, P = \mu_1 = 0.5, \mu_2 = 0.52$ and $\beta = 1$.

```
label = NS, x = ( 2.718282 1.00000 50.903622 )
normal form coefficient of NS= -3.717346e-002
label = BP, x = ( -0.00000 -0.00000 0.50000 )
```

Run 5 shows that the fixed point is stable for small values of the fecundity, i.e., between BP and NS. When F exceeds the threshold $F_c = 50.903622$, i.e., when the inequality sign in (5.47) is reversed, we find a stable invariant curve.
Now we continue with free parameter β_1 . We refer to this as *Run 6*:

```
label = NS, x = ( 3.132638 1.072181 2.793847 )
normal form coefficient of NS = -2.864118e-002
label = PD, x = ( 60.897776 3.007941 0.332657 )
normal form coefficient of PD = 1.075561e+001
```

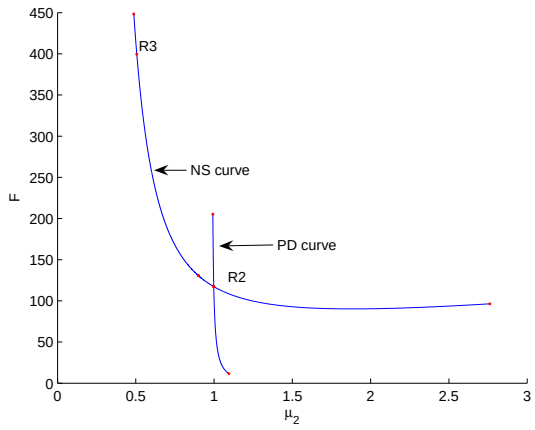


Figure 5.6: Neimark-Sacker bifurcation curve of *Run 3* and the flip curve through the R_2 point.

The fixed point is stable between the PD and NS points, i.e., where $\beta_1 \in [2,\ 2.7987]$, $\beta_2 = 2, \beta_3 = 1$. We proceed with the numerical investigation of stability where β_2 is free, in *Run 7*:

```
label = NS, x = ( 2.932052 1.038207 1.258253 )
normal form coefficient of NS = -3.139363e-002
```

We find that the fixed point is unstable before NS and stable afterwards, i.e., where $\beta_1 = 3, \beta_2 \in [1,\ 1.258201]\,, \beta_3 = 1$. Next, we continue with β_3 free, in *Run 8*:

```
label = NS, x = ( 2.926588 1.001660 1.070402 )
normal form coefficient of NS = -3.266591e-002
label = PD, x = ( 8.864167 0.362921 8.805179 )
normal form coefficient of PD =1.186472e+000
```

By monitoring the multipliers in *Run 8* it is found that the fixed point is stable between the NS and PD points, i.e., the fixed point is stable where $\beta_1 = 3, \beta_2 = 2, \beta_3 \in [1.070402,\ 2]$. Since the normal form coefficient of the PD point is positive, a stable 2-cycle is born for $\beta_3 > 8.805179$. Moreover, it can be seen that increasing β_3, the cannibalism of the immature on the threshold of entering the

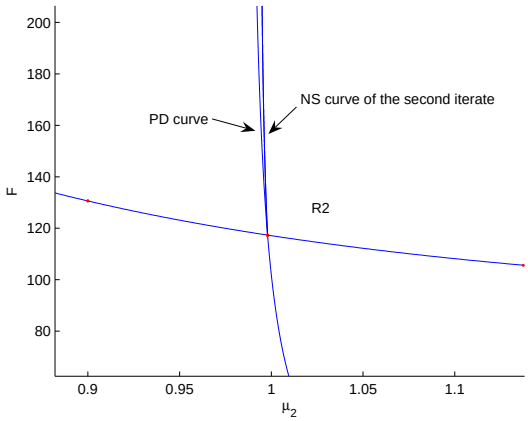


Figure 5.7: Close up of the flip curve and the NS curve of the second iterate rooted in the R_2 point.

mature age, results in a wider range of stability than increasing β_1.
For a further analysis we ignore the condition $\beta_1 > \beta_2 > \beta_3$ and compute the Neimark-Sacker curve, by starting at the NS point in *Run 6*, with free parameters F and β_3, this is *Run 9*:

```
label = R4, x =( 3.119219 1.003084 58.799673 1.131015 0 )
normal form coefficient of R4 : A= -4.610753e+00 +
-1.142472e+00 i
```

Since $|A| > 1$ in the R_4 point in *Run 9*, two cycles of period 4 of the map are born. The fixed points $F_k, k = 1, 2, 3, 4$ of the fourth iterate of the map which are closer to the original fixed point are saddles, while the remote ones $S_k, k = 1, 2, 3, 4$ are attractors. An exact stable 4-cycle for $\beta_3 = 1.131015$ and $F = 58.9$ is given by $C_4 = \{X_1, X_2, X_3, X_4\}$ where
$X_1 = (3.21494, 1.035797); X_2 = (2.93066, 1.01606)$,
$X_3 = (3.031476, 0.97239); X_4 = (3.31453, 0.99085)$
We present this cycle in Figure 5.8. The multipliers of the fourth iterate of the map in X_1 are $\lambda_1 = 0.999819$ and $\lambda_2 = 0.996348$, confirming the stability of the 4-cycle.

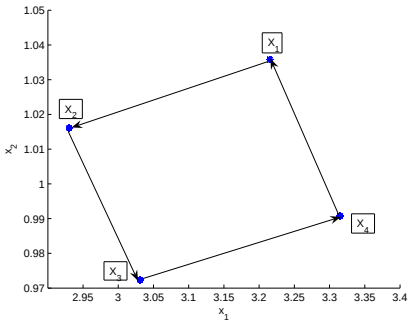


Figure 5.8: An exact 4-cycle for $\mu_1 = \mu_2 = P = 0.5, \beta_1 = 3, \beta_2 = 2, \beta_3 = 1.131015, F = 58.9$.

To compute the stability domain of the 4-cycle we note that since $|A| > 1$, there are two half-lines of fold bifurcation curves of the fourth iterate that emanate from the R_4 point. We present these lines in Figure 5.9.

For each set of parameter values in the wedge between the two half-lines both a stable 4-cycle and an unstable 4-cycle exist. For fixed values of F larger than that of the R_4 point the fixed points of the fourth iterate form a closed curve that changes stability in two fold curves. We note that the stable 4-cycles exist in a wide parameter region but there is no bistability with fixed points of the original map.

The Neimark-Sacker curve, starting from the NS point in *Run 8*, where μ_2 and β_3 are free parameters is computed in *Run 10*:

```
label = R4  , x = ( 3.011107 0.988367 0.511112 1.104883
-0.000000 )
Normal form coefficient of R4 : A = -4.675831e+000 +
-1.079711e+000 i
label = R3  , x = ( 5.026676 0.735955 0.910954 1.795572
```

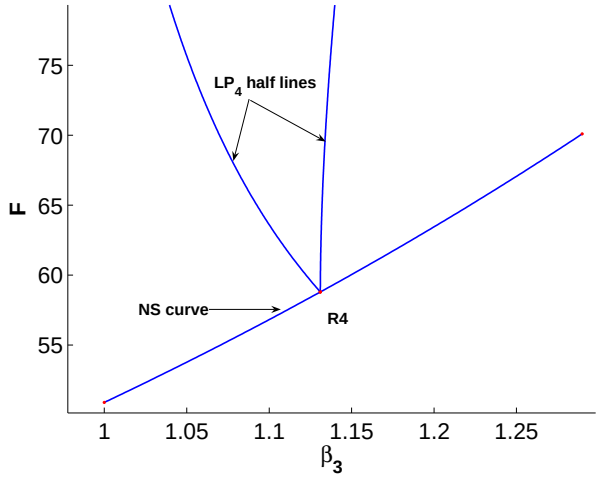


Figure 5.9: Two half-lines of fold bifurcation points emanate from an R_4 point.

```
-0.500000 )
Normal form coefficient of R3 : Re(c_1) = -1.581503e+000
label = R2  , x = ( 6.126237 0.627511 1.395201 1.995804
-1.000000 )
Normal form coefficient of R2 : [c , d] = 6.737115e-002,
-1.789198e-001
```

Case 3

In the last case we assume $\beta_1 < \beta_2 < \beta_3$. It is clear that the left hand sides of (5.46) and (5.47) are positive if x_2^* is small enough, i.e., there exists a parameter interval where (x_1^*, x_2^*) is stable. If $\beta_1 \approx \beta_2 \approx \beta_3$ (but $\beta_1 < \beta_2 < \beta_3$), then the left hand side of (5.47) is approximately equal to $1+(1-\beta_1 x_2^*)\mu_2-(1-\mu_1)e^{-\beta_2 x_2^*}$ and can be negative for some parameter values. That means that there exists a parameter region where a Neimark-Sacker bifurcation may occur. We note that in Case 3 when β_3 becomes large compared to β_1, then $(\beta_1-\beta_3)\mu_2 x_2^*$ in B (left hand side of (5.54)) becomes the dominating negative term which strongly suggests that

in this case there will be a flip bifurcation at the instability threshold. We remark that in the next runs, we are interested only in the range $\beta_1 < \beta_2 < \beta_3$.

We consider the parameter set $\mu_1 = \mu_2 = P = 0.5, F = 200$ and $\beta_i = i, i = 1, 2, 3$. The fixed point

$$(x_1^*, x_2^*) = (72.8206, 1.33341). \tag{5.56}$$

is computed from (5.41) and (5.42). We note that this is an unstable fixed point. Now we continue the fixed point (5.56) where F is the free parameter, we call this *Run 11*:

```
label = PD, x = ( 25.184229 1.056945 64.424674 )
normal form coefficient of PD = 1.910721e-001
```

In *Run 11* there is a supercritical flip bifurcation at the instability threshold, hence a stable 2-cycle is born for $F > 64.424674$. The fixed point is unstable before the PD point and stable afterwards. The new branch of fixed points of the second iterate is given in Figure 5.10.

We proceed with the continuation of fixed points where β_1 is free, we call this *Run 12*:

```
label = PD, x = ( 49.555931 1.231597 1.337322 )
normal form coefficient of PD = 4.476680e-003
label = NS, x = ( 6.567122 0.731554 4.410725 )
normal form coefficient of NS = -7.669904e-003
```

The fixed point is stable between the PD and NS points, i.e., when $\beta_1 \in [1.337322,\ 2]\,, \beta_2 = 2, \beta_3 = 3$. Due to the positive sign of the normal form coefficient of the PD point, a stable 2-cycle coexists with the unstable fixed point of the map for $\beta_1 < 1.337322$.

The fixed point (x_1^*, x_2^*) in (5.56), remains unstable under variation of the parameter β_2, hence increasing the cannibalism pressure on the immature part is not stabilizing factor from a dynamical point of view. We now continue with the free parameter β_3, we call this *Run 13*:

```
label = PD, x = ( 64.840130 1.640192 2.241879 )
normal form coefficient of PD = 2.750335e-003
label = NS, x = ( 29.976635 2.996639 0.768503 )
normal form coefficient of NS = -4.177821e-004
```

The fixed point is stable between the PD and NS points, i.e., when $\beta_1 = 1, \beta_2 = 2, \beta_3 \in [2,\ 2.241879]$. From the sign of the normal form coefficient of the PD point, we see that a stable 2-cycle is born when β_3 exceeds the threshold stability $\beta_3 = 2.241879$. An exact stable 2-cycle for $P = \mu_1 = \mu_2 = 0.5, \beta_1 = 1$, $\beta_2 = 2, \beta_3 = 2.2510715$ and $F = 200$ is given by

$$C_2 = \{X_1, X_2\} = \{(66.459403, 1.69537); (63.349999, 1.578968)\}\ .$$

We proceed with computing the Neimark-Sacker curve encountered in *Run 13*, where F and β_3 are free in the continuation, we call this *Run 14*:

```
label = R3  , x = ( 60.356868 2.997515 402.928800
1.001660 -0.500000 )
Normal form coefficient of R3 :
Re(c_1) = -1.095285e+000
label = R4  , x = ( 8.161668 2.995002 54.393960
 0.334726 0.000000 )
Normal form coefficient of R4 :
 A = -5.155721e+000 + -1.411666e+000 i
```

The R_3 point has the same characteristics (i.e. normal form coefficients with the same sign) as that in *Run 3*. The R_4 point has the same characteristics (absolute value and sign of real and imaginary part) as that in *Run 9*. By *Run 14* there are unstable 3-cycles and stable 4-cycles of fixed points near the R_3 and R_4 points, respectively. We continue by computing the Neimark-Sacker curve forth and back where μ_2 and β_3 are the free parameters, we call this *Run 15*:

```
label = R3  , x = ( 36.106260 2.711598 0.582753
0.898279 -0.500000 )
Normal form coefficient of R3 :
Re(c_1) = -1.141700e+000
label = R2  , x = ( 49.265517 2.192296 0.835266
1.185576 -1.000000 )
Normal form coefficient of R2 :
 [c , d] = -5.112120e-004, -1.100939e-003
label = R4  , x = ( 16.751436 3.820439 0.354409
0.476980 -0.000000 )
Normal form coefficient of R4 :
A = -3.921588e+000 + -2.056128e+000 i
```

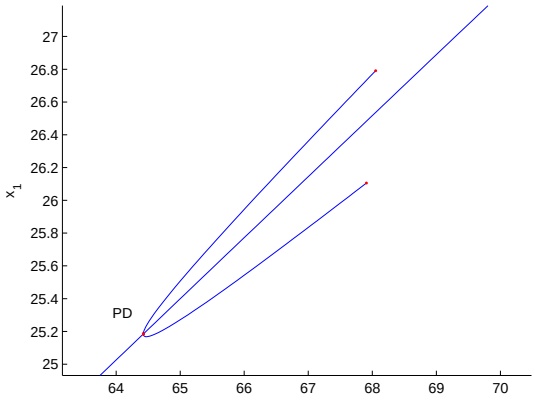


Figure 5.10: Branch of fixed points of the second iterate and of the original map in (F, x_1) space.

The R_3 and R_2 points have the same characteristics (i.e. normal form coefficients with the same sign) as those in *Run 3*. The R_4 point has the same characteristics (absolute value and sign of real and imaginary part) as that in *Run 9*.

Chapter 6

Applications in economics

Many mathematical models of economics try to obtain their goals by adaptive processes, based on trial and error or learning by doing methods. This implies that the mathematical modeling of these processes, where decisions are repeatedly taken over time, are formulated as determininistic discrete maps.

The Cournot-based models are the most frequently discussed models in the economics literature. The pioneering work of Cournot (1838) has initiated a large sequence of studies on static and dynamic models. A comprehensive summary of results on single-product models and a literature review can be found in [77], whilst their multi-product extensions are discussed in [78].

In this chapter, we first consider a duopoly and its extension to an oligopoly model and study their dynamical behaviour under variation of one and two control parameters. We proceed with a general case of a Cournout duopoly model of Kopel and compute stability domains of the first, second, third and fourth iterates of the model.

A part of this chapter was accepted for publication in [39].

6.1 A Cournot duopoly

Economics recognizes different market structures:

- Pure Competition: many competing firms producing the same product
- Monopoly: market controlled by a single seller
- Oligopoly: market dominated by a few large suppliers
- Duopoly: an oligopoly with exactly two firms

In the case of monopoly one single firm dominates the whole market. Its supply influences the market price appreciably, and it can take advantage of this to increase its profits by deliberately limiting the supply. An oligopoly is a market form in which a market or industry is dominated by a small number of sellers (oligopolists). Oligopolistic markets are characterized by interactivity. The decisions of one firm influence the decisions of other firms. There are two principal duopoly models, Cournot duopoly and Bertrand duopoly. In the Cournot model, two firms know each others output and treat this as a fixed amount, and produce in their own firm according to this.

The Cournot duopoly model, see [88], Ch. 5 and see [89], Ch. 7, shows two firms that react to one another's output changes until they eventually reach a position from which neither would wish to depart. Both firms eventually expand to such a degree that they have constant shares in the market and secure only normal profits.

The market demand is assumed to be isoelastic such that the price p is reciprocal to the total demand Q, i.e; $p = \frac{1}{Q}$.

There are two firms, denoted by X and Y, that produce the amounts of goods x and y with constant marginal costs a and b, respectively. Goods are perfect substitutes so that, provided demand equals supply, the total demand equals the total supplies, $Q = x + y$. Their expected profits become accordingly,

$$U(x, y) = \frac{x}{x + y} - ax, \tag{6.1}$$

$$V(x, y) = \frac{y}{x + y} - by. \tag{6.2}$$

The usual procedure to maximize profit leads to unimodal reaction functions which construct the following dynamic process under the naive expectation formation

$$\begin{cases} x_{t+1} = f(y_t), \\ y_{t+1} = g(x_t), \end{cases} \tag{6.3}$$

where $f(y)$ is the reaction function of firm X and $g(x)$ is the reaction function of firm Y. Both are specified as

$$f(y_t) = \sqrt{\frac{y_t}{a}} - y_t, \tag{6.4}$$

$$g(x_t) = \sqrt{\frac{x_t}{b}} - x_t. \tag{6.5}$$

$f(y)$ has its maximum value $\frac{1}{4a}$ at $y = \frac{1}{4a}$, and its domain should be restricted to the interval $[0, \ \frac{1}{a}]$ for nonnegative values of output, and so does $g(x)$ with replacing a with b. When the dynamic process is designed to map the maximum point to an interior point of the interval, it always generates positive productions to both firms. Solving $g(\frac{1}{4a}) \leq \frac{1}{a}$ and $f(\frac{1}{4b}) \leq \frac{1}{b}$ gives the upper and lower bounds of the marginal cost b in terms of the marginal cost a, $\frac{4}{25}a \leq b \leq \frac{25}{4}a$.

If these inequalities are violated, the dynamic system (6.3) induces negative output values.

6.1.1 Stability analysis of the fixed points of a duopoly model

The fixed point of (6.3), which we call the *Cournot point*, is the intersection of the reaction functions (6.4) and (6.5), see Figure 6.1, i.e.

$$\begin{aligned} \sqrt{\tfrac{y}{a}} - y &= x, \\ \sqrt{\tfrac{x}{b}} - x &= y, \end{aligned} \tag{6.6}$$

or

$$\begin{aligned} \tfrac{y}{a} &= (x+y)^2, \\ \tfrac{x}{b} &= (x+y)^2. \end{aligned} \tag{6.7}$$

That gives

$$y = \frac{a}{b}x. \tag{6.8}$$

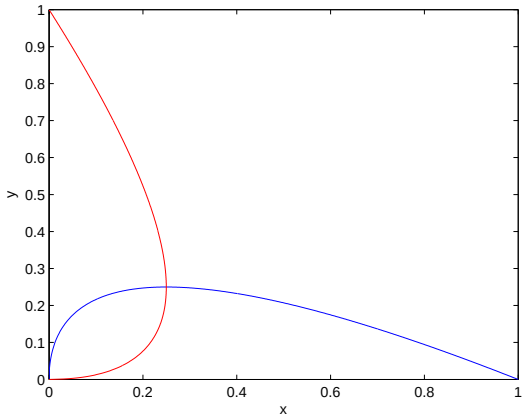


Figure 6.1: Intersection of the reaction curves defined by (6.4) and (6.5) defines the Cournot point of duopoly model.

By substituting (6.8) into the first and second equation of (6.6), respectively, we obtain the quadratic equations:

$$y^2(1+\tfrac{b}{a})^2 - \tfrac{y}{a} = 0, x^2(1+\tfrac{a}{b})^2 - \tfrac{x}{b} = 0. \tag{6.9}$$

Thus the Cournot point is given by:

$$\begin{array}{l} x = \frac{b}{(a+b)^2}, \\ y = \frac{a}{(a+b)^2}. \end{array} \tag{6.10}$$

Substituting these Cournot coordinates in the profit functions (6.1) and (6.2) gives the profits earned at the Cournot point,

$$\begin{array}{l} \prod_x = (\frac{b}{a+b})^2, \\ \prod_y = (\frac{a}{a+b})^2. \end{array} \tag{6.11}$$

Taking ratios of Cournot outputs (6.10) and Cournot profits (6.11) gives

$$\begin{array}{l} \frac{x}{y} = \frac{b}{a}, \\ \frac{\prod_x}{\prod_y} = (\frac{b}{a})^2. \end{array} \tag{6.12}$$

The ratios (6.12) reveal that at the Cournot point, a firm with lower marginal cost produces more output and makes more profit than a firm with higher marginal cost.

We call a firm with lower marginal cost an efficient firm and one with higher marginal cost an inefficient firm. An efficient firm dominates the market and is more profitable than an inefficient firm.

To examine the stability of the Cournot point, we make a linear approximation of the dynamic process (6.3) at the Cournot point and construct the Jacobian matrix,

$$J = \begin{pmatrix} 0 & \frac{b-a}{2a} \\ \frac{a-b}{2b} & 0 \end{pmatrix}. \tag{6.13}$$

The characteristic polynomial of (6.13) is given by:

$$\lambda^2 + \frac{(b-a)^2}{4ab} = 0. \tag{6.14}$$

The eigenvalues, roots of (6.14), are given by:

$$\lambda_{1,2} = \pm i(\frac{b-a}{2\sqrt{ab}}), \;\; a, b > 0, \tag{6.15}$$

The Cournot point is stable if

$$|\lambda_j| < 1, \;\; j = 1, 2. \tag{6.16}$$

This is equivalent to:

$$\frac{|b-a|}{a} < 2\sqrt{\frac{b}{a}}, \tag{6.17}$$

or

$$(\frac{b}{a})^2 - 6(\frac{b}{a}) + 1 < 0. \tag{6.18}$$

This holds if $(3-2\sqrt{2}) < \frac{b}{a} < (3+2\sqrt{2})$. We note that loss of stability occurs via a Neimark-Sacker bifurcation when $\frac{a}{b}$ crosses the thresholds $(3 \pm 2\sqrt{2})$, where a conjugate pair of complex eigenvalues (6.15) crosses the unit circle at $e^{\pm i\theta}$, $\theta = \frac{\pi}{4}$.

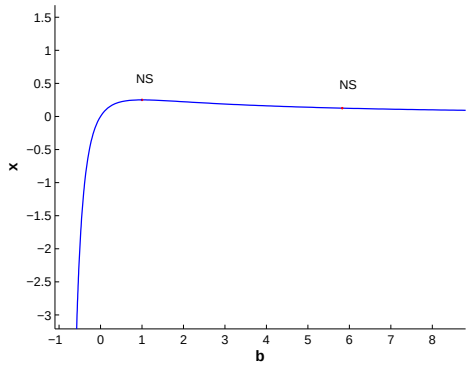


Figure 6.2: A curve of fixed points starting from the Cournot point $(\frac{1}{4}, \frac{1}{4})$.

Now we do a numerical continuation of Cournot points starting from the fixed point $(x, y) = (\frac{1}{4}, \frac{1}{4})$, obtained from (6.10), $(a, b) = (1, 1)$. The control parameter is b while a is fixed in the continuation. The fixed point loses stability via the NS points $(x, y) = (0.125000, \ 0.728553)$ where $b = 0.171573$, and $(x, y) = (0.125000, \ 0.0214475.828427)$ where $b = 5.828427$. We note that in the computed NS points $\frac{b}{a} = 0.171573 = 3-2\sqrt{2}$ and $\frac{b}{a} = 5.828427 = 3+2\sqrt{2}$. At the NS points (6.13) has the multipliers $\pm i$, i.e. the point is a resonance R_4 point. Near the NS points the system (6.3) oscillates between 4 different values, i.e. (6.3) has a 4-cycle.

A 4-cycle is given by $C_4 = \{X_1^4, X_2^4, X_3^4, X_4^4\}$ where

$$X_1^4 = (0.132780, 0.746921), X_2^4 = (0.117325, 0.746921),$$

$$X_3^4 = (0.117325, 0.709596), X_4^4 = (0.132771, 0.709597).$$

This 4-cycle with the parameter values is depicted in Figure 6.3.

We remark that our analytical and numerical results show that the Cournot point (6.11) loses stability via a NS point. However, our results conflict with the bifurcation analysis that is given, in [88] and [89], Figures 5.3 and 7.3, respec-

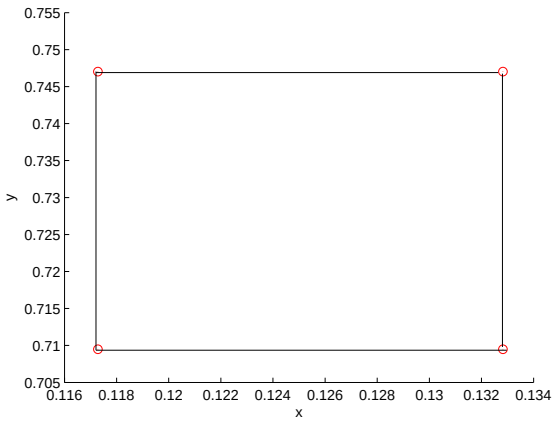


Figure 6.3: A 4-cycle near the NS point for the parameter values $(a, b) = (1, 0.171578)$.

tively, where the Cournot point is said to lose stability via a flip point and becomes chaotic via a cascade of flip bifurcations.

6.2 Cournot oligopoly with three firms

We keep the notation introduced up to now, but add a third firm, whose profit is W, whose output is denoted by z and whose marginal cost is c. The profits of the three oligopolists become:

$$\begin{aligned} U(x, y, z) &= \tfrac{x}{x+y+z} - ax, \\ V(x, y, z) &= \tfrac{y}{x+y+z} - by, \\ W(x, y, z) &= \tfrac{z}{x+y+z} - cz. \end{aligned} \tag{6.19}$$

The reaction function of this model is given by

$$\begin{aligned} x_{t+1} &= f(x_t, y_t, z_t) = \sqrt{\tfrac{y_t+z_t}{a}} - y_t - z_t, \\ y_{t+1} &= g(x_t, y_t, z_t) = \sqrt{\tfrac{x_t+z_t}{b}} - x_t - z_t, \\ z_{t+1} &= s(x_t, y_t, z_t) = \sqrt{\tfrac{y_t+x_t}{c}} - y_t - x_t. \end{aligned} \tag{6.20}$$

6.2.1 Stability analysis of the fixed points of the oligopoly model

Similar to the Duopoly Cournot model, the Cournot point of (6.20) is given by:

$$\begin{aligned} x &= \tfrac{2(b+c-a)}{a+b+c}^2, \\ y &= \tfrac{2(c+a-b)}{a+b+c}^2, \\ z &= \tfrac{2(a+b-c)}{a+b+c}^2, \end{aligned} \tag{6.21}$$

and is economically meaningful if $x \geq 0,\ y \geq 0$ and $z \geq 0$. Subsituting (6.21) into (6.20), we find the profits in the Cournot point:

$$\begin{aligned} \textstyle\prod_x &= \tfrac{(b+c-a)^2}{(a+b+c)^2}, \\ \textstyle\prod_y &= \tfrac{(c+a-b)^2}{(a+b+c)^2}, \\ \textstyle\prod_z &= \tfrac{(a+b-c)^2}{(a+b+c)^2}. \end{aligned} \tag{6.22}$$

The Jacobian matrix of (6.20) at the Cournot point (6.21) is:

$$J = \begin{pmatrix} 0 & \frac{b+c-3a}{4a} & \frac{b+c-3a}{4a} \\ \frac{c+a-3b}{4b} & 0 & \frac{c+a-3b}{4b} \\ \frac{a+b-3c}{4c} & \frac{a+b-3c}{4c} & 0 \end{pmatrix}. \tag{6.23}$$

The characteristic polynomial of (6.23) is given by:

$$p(\lambda) = -\lambda^3 + A\lambda + B = 0, \tag{6.24}$$

where

$$A = \frac{6(a^3+b^3+c^3) - 5(a^2+b^2+c^2)(a+b+c) + 30abc}{16abc}, \tag{6.25}$$

$$B = \frac{(a+b-3c)(a+c-3b)(b+c-3a)}{32abc}. \tag{6.26}$$

The coefficients A and B depend only on the ratios of marginal costs $h = \frac{b}{a}$ and $k = \frac{c}{a}$. These coefficients then take the forms:

$$A = \frac{6(1+h^3+k^3) - 5(1+h^2+k^2)(1+h+k) + 30hk}{16hk}, \tag{6.27}$$

$$B = \frac{(1+h-3k)(1+k-3h)(h+k-3)}{32hk}. \tag{6.28}$$

Let $\lambda_i,\ i = 1,2,3$ be the roots of (6.24). A Cournot point is stable when $|\lambda_i| < 1,\ \ i = 1,2,3$.
Suppose (6.24) has a root 1. Then we get:

$$p(1) = A + B - 1 = 0, \tag{6.29}$$

which is a criterion to detect a fold point. Substituting from (6.27) and (6.28), we get:

$$A + B - 1 = \frac{(h+k+1)^3}{32hk} = 0, \tag{6.30}$$

which cannot be fulfilled for positive h and k, i.e. there can be no transition from stability to instability through a fold bifurcation.

Similarly, let (6.24) has a root -1. Then we get:

$$p(-1) = A - B = 1, \tag{6.31}$$

which is a criterion to detect a flip bifurcation which defines a curve in (h,k)-space. The solution curve was computed numerically and is depicted in Figure 6.4. We note that along this curve, the Cournot point has a negative coordinate. So, the flip curve (6.31) is not economically meaningful.

Now we suppose the Cournot point loses stability via a NS bifurcation. The characteristic equation (6.24) can be decomposed in terms of its roots, i.e.

$$(\lambda_1 - \lambda)(\lambda_2 - \lambda)(\lambda_3 - \lambda) = 0. \tag{6.32}$$

Equating the coefficients of (6.24) and (6.32), leads to:

$$\lambda_1 + \lambda_2 + \lambda_3 = 0, \tag{6.33}$$

$$\lambda_1\lambda_2 + \lambda_1\lambda_3 + \lambda_2\lambda_3 = -A, \tag{6.34}$$

$$\lambda_1\lambda_2\lambda_3 = B. \tag{6.35}$$

Suppose λ_1 is real and $\lambda_{2,3} = \alpha \pm i\beta$ is a conjugate pair of complex eigenvalues. By (6.33), we have $\lambda_1 = -2\alpha$. Substituting λ_1 in (6.34) and (6.35), we obtain:

$$A = 3\alpha^2 - \beta^2, \tag{6.36}$$

$$B = -2\alpha(\alpha^2 + \beta^2). \tag{6.37}$$

The complex pair $\lambda_{2,3}$ crosses the unit circle when $|\lambda_{2,3}| = 1$, or equivalently $\alpha^2 + \beta^2 = 1$. Substituting $\beta^2 = 1 - \alpha^2$ into (6.36) and (6.37), gives:

$$A = 4\alpha^2 - 1, \tag{6.38}$$

and

$$B = 2\alpha. \tag{6.39}$$

By eliminating α between (6.38) and (6.39), we finally obtain:

$$B^2 - A = 1. \tag{6.40}$$

This is a criterion to detect a NS bifurcation point. A plot of (6.40), using CL_MATCONTM is given in Figure 6.4.

We now do a numerical continuation of the Cournot point. For the given parameter set $(h, k) = (1, 1)$, the Cournot point $(x, y, z) = (\frac{2}{9}, \frac{2}{9}, \frac{2}{9})$ is stable. We do a numerical continuation when the control parameter is h. The Cournot point loses stability via a subcritical NS point $x = (0.102701, 0.678395, 0.102701)$ when $h = 0.262966$. We continue the NS curve with two control parameters h and k, starting from the detected NS point:

```
tangent vector to first point found
label = CH, x =( 0.005416 0.766126 0.223011 0.230936
 0.780015 -0.208045 )
```

```
Normal form coefficient of CH = -1.641210e+000
label = CH, x =( 0.005416 0.223011 0.766126 0.780015
 0.230936 -0.208045 )
Normal form coefficient of CH = -1.641210e+000
label = CH, x =( 0.173952 0.004225 0.597590 1.282026
 0.296066  -0.208045 )
Normal form coefficient of CH = -4.433548e+000
label = CH, x =( 0.176926 0.001251 0.051501 4.330198
 3.377621  -0.208045 )
Normal form coefficient of CH = -5.770253e+002
label = CH, x =( 0.176926 0.051501 0.001251 3.377621
 4.330198  -0.208045 )
Normal form coefficient of CH = -5.770254e+002
label = CH, x =( 0.173952 0.597590 0.004225 0.296066
1.282026 -0.208045 )
Normal form coefficient of CH = -4.433547e+000
```

The continuation leads to a closed curve. The normal form coefficients along this curve change sign when crossing the CH bifurcation points. The Cournot point is economically relevant if $x \geq 0$, $y \geq 0$ and $z \geq 0$ in (6.21). These conditions define the region in parameter space that is bounded by the lines $b+c \geq a$, $c+a \geq b$ and $a+b \geq c$, respectively. These lines intersect the NS curve at some of the CH points. The stability region of the Cournot point that is economically relevant, is given in Figure 6.5. It is bounded by the three straight lines and the subcritical parts of the NS curve.

We remark that the NS bifurcation curve that is obtained by numerical continuation, Figure 6.5, coincidences with the NS curve defined by (6.40) and depicted in Figure 6.4. However, this curve differs from the NS curve that is given in [88] and [89], Figure 5.7 and 7.16, respectively.

6.3 A Cournot duoploy model of Kopel

6.3.1 Introduction

The first well-known model which gives a mathematical description of competition in a duopoly market dates back to the French economist Antoine Augustin

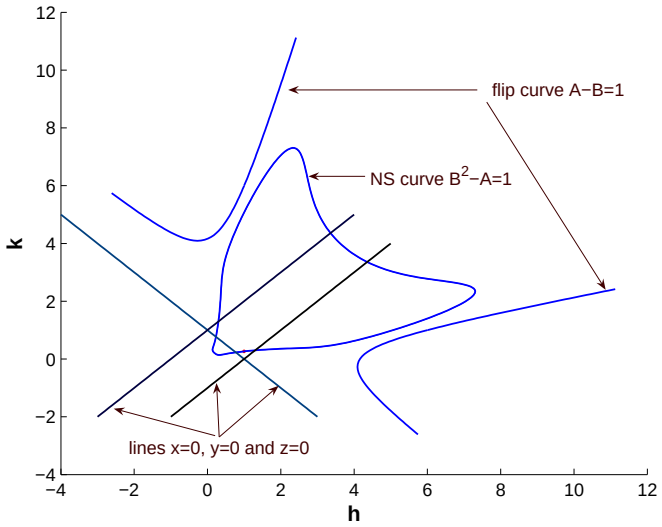


Figure 6.4: The flip bifurcation curve (6.31), the NS bifurcation (6.40) and the straight lines $x = 0,\ \ y = 0$ and $z = 0$ defined by (6.21), in (h, k) space.

Cournot [23] (1838) with the highlighted characteristics:

- Competing firms produce goods that are perfect substitutes.
- Both firms must consider the actions and reactions of the competitor.
- Each firm forms expectations of the other firm's output in order to determine a profit maximizing quantity to produce in the next time period (this situation is called strategic interdependence)

The model that he presented has been much studied for its ability to generate complex dynamics and also because of its more general foreshadowing of game theory. It has often been noted that the Cournot equilibrium is but a special case of the Nash-Equilibrium [72], the more general formulation used by modern industrial organization economists in studying oligopoly theory.

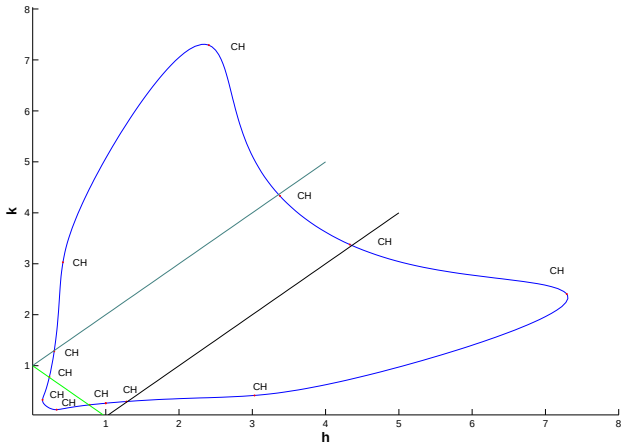


Figure 6.5: Economically relevant stability region of the Cournot point in (h, k) space.

Recently, several works have shown that the Cournot model may lead to complex behaviors such as cyclic and chaotic, see, for example [1, 57, 90, 91, 92]. Among the first to do this was Puu [90, 91] who found a variety of complex dynamics arising in the Cournot duopoly case including the appearance of attractors with fractal dimension. Dynamics of a Cournot game by players with bounded rationality has been studied in [5]. Local stability of a duopoly game with heterogeneous expectation has been studied in [4]. Multistability, cyclic and chaotic behaviour of a Cournot game have been studied in [15], where in the model the reaction functions have the form of the logistic map. Some preliminary results on the local bifurcations of a Kopel map were obtained in [57]. Explicit boundaries of local stability of the fixed point of a Kopel map have been derived in [2]. Basins of attraction in a Kopel map have been studied in [9]. Other studies on the dynamics of oligopoly models with more firms and other modifications include Ahmed and Agiza [6], Agiza [1] and Agiza et al. [3]. The development of complex oligopoly dynamics theory has been reviewed in [95].

In this study we consider the general case of a duopoly model, see [2], introduced in [57] with positive adjustment coefficient ρ. The main aim is to investigate the overal dynamic behaviour of the model when $\rho > 0$ and to compute stability domains of the first, second, third and fourth iterates of the map. In Section 2 we introduce the model and discuss the general stability and branching of the fixed points. In particular, we compute analytically the critical normal form coefficients in the case of period doubling bifurcations to reveal sub- or supercriticality. In Section 3 we concentrate on the economically relevant case $\rho \leq 1$ and numerically compute curves of codim 1 bifurcations and the critical normal form coefficients of codim 2 bifurcation points, using the CL_MATCONTM. These tools enable us to compute stability boundaries of 2, 3 and 4-cycles. Furthermore, by considering the critical normal form coefficients of the R4 resonance point, we determine the bifurcation scenario of the map near this point.

In Section 4 we briefly describe R3 and R2 bifurcation points in the region $\rho > 1$ which is of no interest for the economic model.

6.3.2 The map and the local stability analysis of its fixed points

The model that we use is a two-dimensional map described in [57, 2]. Two firms are homogeneous with regard to their expectation formation and the action effect on each other. The duopoly Kopel model assumes that at each discrete time *t* the two firms produce the quantities $x_1(t)$ and $x_2(t)$ respectively, and decide their productions for the next period $x_1(t+1)$ and $x_2(t+1)$. The time evolution of the model is determined by the two-dimensional map T_K:

$$T_K : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} (1-\rho)x_1 + \rho\mu x_2(1-x_2) \\ (1-\rho)x_2 + \rho\mu x_1(1-x_1) \end{pmatrix}. \tag{6.41}$$

where ρ and μ are two model parameters. The positive parameter μ measures the intensity of the effect that one firm's actions has on the other firm. Firms do not change their productions according to the computed optimal productions (i.e. the 'logistic' reaction functions) but they prefer to choose a weighted average between the previous production and the computed one, with weights $1-\rho$ and ρ respectively; ρ is called the adjustment coefficient. The meaning of the model implies that the parameter $\rho \in [0,\ 1]$. However it is best to ignore this restriction

in a first global study of the properties of the model, cf. [2].

The fixed points of (6.41) and their stability were studied analytically in [2], §2.1-4. We summarize the obtained results briefly. For $\rho \neq 0$, the fixed points of (6.41) are the solutions to:

$$x_1^* = \mu x_2^*(1 - x_2^*), \;\; x_2^* = \mu x_1^*(1 - x_1^*). \tag{6.42}$$

Besides the trivial solution $E_1 : (x_1^*, x_2^*) = (0, 0)$, a positive symmetric fixed point exists for $\mu > 1$, given by $E_2 : (x_1^*, x_2^*) = (\frac{\mu-1}{\mu}, \frac{\mu-1}{\mu})$.

Two further nonsymmetric Nash-Equilibria, given by

$$E_3 : (x_1^*, x_2^*) = (\frac{\mu + 1 + \sqrt{(\mu+1)(\mu-3)}}{2\mu}, \frac{\mu + 1 - \sqrt{(\mu+1)(\mu-3)}}{2\mu}), \tag{6.43}$$

and its $(x_1, x_2) \mapsto (x_2, x_1)$ reflection E_4, exist for $\mu \geq 3$.

The study of the local stability of fixed points is based on the linearization of (6.41). In an equilibrium point the Jacobian $J(x_1, x_2)$ of (6.41) has the eigenvalues:

$$\lambda_{1,2} = (1 - \mu) \pm \rho\mu\sqrt{(1 - 2x_1)(1 - 2x_2)}. \tag{6.44}$$

Depending on the values of x_1 and x_2, these may be real or form a conjugate complex pair. A fixed point of (6.41) is stable if

$$|\lambda_j| < 1, \;\; j = 1, 2. \tag{6.45}$$

Proposition 6.3.1. *The equilibrium solution E_1 is asymptotically stable for $(\mu, \rho) \in \Omega^S(E_1)$ where*

$$\Omega^S(E_1) = \left\{ (\mu, \rho) | 0 < \mu < 1, 0 < \rho < \rho_1(\mu) = \frac{2}{1 + \mu} \right\}.$$

It loses stability via a flip bifurcation when crossing the threshold $\rho_1(\mu), 0 < \mu < 1$ and via branching along $\mu = 1$.

Proof. The stability boundaries of E_1 can be derived by imposing the stability conditions (6.45). These boundaries were computed in [2] and are presented in Figure 6.6 ($\Omega^S(E_1)$).

What remains to be proved is that E_1 loses its stability and bifurcates to a new branch of fixed points at $\mu = 1$. To do this we show that the discriminant of the *algebraic branching equation* (ABE), §2.2, is positive. We consider the Jacobian matrix $F_X = [T_x - I|T_\mu]$, evaluated in E_1 that is:

$$\begin{pmatrix} -\rho & \rho\mu & 0 \\ \rho\mu & -\rho & 0 \end{pmatrix}. \tag{6.46}$$

This matrix is clearly rank deficient along $\mu = 1$. We first compute vectors ϕ_1, ϕ_2 and ψ which form a basis for the null space of $N([(T_K)_x - I|(T_K)_\mu])$ and $N([(T_K)_x - I|(T_K)_\mu]^*)$ respectively. Possible choices are:

$$\phi_1 = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0)^T, \;\; \phi_2 = (0, 0, 1)^T, \;\; \psi = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})^T.$$

Now we consider the ABE :

$$c_{11}\alpha^2 + 2c_{12}\alpha\beta + c_{22}\beta^2 = 0, \tag{6.47}$$

where $c_{jk} = \langle \psi, F^0_{YY}\phi_j\phi_k \rangle$, for $j, k = 1, 2$. Here the $2 \times 3 \times 3$ tensor F^0_{YY} is given by:

$$F^0_{YY}(:,:,1) = \begin{pmatrix} 0 & 0 & 0 \\ -2\mu\rho & 0 & \rho(1-x_1) - \rho x_1 \end{pmatrix}, \tag{6.48}$$

$$F^0_{YY}(:,:,2) = \begin{pmatrix} 0 & -2\mu\rho & \rho(1-x_2) - \rho x_2 \\ 0 & 0 & 0 \end{pmatrix}, \tag{6.49}$$

$$F^0_{YY}(:,:,3) = \begin{pmatrix} 0 & \rho(1-x_2) - \rho x_2 & 0 \\ \rho(1-x_1) - \rho x_1 & 0 & 0 \end{pmatrix}. \tag{6.50}$$

We now obtain $c_{11} = -\sqrt{2}\rho$, $c_{12} = \rho$, $c_{22} = 0$. So the discriminant of (6.47), $c_{12}^2 - c_{11}c_{22} = \rho^2 > 0$ is clearly positive. This shows that we have a branch point when $\mu = 1$.

Proposition 6.3.2. *E_2 is asymptotically stable for $(\mu, \rho) \in \Omega^S(E_2) = \Omega^S(E_{21}) \cup \Omega^S(E_{22})$ where:*

$$\Omega^S(E_{21}) = \left\{ (\mu, \rho) | 1 < \mu < 2, 0 < \rho < \rho_{21}(\mu) = \frac{2}{3-\mu} \right\},$$

and

$$\Omega^S(E_{22}) = \left\{(\mu,\rho)|2<\mu<3, 0<\rho<\rho_{22}(\mu)=\frac{2}{\mu-1}\right\}.$$

It loses stability via a flip bifurcation point on the boundaries of:

(i) $\rho=\rho_{21}(\mu), 1<\mu<2.$

(ii) $\rho=\rho_{22}(\mu), 2<\mu<3.$

Furthermore, it loses stability via a branch point when $\mu=1$ *and* $\mu=3$.

Proof. The stability domain of E_2 is given in [2] and presented in Figure 6.6 ($\Omega^S(E_2)$). By the same procedure as in Proposition 6.3.1, we can show that E_2 bifurcates to the branches of fixed points E_1 and E_3 (E_4) at $\mu=1$ and $\mu=3$, respectively.

Proposition 6.3.3. *E_3 (E_4) is asymptotically stable for* $(\mu,\rho)\in\Omega^S(E_3)=\Omega^S(E_{31})\cup\Omega^S(E_{32})$ *where*

$$\Omega^S(E_{31}) = \left\{(\mu,\rho)|3<\mu<1+\sqrt{5}, 0<\rho<\rho_{31}(\mu)=\frac{2}{1+\sqrt{5-(\mu-1)^2}}\right\},$$

and

$$\Omega^S(E_{32}) = \left\{(\mu,\rho)|\mu>1+\sqrt{5}, 0<\rho<\rho_{32}(\mu)=\frac{2}{(\mu-1)^2-4}\right\}.$$

It loses stability :

(i) via a flip point when $\rho=\rho_{31}(\mu), 3<\mu<1+\sqrt{5}$.

(ii) via a Neimark-Sacker bifurcation point when $\rho=\rho_{32}(\mu), \mu>1+\sqrt{5}$.

Moreover, it loses stability and bifurcates to the branch of E_2 *fixed points along* $\mu=3$.

Proof. The stability boundaries of E_3 were computed in [2] and are sketched in Figure 6.6 ($\Omega^S(E_{3,4})$). It is easy to prove that E_3 bifurcates to a branch of fixed points E_2 at $\mu=3$, by the same procedure as in Proposition 6.3.1.

Proposition 6.3.4. *The flip bifurcation in Proposition 6.3.1 is subcritical.*

Proof. We show that E_1 undergoes a subcritical flip bifurcation when $\rho = \frac{2}{1+\mu}$, $0 < \mu < 1$. It is sufficient to show that the critical normal form coefficient b,

$$b = \frac{1}{6}\left\langle p, C(q,q,q) + 3B(q,(I - A^{(1)})^{-1}B(q,q))\right\rangle, \tag{6.51}$$

derived by the parameter-dependent center manifold reduction is negative, see §2.3.2, where $A^{(1)}$ is the Jacobian of (6.41) at E_1, $B(.,.)$ and $C(.,.,.)$ are the second and third order multilinear forms respectively, p and q are the left and right eigenvectors of $A^{(1)}$ for the eigenvalue -1, respectively. These vectors are normalized by $\langle p, q\rangle = 1$, $\langle q, q\rangle = 1$, where $\langle .,.\rangle$ is the standard scalar product in $\mathbb{R}^2$. We obtain:

$$q = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = p = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix}, \tag{6.52}$$

$$[B(q,q)]_1 = \sum_{j,k=1}^{n} \frac{\partial^2((1-\rho)x_1 + \rho\mu x_2(1-x_2))}{\partial x_j \partial x_k} q_j q_k = -2\rho\mu q_2 q_2 = -\rho\mu, \tag{6.53}$$

$$[B(q,q)]_2 = \sum_{j,k=1}^{n} \frac{\partial^2((1-\rho)x_2 + \rho\mu x_1(1-x_1))}{\partial x_j \partial x_k} q_j q_k = -2\rho\mu q_1 q_1 = -\rho\mu. \tag{6.54}$$

Let $\zeta = (I - A^{(1)})^{-1}B(q,q)$, then we have $\zeta = \begin{pmatrix} \frac{\mu}{\mu-1} \\ \frac{\mu}{\mu-1} \end{pmatrix}$ and find

$$[B(q,\zeta)]_1 = -\rho\mu q_2\zeta_2 = \sqrt{2}\frac{\rho\mu^2}{\mu-1}, \quad [B(q,\zeta)]_2 - \rho\mu q_1\zeta_1 = -\sqrt{2}\frac{\rho\mu^2}{\mu-1}. \tag{6.55}$$

Since the third order multilinear form $C(q,q,q)$ is zero, the critical normal form coefficient b is given by $b = \frac{\rho\mu^2}{\mu-1}$. It is clear that $b < 0$, since $0 < \mu < 1$ and $\rho > 0$ in $\Omega^S(E_1)$.

Proposition 6.3.5. *The flip point in Proposition 6.3.2 is sub- or supercritical in the cases (i) and (ii) respectively.*

Proof. First we consider the case (*i*) and show that the flip point is subcritical. It is sufficient to show that $b < 0$ where b is defined in (6.51). The normalized left and right eigenvectors for $A^{(2)}$ are given by:

$$q = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = p = \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix}, \tag{6.56}$$

where $A^{(2)}$ is the Jacobian of (6.41) at E_2. $B(q,q)$ is given by:

$$[B(q,q)]_1 = -2\rho\mu q_2 q_2 = -\rho\mu, \;\; [B(q,q)]_2 = -2\rho\mu q_1 q_1 = -\rho\mu. \tag{6.57}$$

We proceed with the computation of $\zeta = (I - A^{(2)})^{-1} B(q,q)$ and obtain:

$$\zeta = \begin{pmatrix} \frac{\mu}{1-\mu} \\ \frac{\mu}{1-\mu} \end{pmatrix}, \;\; b = \frac{\rho\mu^2}{1-\mu}. \tag{6.58}$$

So $b < 0$, since $1 < \mu < 2$ and $\rho > 0$ in $\Omega^S(E_2)$.

In case (*ii*) we obtain $b = \frac{\rho\mu^2}{3(\mu-1)}$. So $b > 0$, since $2 < \mu < 3$ and $\rho > 0$ in $\Omega^S(E_2)$.

Proposition 6.3.6. *The flip bifurcation in Proposition 6.3.3 is subcritical.*

Proof. Similar to the previous cases we show that the critical normal form coefficient $b < 0$. The Jacobian matrix (6.41) at E_3 (E_4) is:

$$A^{(3)} = \begin{pmatrix} 1-\rho & -\rho(1+\sqrt{(\mu+1)(\mu-3)}) \\ -\rho(1-\sqrt{(\mu+1)(\mu-3)}) & 1-\rho \end{pmatrix}, \tag{6.59}$$

and has a multiplier -1 when $\rho = \frac{2}{1+\sqrt{5-(\mu-1)^2}}$, $3 < \mu < 1+\sqrt{5}$. The left and right eigenvectors associated to the eigenvalue -1 are given by:

$$q = \begin{pmatrix} -\sqrt{4-\mu^2+2\mu} \\ -1+\sqrt{-3+\mu^2-2\mu} \end{pmatrix}, \;\; p = \begin{pmatrix} -1+\sqrt{-3+\mu^2-2\mu} \\ -\sqrt{4-\mu^2+2\mu} \end{pmatrix}. \tag{6.60}$$

To avoid complicated computations we do not normalize p and q, since rescaling does not change the sign of b provided $\langle p, q\rangle > 0$ (it can be proved easily that this is the case). $B(q,q)$ is computed as:

$$B(q,q) = \begin{pmatrix} \frac{4\mu(-1+\sqrt{(-3+\mu^2-2\mu)^2})}{(1+\sqrt{4-\mu^2+2\mu})(-4+\mu^2-2\mu)} \\ -\frac{4\mu}{(1+\sqrt{4-\mu^2+2\mu})} \end{pmatrix}. \tag{6.61}$$

The vector $\zeta = (I - A^{(3)})^{-1}B(q,q)$, is given by:

$$\zeta = \begin{pmatrix} \frac{2\mu(-1+\sqrt{(-3+\mu^2-2\mu)^2})}{(\mu+1)(\mu-3)(-4+\mu^2-2\mu)} + \frac{2(1+\sqrt{(\mu+1)(\mu-3)})}{(\mu+1)(\mu-3)} \\ \frac{2(-1+\sqrt{(\mu+1)(\mu-3)})\mu(-1+\sqrt{(-3+\mu^2-2\mu)^2})}{(\mu+1)(\mu-3)(-4+\mu^2-2\mu)} - \frac{2\mu}{(\mu+1)(\mu-3)} \end{pmatrix}. \tag{6.62}$$

$B(q,\zeta)$ can be computed from (6.60) and (6.62):

$$B(q,\zeta) = \begin{pmatrix} -\frac{8(12-4\mu^2+8\mu+\sqrt{(-3+\mu^2-2\mu)}\mu^2-2\sqrt{(-3+\mu^2-2\mu)}\mu)(-1+\sqrt{(-3+\mu^2-2\mu)})\mu^2}{(4-\mu^2+2\mu)^{\frac{3}{2}}(\mu-3)(\mu+1)(1+\sqrt{(4-\mu^2+2\mu)})} \\ -\frac{8(-6-6\sqrt{(-3+\mu^2-2\mu)}+2\mu^2-4\mu+\sqrt{(-3+\mu^2-2\mu)}\mu^2-2\sqrt{(-3+\mu^2-2\mu)}\mu)\mu^2}{(-4+\mu^2-2\mu)(\mu-3)(\mu+1)(1+\sqrt{4-\mu^2+2\mu})} \end{pmatrix}. \tag{6.63}$$

Finally the normal form coefficient b can be computed:

$$b = -\frac{576(\mu^2-2\mu-2\sqrt{-3+\mu^2-2\mu}-2)\mu^2}{(1+\sqrt{4-\mu^2+2\mu})(-4+\mu^2-2\mu)^2}. \tag{6.64}$$

We will prove that the factor $h_1(\mu) = \mu^2 - 2\mu - 2(\sqrt{-3+\mu^2-2\mu}+1)$ in (6.64) is positive when $3 < \mu < 1+\sqrt{5}$. Equivalently we have to prove that $(\mu^2-2\mu-2)^2 - 4(\mu^2-2\mu-3) \geq 0$. Since $\frac{dh_1}{d\mu}(\mu) = 4(\mu-1)(\mu^2-2\mu-4) < 0$, $h_1(3) = 1$ and $h_1(1+\sqrt{5}) = 0$, we conclude $h_1 \geq 0$. So $b < 0$.

We remark that our numerical evidence indicates that the Neimark-Sacker bifurcation in Proposition 6.3.3 is supercritical. This is based on the numerical computation of the normal form coefficient d, §2.3.3. However, we were not able to prove this analytically.

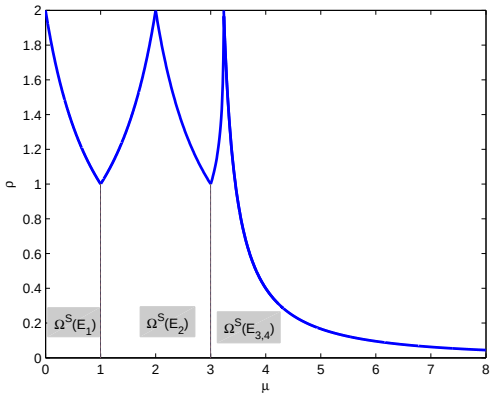


Figure 6.6: Stablility regions of $E_i, i = 1, 2, 3, 4$.

6.3.3 Numerical bifurcation in the economically relevant region

In this section we concentrate on the region $\rho \leq 1$ which is economically relevant. Since a complete analytical bifurcation study of the iterates of (6.41) is not feasible, we perform a numerical bifurcation analysis by using CL_MATCONTM.

Numerical bifurcation of E_2

By continuation of E_2 with $\mu = 2.5$ and ρ free, we see that E_2 loses stability via a supercritical PD point when crossing the hyperbola $\rho = \rho_{22}(\mu)$. A stable 2-cycle is given by $C_2 = \{X_1^2, X_2^2\}$ where $X_1^2 = (0.658212, 0.658212), X_2^2 = (0.527341, 0.527341)$, for $\rho = 1.366229$. This 2-cycle loses stability at a supercritical PD point (of the second iterate) for $\rho = 1.490763$. A stable 4-cycle is given by $C_4 = \{X_1^4, X_2^4, X_3^4, X_4^4\}$ where $X_1^4 = (0.3851221, 0.479532), X_2^4 = (0.745563, 0.649279), X_3^4 = (0.479532, 0.385122), X_4^4 = (0.649279, 0.745563)$.

The multipliers of the fixed point of the fourth iterate in X_1^4 are 0.406438 and 0.129274. This 4-cycle with the parameter values is depicted in Figure 6.7. We note that the 4-cycle is invariant under the reflection $(x_1, x_2) \mapsto (x_2, x_1)$. This

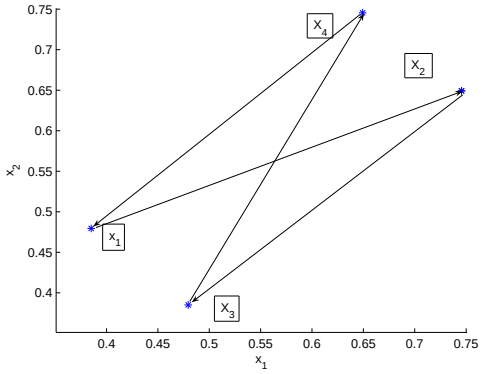


Figure 6.7: A stable 4-cycle for $\rho = 1.509191$ and $\mu = 2.5$.

4-cycle loses stability via a supercritical Neimark-Sacker bifurcation.

Stability regions of the 2-cycles ($\Omega_2^{S,i},\ i = 1,2,3$) and 4-cycles ($\Omega_4^{S,i},\ i = 1,2,3$) are given in Figure 6.8. They stretch into the economically relevant region $\rho \leq 1$. In this figure the regions $\Omega_2^{S,2}$ and $\Omega_4^{S,2}$ indicate bistability of 2- and 4-cycles with E_3 (E_4), respectively. We note that the stability region of the 2-cycle is bounded by the PD^2 curve and a curve of branch points of the second iterate, when $\mu \geq 3$. This curve of branch points bifurcates from the LPPD point on the PD curve of the original map. This curve is shown by $*$ in Figure 6.8 and is completely in the economically relevant region. We note that the LPPD point is on the boundary of the economically relevant region.

Numerical bifurcation study of E_3 (E_4)

We now do a continuation of the fixed point E_3 starting from $\mu = 4,\ \rho = 0.1$ in the stable region bounded by the curve $\rho = \rho_{32}(\mu)$. The parameter ρ is free, we call this *Run 1*:

```
label = NSm, x = ( 0.904508 0.345492 0.400000 )
normal form coefficient of NSm = -7.372800e+000
```

A supercritical Neimark-Sacker bifurcation point is detected for $\rho = 0.4$.

Thus, the fixed point E_3 (E_4) is transformed from stable to unstable through a NS point at which a closed invariant curve is created around the unstable fixed point E_3 (E_4). We now compute the Neimark-Sacker curve, by starting from the NS point in *Run 1*, with free parameters μ and ρ, this is *Run 2*:

```
label = R4  ,
x = ( 0.849938 0.439960 1.000000 3.449490 0.0000 )
normal form coefficient of R4 :
A = -3.000000e+000 + -2.019371e-017 i
label = R3 ,
 x =( 0.825542 0.476627 1.500000 3.309401 -0.500)
normal form coefficient of R3 :
Re(c_1) = -1.333333e+000
label = R2  ,
x =( 0.809017 0.500000 2.000000 3.236068 -1.00 )
normal form coefficient of R2 :
 [c , d] = 1.340433e+003, -3.351046e+003
```

A picture of the Neimark-Sacker curve of *Run 2* is given in Figure 6.8.

Since the R2 and R3 points are not in the region $\rho \leq 1$ we postpone their study to §6.3.4. We now consider the R4 point in *Run 2*. Since $|A| > 1$, two cycles of period 4 of the map are born. A stable 4-cycle for $\rho = 0.990844$ and $\mu = 3.466353$ is given by $C_4 = \{X_1, X_2, X_3, X_4\}$ where $X_1 = (0.841774, 0.407047)$, $X_2 = (0.836685, 0.461186)$, $X_3 = (0.861140, 0.473539)$, $X_4 = (0.864133, 0.4150395)$. We present this cycle in Figure 6.9. We note that it is not invariant under the reflection $(x_1, x_2) \mapsto (x_2, x_1)$. The multipliers in X_1 are $\lambda_1 = 0.901140$ and $\lambda_2 = 0.675526$, confirming the stability of the 4-cycle.

To determine the stability domain of the 4-cycle we compute in *Run 3* two branches of fold curves of the fourth iterate, emanating from the R_4 point, by switching at the R4 point. These fold curves exist because $|A| > 1$, where A is the normal form coefficient of the R4 point. The stable fixed points of the fourth iterate exist in the wedge between the two fold curves. We note that there is no bistability with fixed points of the original map.

```
label = CP,
 x = ( 0.849982 0.439945 0.999745 3.449889 )
normal form coefficient of CP s= 4.009280e+002
label = LPPD ,
```

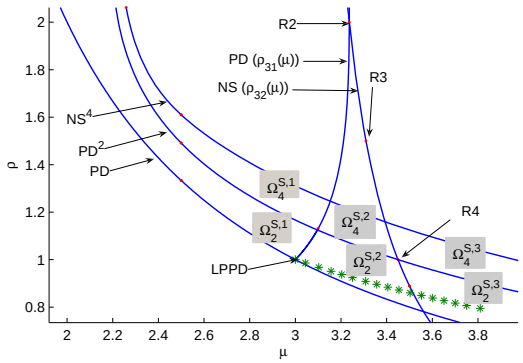


Figure 6.8: The Neimark-Sacker curve of *Run 2*, the flip curve of *Run 6* of Section 6.3.4 and the curve of branch points of the second iterate, the stability regions of Ω_2^S and Ω_4^S in (μ, ρ) space.

```
x = ( 0.841586 0.354516 0.935299 3.566686 )
normal form coefficient of LPPD :
[a/e , be]= 2.574002e+000, -5.829597e+001,
label = CP ,
 x = ( 0.849982 0.439945 0.999745 3.449889 )
normal form coefficient of CP s= 4.009280e+002
label = LPPD ,
 x = ( 0.836428 0.522216 1.071080 3.486079 )
normal form coefficient of LPPD :
[a/e , be]= 3.733856e+000, -2.471512e+001,
```

Two cusps, CP, and two LPPD bifurcation points are detected on the fold branches of the fourth iterate. The CP points are merely the R4 point from which we started. We can further compute the stability boundaries of the 4-cycle by computing the flip curve of the fourth iterate rooted at the detected LPPD points. The stable region Ω_4^S of C_4 is bounded by two fold curves and a flip curve of the fourth iterate, see Fig 6.10. Furthermore, if we continue the fixed point of the fourth iterate starting from X_1, it loses stability via a supercritical PD point where $\mu = 3.545530$. It means that a stable 8-cycle is born when $\mu > 3.545530$.

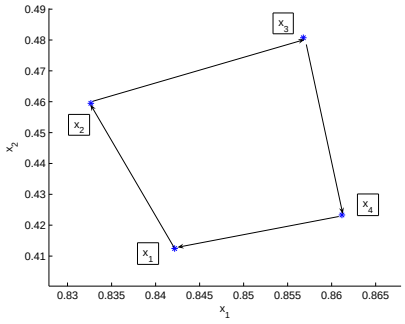


Figure 6.9: A stable 4-cycle for $\rho = 0.9999617$ and $\mu = 3.449802$.

We note that we have bistability of three different 4-cycles in a region bounded by the curves of the PD of the second iterate, a fold and the PD curve of the fourth iterate. This region is shown as $\Omega^S_{4,4}$ in Figure 6.11. Furthermore, we have a small bistability region of two 4-cycles and a 2-cycle. This bistability region is shown as $\Omega^S_{2,4}$ in Figure 6.11.

6.3.4 Bifurcations of E_3 (E_4) in the region $\rho > 1$

Now we consider the R2 point computed in *Run 3* of Section 6.3.3. Since the first component of the normal form coefficient $c = 1.340433e + 003$ is positive, the bifurcation scenario near the R2 point is analogous to [62], Fig. 9.9 (case $s = 1$). For the parameter values in the wedge between the PD (ρ_{31}) and NS (ρ_{32}) curves, the map has an unstable 2-cycle that coexists with a stable fixed point.

Next we consider the resonance 1:3 point in *Run 3* of Section 6.3.3. Since its normal form coefficient is negative, the bifurcation picture near the R3 point is qualitatively the same as presented in [62], Fig. 9.12. In particular, there is a region near the R3 point where a stable invariant closed curve coexists with an unstable equilibrium. For parameter values close to the R3 point, the map has a saddle cycle of period three.

Furthermore, a Neutral Saddle bifurcation curve of fixed points of the third iterate emanates. We compute this curve by branch switching at the R3 point of

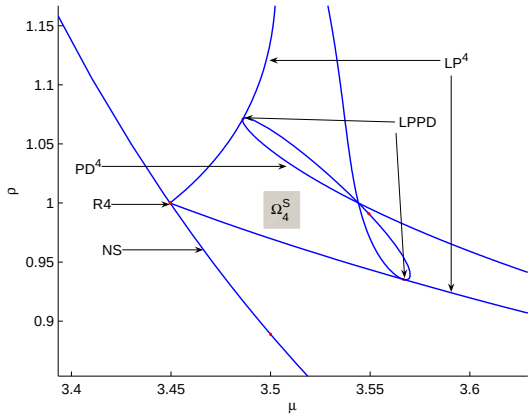


Figure 6.10: Two fold bifurcation curves of the fourth iterate emanate from the R4 point in (μ, ρ) space.

Run 5. This curve is presented in Figure 6.12. Further, a stable 3-cycle exists not far from the R_3 point(this is not guaranteed by the theory but it is found in many examples, e.g [41]). The stability region of this cycle is bounded by *fold* and NS bifurcation curves of the third iterate of the map (Ω_3^S). These boundary curves are given in Figure 6.12. We have bistability of the fixed point E_3 (E_4) with the fixed point of the third iterate of the map in the region that is bounded by the fold and NS curves and the hyperbola $\rho = \rho_{32}(\mu)$.

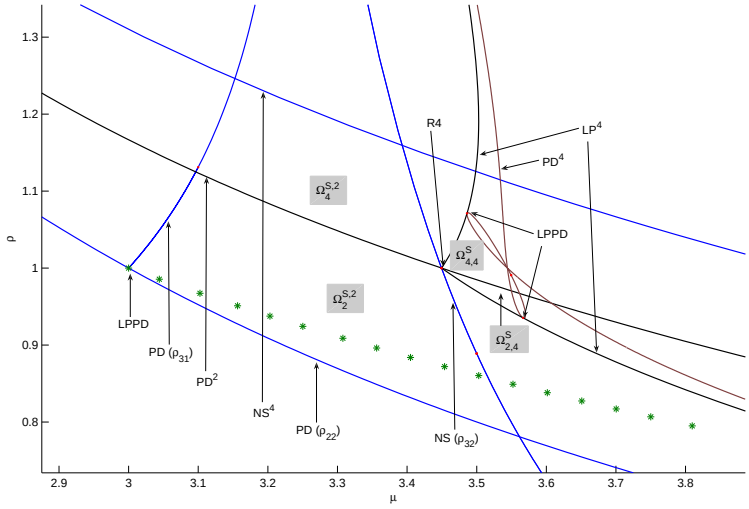


Figure 6.11: Bistability regions of 4-cycles, 2-cycles and fixed points.

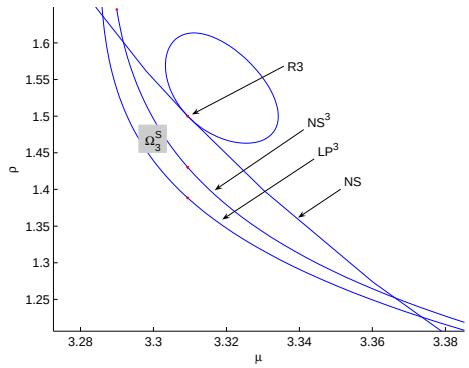


Figure 6.12: Two stability boundary curves (LP and NS) for the stable 3-cycle, together with the NS curve of *Run 3* and the curve of Neutral Saddle bifurcation points of the third iterate.

Chapter 7

Numerical continuation of connecting orbits of maps

The accurate computation of orbits connecting fixed points of an iterated map, and the study of associated topological properties have long been recognized as a very important problem both in the theory of nonlinear dynamical systems and in a variety of applied problems, e.g. in models for economical, biological, and physical phenomena. Indeed, as discovered by Poincaré and Birkhoff, such orbits may generate rich dynamics. For example, an orbit that connects a hyperbolic fixed point to itself (a homoclinic orbit) generically implies the existence of an infinite number of periodic orbits nearby, see [99, 74, 98] and tutorial presentations in [51, 103, 79]. As discovered in [34, 35, 48], the appearance of a pair of such homoclinic orbits is accompanied by an infinite sequence of fold and period-doubling bifurcations of periodic orbits, for more details see [75, 86], as well as [62]. Moreover, since a homoclinic orbit of a planar map belongs to the intersection of the stable and the unstable invariant curves of a saddle fixed point, such orbits can be involved in the destruction of a closed invariant curve which is born, for example, at a Neimark-Sacker bifurcation [73, 96, 97]. This destruction mechanism has been studied in [8, 17].

Numerical methods for bifurcation analysis of maps have received considerable attention recently and are supported by existing software. Algorithms for

the computation of the one-dimensional manifolds are implemented in DSTOOL [58, 32] and DYNAMICS [108, 109], while those for the continuation of homoclinic orbits and their tangencies [12] are implemented in an AUTO-driver [107].

This chapter starts with the basic concepts of the connecting orbits of a map. Then we introduce continuation of invariant subspaces in a novel way, using only linear algebra arguments. We continue with the continuation of heteroclinic connections and its implementation in CL_MATCONTM. We proceed with the computation of the symbolic Jacobian of the defining systems of the connecting orbits and continuation of homoclinic connections. We continue this chapter with the computation of one-dimensional stable and unstable manifolds. Finally, to illustrate the implemented techniques, we consider a generalized Hénon map. We compute branches of heteroclinic connections of the second iterate, and continue the corresponding heteroclinic tangencies in two parameters. We also consider continuation of a homoclinic connection and the continuation of the corresponding homoclinic tangencies.

A part of this chapter was submitted for publication [55].

7.1 Continuation of heteroclinic connections

We consider the J-th iterate of a map at some parameter as follows:

$$x \mapsto f^{(J)}(x,\alpha), \quad f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n, \tag{7.1}$$

where

$$f^{(J)}(x,\alpha) = \underbrace{f(f(f(\cdots f}_{J \text{ times}}(x,\alpha),\alpha),\alpha),\alpha).$$

A sequence $(x_k)_{k\in\mathbb{Z}}$ is called a *connecting orbit* of the map $f^{(J)}(\cdot,\alpha)$ at $\alpha = \bar{\alpha}$ if

$$\begin{aligned} &\lim_{k\to-\infty} x_k = x_{-\infty}, \\ &f^{(J)}(x_k,\bar{\alpha}) = x_{k+1}, \ \text{ for all } \ k \in \mathbb{Z} \\ &\lim_{k\to+\infty} x_k = x_{+\infty}. \end{aligned} \tag{7.2}$$

It is called *homoclinic* if $x_{-\infty} = x_{+\infty}$ and *heteroclinic* otherwise. From a geometrical point of view, the connecting orbit lies in the intersection of the unstable

manifold $W^u_{-\infty}$ of $x_{-\infty}$ and the stable manifold $W^s_{+\infty}$ of $x_{+\infty}$. A connecting orbit is *regular* if $x_{-\infty}$ and $x_{+\infty}$ are hyperbolic and the stable manifold $W^u_{+\infty}$ and the unstable manifold $W^s_{-\infty}$ have transversal intersections at x_k for all $k \in \mathbb{Z}$.

Degenerate cases occur when either the orbit loses transversality or one of its fixed points becomes nonhyperbolic. We will deal only with the former case, i.e. the case of nontransversality. In the latter case the unstable and center-stable manifolds have a transversal intersection, which produces a connecting orbit with a singular endpoint. In the simplest case there is precisely one multiplier 1 or -1, or one conjugate pair of multipliers of $f^{(J)}(x,\alpha)$ on the unit circle. This gives us the *saddle-fold, saddle-flip, saddle-Neimark-Sacker* connecting orbits, respectively, see e.g. [52, 13].

The corresponding numerical problem, for a regular heteroclinic connection between hyperbolic fixed points x_1 and x_N of (7.1), is that of finding a solution $(x_k)_{k=1,2,\ldots,N}$ of the following system [12]:

$$\begin{aligned} &x_1 = f^{(J)}(x_1,\alpha),\\ &x_{k+1} = f^{(J)}(x_k,\alpha),\ \ k = 2,\ldots,N-2\\ &x_N = f^{(J)}(x_N,\alpha) \end{aligned} \tag{7.3}$$

such that $(x_k)_{k=2..,N-1}$ leave x_1 along its unstable manifold and enter x_N along its stable manifold. These requirements are then substituted by *projection boundary conditions* which place x_2 and x_{N-1} into the corresponding tangent spaces [12].

We use an improved algorithm for locating and continuing connecting orbits, which includes an algorithm for the continuation of invariant subspaces (CIS) as described in [25, 27]. Assume the eigenvalues of $(f^{(J)}(x_1,\alpha))_x$ and $(f^{(J)}(x_N,\alpha))_x$ are ordered, respectively, as follows:

$$|\lambda^U_n| \le \ldots \le |\lambda^U_{n_U+1}| < 1 < |\lambda^U_1| \le \ldots \le |\lambda^U_{n_U}|,$$
$$|\lambda^S_1| \le \ldots \le |\lambda^S_{n_S}| < 1 < |\lambda^S_{n_S+1}| \le \ldots \le |\lambda^S_n|.$$

The algorithm requires the evaluation of various projections associated with the eigenspaces of $(f^{(J)}(x_1,\alpha))_x$ and $(f^{(J)}(x_N,\alpha))_x$. These projections are constructed using the real Schur factorizations.

$$(f^{(J)}(x_1,\alpha))_x = Q_1 R_1 Q_1^T,\ \ (f^{(J)}(x_N,\alpha))_x = Q_2 R_2 Q_2^T.$$

where Q_1, R_1, Q_2 and R_2 are $n \times n$-matrices.

The first factorization has been chosen so that the first n_U columns $q_1^U, \ldots, q_{n_U}^U$ of Q_1 form an orthonormal basis of the right invariant subspace S_1 of $(f^{(J)}(x_1, \alpha))_x$, corresponding to $\lambda_1^U, \ldots, \lambda_{n_U}^U$ and the remaining $n - n_U$ columns $q_{n_U+1}^U, \ldots, q_n^U$ of Q_1 form an orthonormal basis of the orthogonal complement $S_1^\perp$. Similarly, the first n_S columns $q_1^S, \ldots, q_{n_S}^S$ of Q_2 form an orthonormal basis of the right invariant subspace S_N of $(f^{(J)}(x_N, \alpha))_x$, corresponding to $\lambda_1^S, \ldots, \lambda_{n_S}^S$ and the remaining $n - n_S$ columns $q_{n_S+1}^S, \ldots, q_n^S$ of Q_2 form an orthonormal basis of the orthogonal complement $S_N^\perp$.

The problem of heteroclinic connections is to find a connection $\{x_n\}$ with:

- Stationary state conditions for the initial fixed point:

$$f^{(J)}(x_1, \alpha) - x_1 = 0, \tag{7.4}$$

- The iteration conditions

$$f^{(J)}(x_k, \alpha) - x_{k+1} = 0, \;\; k = 2, 3, \ldots, N-2, \tag{7.5}$$

- Stationary state conditions for the final fixed point:

$$f^{(J)}(x_N, \alpha) - x_N = 0, \tag{7.6}$$

- The left boundary conditions

$$\langle (x_2 - x_1), q_{n_{U+i}}^U \rangle = 0, \;\; i = 1, \ldots, n - n_U, \tag{7.7}$$

- The right boundary conditions

$$\langle (x_{N-1} - x_N), q_{n_{S+i}}^S \rangle = 0, \;\; i = 1, \ldots, n - n_S, \tag{7.8}$$

A regular zero of a system of equations (7.4), (7.5), (7.6), (7.7) and (7.8) corresponds to a transversal heteroclinic orbit. Thus, a zero for this system can be continued in one parameter.

In the computational process the conditions in (7.7) and (7.8) imply that we need to access the unstable and stable eigenspaces of the map (7.1) at the fixed points x_1 and x_N at each step of the continuation, respectively. It is not efficient to recompute these spaces from scratch in each continuation-step. In the next section we explain an algorithm for continuing the invariant subspaces S_1 and S_2 effectively. Contrary to [25, 27], our algorithm is purely based on linear algebra arguments.

7.1.1 Continuation of invariant subspaces

Let $A(\alpha) \in \mathbb{R}^{n\times n}$ denote $(f^{(J)}(x_1,\alpha))_x$. The basic continuation algorithm requires at each continuation step the computation of the orthogonal complement of the right invariant (unstable) n_U-dimensional subspace $S(\alpha)$ of $A(\alpha)$. In general, the function A is smooth in α, and it is important that $S(\alpha)$ be smooth also, as otherwise convergence difficulties can be expected.

We show how to constructively obtain smooth bases for the unstable eigenspace and its orthogonal complement.

Continuation of invariant subspaces was introduced in [25]. We introduce it in a novel way, using only linear algebra arguments. To justify our construction, we recall that in our continuation procedure we parameterize a solution branch in terms of so called pseudo-arclength; let s denote the pseudo-arclength variable. Thus, both fixed points x_1 and x_N as well as the parameter(s) α are smooth functions of s. The matrix-valued function $A : \alpha \in \mathbb{R}^{n_\alpha} \longmapsto \mathbb{R}^{n\times n}$ can thus be viewed as a smooth function from $s \in \mathbb{R} \longmapsto \mathbb{R}^{n\times n}$. As a consequence, we consider the continuation of invariant subspaces with respect to the scalar pseudo-arclength variable s. For this reason, we use the notation $A(s)$ for $A(\alpha)$.

We first consider x_1 and its unstable eigenspace. Suppose that initially we have the (real) block Schur factorization

$$A(0) = Q(0)R(0)Q^T(0), \;\; Q(0) = [Q_1(0) \;\; Q_2(0)], \tag{7.9}$$

where $A(0), R(0)$ and $Q(0)$ are $n \times n$-matrices, $Q(0)$ is orthogonal, $Q_1(0)$ has dimensions $n \times n_U$ and $R(0)$ is block upper triangular

$$R(0) = \begin{bmatrix} R_{11}(0) & R_{12}(0) \\ 0 & R_{22}(0) \end{bmatrix}, \tag{7.10}$$

where $R_{11}(0)$and $R_{22}(0)$ are $n_U \times n_U$- and $(n-n_U)\times(n-n_U)$-matrices, respectively. $R_{ii}(0), i = 1, 2$, are not required to be triangular. The columns of $Q_1(0)$ span the unstable invariant subspace $S(0)$ of $A(0)$, and the columns of $Q_2(0)$ span the orthogonal complement $S^{\perp}(0)$. We want to obtain a block Schur factorization for the matrix $A(s)$, close to $A(0)$.

Suppose that the matrix $A(s)$ has two groups of eigenvalues, $\Lambda_1(s)$ (with modulus > 1) and $\Lambda_2(s)$ (with modulus < 1), which stay disjoint for all s around 0. Then, in an interval about $s = 0$, we need a smooth factorization

$$A(s) = Q(s)R(s)Q^T(s), \;\; Q(s) = [Q_1(s) \;\; Q_2(s)], \tag{7.11}$$

where $R(s)$ is in block Schur form

$$R(s) = \begin{bmatrix} R_{11}(s) & R_{12}(s) \\ 0 & R_{22}(s) \end{bmatrix}, \tag{7.12}$$

Here, R_{11} has eigenvalues $\Lambda_1(s)$ and R_{22} has eigenvalues $\Lambda_2(s)$. As shown in [27], it is always possible to obtain a smooth path of block Schur factorizations that satisfies (7.11) and (7.12). However, this smooth path is usually not unique.

Thus we can write

$$Q(s) = Q(0)U(s), \;\; \text{with} \;\; U(0) = I, \tag{7.13}$$

so that we only need to compute the $n \times n$-matrix $U(s)$. Partitioning $U(s)$ in blocks of the same size as $R(0)$ in (7.10):

$$U(s) = [U_1(s) \;\; U_2(s)] = \begin{bmatrix} U_{11}(s) & U_{12}(s) \\ U_{21}(s) & U_{22}(s) \end{bmatrix}, \tag{7.14}$$

so $U_{11}(s)$ and $U_{22}(s)$ are $n_U \times n_U$- and $(n-n_U)\times(n-n_U)$-matrices, respectively.

We now show that we can always assume that $U_{11}(s)$ and $U_{22}(s)$ are symmetric positive-definite by redefining $Q(s)$ and $R(s)$ if necessary and that this defines $Q(s)$ and $R(s)$ in a unique way.

Proposition 7.1.1. *Suppose $Q(0)$, $R(0)$ are chosen such that (7.9) and (7.10) hold. Then for all s sufficiently close to 0 there exist a unique orthogonal matrix*

$Q(s) = [Q_1, Q_2]$ of size $n \times n$ such that the columns of Q_1 span the unstable invariant subspace of $A(s)$ and the columns of Q_2 span the orthogonal complement of the unstable eigenspace, and a unique block triangular matrix

$$R(s) = \begin{bmatrix} R_{11}(s) & R_{12}(s) \\ 0 & R_{22}(s) \end{bmatrix}, \tag{7.15}$$

of size $n \times n$ where R_{11} has eigenvalues $\Lambda_1(s)$ with modulus > 1 and R_{22} has eigenvalues $\Lambda_2(s)$ with modulus < 1, such that

$$A(s)Q(s) = Q(s)R(s), \tag{7.16}$$

and

$$Q^{-1}(0)Q(s) = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix}, \tag{7.17}$$

where the blocks U_{11} and U_{22} are symmetric positive definite (SPD).

Proof. Suppose that $Q(s)$ and $R(s)$ satisfy (7.15) and (7.16). Let $Q^{'}(s)$ and $R^{'}(s)$ be any other pair that satisfies (7.15) and (7.16). Then we must have

$$Q^{'}(s) = Q(s)T(s) = Q(s) \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix}, \tag{7.18}$$

where T is orthogonal, and also block diagonal.

Suppose also that in

$$Q^{-1}(0)Q^{'}(S) = \begin{bmatrix} U^{'}_{11} & U^{'}_{12} \\ U^{'}_{21} & U^{'}_{22} \end{bmatrix}, \tag{7.19}$$

both $U^{'}_{11}$ and $U^{'}_{22}$ are SPD. By (7.18) and (7.19), we have

$$Q^{-1}(0)Q^{'}(s) = Q^{-1}(0)Q(s) \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} = \begin{bmatrix} U^{'}_{11} & U^{'}_{12} \\ U^{'}_{21} & U^{'}_{22} \end{bmatrix}, \tag{7.20}$$

Or, equivalently

$$Q^{-1}(0)Q(s) = \begin{bmatrix} U'_{11}T_1^{-1} & U'_{12}T_2^{-1} \\ U'_{21}T_1^{-1} & U'_{22}T_2^{-1} \end{bmatrix}. \tag{7.21}$$

According to the polar decomposition of matrices [37], §4.2.10, A_1 and A_2, the upper left and lower right blocks of $Q^{-1}(0)Q(s)$ respectively, are uniquely presented as a product of an SPD matrix and an orthogonal matrix. This implies that T_1 and T_2 are uniquely defined. □

Since $U(0) = I$, there is an open interval about 0, call it I_0, where we can require that U_1 has the structure

$$U_1(s) = \begin{bmatrix} I \\ U_{21}U_{11}^{-1} \end{bmatrix} U_{11}. \tag{7.22}$$

Next, for all $s \in I_0$, we define

$$Y_U(s) = U_{21}(s)U_{11}^{-1}. \tag{7.23}$$

Using the orthogonality relation $U_1^T U_1 = I$, we get:

$$U_1^T(s)U_1(s) = U_{11}^T(s)U_{11}(s) + U_{21}^T(s)U_{21}(s) = I. \tag{7.24}$$

Using (7.24), we obtain

$$\begin{aligned} I + Y_U^T Y_U &= I + U_{11}^{-T}(s)U_{21}^T(s)U_{21}(s)U_{11}^{-1}(s) \\ &= I + U_{11}^{-T}(s)\left[I - U_{11}^T(s)U_{11}(s)\right]U_{11}^{-1}(s) \\ &= I + U_{11}^{-T}(s)U_{11}^{-1}(s) - I \\ &= U_{11}^{-T}(s)U_{11}^{-1}(s). \end{aligned}$$

Now because U_{11} is symmetric positive definite, U_{11}^{-1} is the unique square root of $I + Y_U^T Y_U$. This implies that we can rewrite (7.22) in terms of Y_U and choose U_{11} symmetric, to obtain

$$U_1 = \begin{bmatrix} I \\ Y_U \end{bmatrix} (I + Y_U^T Y_U)^{-\frac{1}{2}}. \tag{7.25}$$

In a similar way, for U_2 we use $U_2^T U_2 = I$ and $U_1^T U_2 = 0$, to eventually obtain, for every $s \in I_0$,

$$U(s) = \left[\begin{pmatrix} I \\ Y_U \end{pmatrix} (I + Y_U^T Y_U)^{-\frac{1}{2}} \quad \begin{pmatrix} -Y_U^T \\ I \end{pmatrix} (I + Y_U Y_U^T)^{-\frac{1}{2}} \right]. \tag{7.26}$$

Hence the columns of

$$Q_U(0) \begin{bmatrix} I \\ Y_U \end{bmatrix}, \tag{7.27}$$

form a base for the unstable eigenspace at x_1 and the columns of

$$Q_U^{\perp}(s) = Q_U(0) \begin{bmatrix} -Y_U^T \\ I \end{bmatrix}. \tag{7.28}$$

form a base for the orthogonal complement of the unstable eigenspace. We note that these bases are in general not orthogonal.

Thus, we need to find the matrix $Y_U \in \mathbb{R}^{(n-n_U)\times n_U}$ in (7.26). For any given $s \in I_0$, define $\hat{R}_{11}, \hat{R}_{12}, E_{21}$ and $\hat{R}_{22}$ by

$$Q^T(0) A(s) Q(0) = \begin{bmatrix} \hat{R}_{11} & \hat{R}_{12} \\ E_{21} & \hat{R}_{22} \end{bmatrix}, \tag{7.29}$$

where $\hat{R}_{11}$ is of size $n_U \times n_U$ and $\hat{R}_{22}$ is an $(n - n_U) \times (n - n_U)$ matrix.

By (7.11) and (7.12) we obtain the invariant subspace relation,

$$Q_2^T(s) A(s) Q_1(s) = 0. \tag{7.30}$$

Now we substitute $Q(s)$ given by (7.13), (7.26), and $A(s)$ obtained from (7.29) into (7.30)

$$[-Y_U \;\; I]\, Q^T(0) Q(0) \begin{bmatrix} \hat{R}_{11} & \hat{R}_{12} \\ \hat{E}_{21} & \hat{R}_{22} \end{bmatrix} Q^T(0) Q(0) \begin{bmatrix} I \\ Y_U \end{bmatrix} = 0, \tag{7.31}$$

to obtain the following algebraic Riccati equation for Y_U :

$$F(Y_U) = 0, \;\; F(Y_U) := \hat{R}_{22} Y_U - Y_U \hat{R}_{11} + E_{21} - Y_U \hat{R}_{12} Y_U. \tag{7.32}$$

We now look at x_N. In the same way we can compute a right invariant (stable) n_S-dimensional subspace $S(\alpha)$ of $A(\alpha)$.

First we consider $Q(\alpha) = [Q_1(\alpha)\ Q_2(\alpha)] \in \mathbb{R}^{n\times n}, Q_1(\alpha) \in R^{n\times n_S}, Q_2(\alpha) \in R^{n\times(n-n_S)}$ so that $Q_1(\alpha)$ spans $S(\alpha)$ and $Q_2(\alpha)$ spans the orthogonal complement $S^{\perp}(\alpha)$.

Using the same procedure, as used in the computation of the unstable subspace for x_1, we can obtain the relations

$$Q(s) = Q(0)U(s), \ \text{ with } \ U(0) = I, \tag{7.33}$$

and

$$U(s) = \left[\begin{pmatrix} I \\ Y_S \end{pmatrix} (I + Y_S^T Y_S)^{-\frac{1}{2}} \ \begin{pmatrix} -Y_S^T \\ I \end{pmatrix} (I + Y_S Y_S^T)^{-\frac{1}{2}} \right], \tag{7.34}$$

and eventually the algebraic Riccati equation for Y_S:

$$F(Y_S) = 0, \ \ F(Y_S) := \hat{R}_{22} Y_S - Y_S \hat{R}_{11} + E_{21} - Y_S \hat{R}_{12} Y_S. \tag{7.35}$$

Solving (7.35) for Y_S of size $(n - n_S) \times n_S$, enables us to compute the span of the stable invariant subspace of x_N and its orthogonal complement. If $Q_S(0)$ is the orthogonal matrix from the starting heteroclinc orbit, related to the stable invariant subspace, then a basis for the stable eigenspace in the new step at x_N is given by the columns of

$$Q_S(0) \begin{bmatrix} I \\ Y_S \end{bmatrix}. \tag{7.36}$$

A basis for the orthogonal complement of the subspace in the new step $Q_S^{\perp}$, is given by the columns of

$$Q_S^{\perp}(s) = Q_S(0) \begin{bmatrix} -Y_S^T \\ I \end{bmatrix}. \tag{7.37}$$

These bases are in general not orthogonal.

7.1.2 Implementation

We now discuss the implementation of the algorithm in CL_MATCONTM to continue the heteroclinic connection from the fixed point x_1 to the fixed point x_N.

- *Continuation variables*

  The continuation variables are stored in a K-vector, where $K = Nn+(n-n_U)n_U + (n-n_S)n_S + 1$, contaning:

  - A n-vector with the coordinates of the initial fixed point.
  - $(N-2)$ n-vectors with the coordinates of the mesh points $x_2, \ldots, x_{N-1}$.
  - A n-vector x_N with the coordinates of the final fixed point.
  - The vector Y_U^v, i.e. columnwise vectorized Y_U.
  - The vector Y_S^v, i.e. columnwise vectorized Y_S.
  - An active parameter ap.

- *Defining system*
  The defining systems consists of $Nn+(n-n_U)n_U+(n-n_S)n_S$ equations:

  - The initial fixed point constraint $f^{(J)}(x_1, \alpha) - x_1 = 0$.
  - The constraints $f^{(J)}(x_{j-1}, \alpha) - x_j = 0, \;\; j = 3, \ldots, N-1$.
  - The final fixed point constraint $f^{(J)}(x_N, \alpha) - x_N = 0$.
  - The rowwise vectorized Riccati equation (7.32) for Y_U.
  - The rowwise vectorized Riccati equation (7.35) for Y_S.
  - The initial boundary conditions (7.7).
  - The final boundary conditions (7.8).

- *Initialization*

  To implement the algorithm in CL_MATCONTM, we need to initialize the connection curve, i.e. we set the problem parameter vector α, mesh points $x_1, \ldots, x_N$, compute $Q_1(0)$ and $Q_2(0)$ corresponding to x_1 and x_N by (7.9) and initialize the vector $Y_U = 0$ and $Y_S = 0$ corresponding to the unstable

and stable eigenspaces of x_1 and x_N of sizes $((n - n_U) \times n_U)$ and $((n - n_S) \times n_S)$, respectively. We also set a global structure *hetds* containing the following fields:

- Dimension of the state space (*hetds.nphase*).
- Number of mesh points, including the two fixed end points (*hetds.npoints*)
- The iteration number of the map J (*hetds.niteration*)
- Mapfile where the map is defined (*hetds.mapfile*)
- Vector of starting values of parameters and index of the active parameter (*hetds.*P_0 and hetds.ActiveParams)
- Dimensions of the stable and unstable manifolds (*hetds.nu* and *hetds.ns*)
- The matrices Q_U and Q_S, bases for unstable and stable subspaces respectively (*hetds.*Q_U and *hetds.*Q_S)

- *Adaptation*
  At each continuation point a basis for the unstable eigenspace of x_1 is given by

$$hetds.Q_U \begin{bmatrix} I \\ Y_U \end{bmatrix},$$

  and for its orthogonal complement by

$$hetds.Q_U \begin{bmatrix} -Y_U^T \\ I \end{bmatrix}.$$

  However, these bases are not orthogonal. To restore orthogonality we must adapt Q_U from time to time. The base Q_U can be adapted using the *singular value decomposition* (SVD)

$$[U, S, V] = svd\left(hetds.Q_U \begin{bmatrix} I \\ Y_U \end{bmatrix}\right), \tag{7.38}$$

  where U and V are unitary matrices of sizes $n \times n$ and $n_U \times n_U$, respectively, and S is a diagonal matrix of size $n \times n_U$. An adapted orthogonal base of the unstable subspace is given by U. Then, the vector Y_U is set to zero.

  By using a similar procedure we can adapt the matrices Q_S and Y_S.

7.1.3 Computing the Jacobian matrix

To continue a connecting orbit, we need to compute the Jacobian matrix of the corresponding defining system. The Jacobian matrix can be computed by a finite difference approximation or by symbolic derivatives. However, using finite differences leads to an inaccurate orbit. Moreover, continuation that uses finite differences is much slower in comparison with the symbolic Jacobian.

To compute the Jacobian matrix we first initialize the Jacobian matrix J as a zero sparse matrix of size $k \times (k+1)$, where $k = Nn+(n-n_U)n_U+(n-n_S)n_S$. We then compute the nonzero entries of J by taking the derivatives of the *defining system* equations with respect to the continuation variables.

- For the constraints in (7.4) we set $J(1:n, 1:n) = A(x_1) - I$ where $A(x_1)$ is the Jacobian of (7.1) at x_1 and I is an identity matrix of size n. For derivatives of (7.4) w.r.t the control parameter we set $J(1:n, k+1) = (A_\alpha)_1(:, \text{hetds.ActiveParams})$, where $(A_\alpha)_1$ is the Jacobian of $f^{(J)}$ w.r.t parameter at x_1.

- For the $(N-3)$ constraints defined in (7.5), we get for $j = 3, \ldots, N-1$, $J((j-2)n+1 : (j-1)n, (j-2)n+1 : (j-1)n) = A(x_{j-1})$ and $J((j-2)n+1 : (j-1)n, (j-1)n+1 : jn) = -I$.

  For derivatives of (7.5) w.r.t the control parameter we set $J((j-2)n+1 : (j-1)n, k+1) = (A_\alpha)_j(:, \text{hetds.ActiveParams})$, where $j = 3, \ldots, N-1$ and $(A_\alpha)_j$ is the Jacobian w.r.t the control parameter at x_j.

- For the final fixed point constraint (7.6), J is computed as: $J((N-2)n+1 : (N-1)n, (N-1)n+1 : Nn) = A(x_N) - I$. For derivatives of (7.6) w.r.t the control parameter we set $J((N-2)n+1 : (N-1)n, k+1) = (A_\alpha)_N(:, \text{hetds.ActiveParams})$, where $(A_\alpha)_N$ is the Jacobian of $f^{(J)}$ w.r.t parameter at x_N.

- Now we compute the entries of J corresponding to (7.32) at x_1. First we consider the derivatives with respect to the components of Y_U. For simplicity of the computations we divide (7.32) into 3 terms $D_1 = \hat{R}_{22}Y_U$, $D_2 = -Y_U\hat{R}_{11}$, $D_3 = -Y_U\hat{R}_{12}Y_U$.

We start with D_1 whose derivatives with respect to the components of Y_U are written into a block matrix of size $((n-n_U)\times n_U)\times((n-n_U)\times n_U)$. D_1 is an $(n-n_U)\times n_U$ matrix with general form $(D_1)_{(j,i)} = \sum_{l=1}^{n-n_U}(\hat{R}_{22})_{(j,l)}(Y_U)_{(l,i)},\ \ j=1,\ldots,n-n_U,\ \ i=1,\ldots,n_U$. Hence all nonzero derivatives arise from the fact that the derivative of $(D_1)_{(j,i)}$ with respect to $(Y_U)_{(s,i)}$ is $(\hat{R}_{22})_{(j,s)},\ 0\le s\le n-n_U$. Now if $l=n(N-1)$ and $h=Nn$, then

- $(D_1)_{(j,i)}$ is at row position $l+i+(j-1)n_U$.
- $(Y_U)_{(s,i)}$ is at column position $h+s+(i-1)(n-n_U)$.

Therefore we update

$$J(l+i+(j-1)n_U, h+s+(i-1)(n-n_U)) := (\hat{R}_{22})_{(j,s)},$$

whenever $1\le j\le n-n_U$, $1\le i\le n_U$ and $1\le s\le n-n_U$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_1

```
for j = 1 : n − n_U
 for i = 1 : n_U
  idx1 = l + i + (j − 1) ∗ n_U;
  idx2 = h + 1 + (i − 1) ∗ (n − n_U);
  idx3 = h + (n − n_U) + (i − 1) ∗ (n − n_U);
  J(idx1, idx2: idx3) = J(idx1, idx2: idx3) + R̂22(j, 1 : n − n_U);
 end
end
```

- D_2 is an $(n-n_U)\times n_U$ matrix with general form

$$(D_2)_{(j,i)} = -\sum_{l=1}^{n_U}(Y_U)_{(j,l)}(\hat{R}_{11})_{(l,i)},\ \ j=1,\ldots,n-n_U,\ \ i=1,\ldots,n_U$$

Hence all nonzero derivatives arise from the fact that the derivative of $(D_2)_{(j,i)}$ with respect to $(Y_U)_{(j,s)}$ is $(-\hat{R}_{11})_{(s,i)},\ 1\le s\le n_U$. Now if $l=n(N-1)$ and $h=Nn$, then

– $(D_2)_{(j,i)}$ is at row position $l + i + (j-1)n_U$.
– $(Y_U)_{(j,s)}$ is at column position $h + j + (s-1)(n-n_U)$.

Therefore we update

$$J(l+i+(j-1)n_U, h+j+(s-1)(n-n_U)) :=$$
$$J(l+i+(j-1)n_U, h+j+(s-1)(n-n_U)) - (\hat{R}_{11})_{(s,i)},$$

whenever $1 \le j \le n-n_U$, $1 \le i \le n_U$ and $1 \le s \le n_U$. The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_2

```
for j = 1 : n − n_U
 for s = 1 : n_U
  idx1 = l + 1 + (j − 1) ∗ n_U;
  idx2 = l + n_U + (j − 1) ∗ n_U;
  idx3 = h + j + (s − 1) ∗ (n − n_U);
  J(idx1 : idx2, idx3) = J(idx1 : idx2, idx3) − (R̂11(s, 1 : n_U))′;
 end
end
```

- $D_3 = -Y_U \hat{R_{12}} Y_U$ is an $(n-n_U) \times n_U$ matrix. We introduce $D_{31} = -Y_U \hat{R_{12}}$ and $D_{32} = -\hat{R_{12}} Y_U$. With this notation, we have

$$D_3' = D_{31} Y_U' + Y_U' D_{32}, \tag{7.39}$$

First we consider $D_{31} Y_U'$ that is a $(n-n_U) \times n_U$ matrix with the general form

$$(D_{31} Y_U')_{(j,i)} = \sum_{l=1}^{n-n_U} (D_{31})_{(j,l)} (Y_U')_{(l,i)}$$

This contributes to the derivative with respect to $(Y_U)_{(r,s)}$ if $i = s$ with the term $(D_{31})_{(j,r)}$. Now if $l = n(N-1)$ and $h = Nn$, then

– $(D_{31}Y_U^{'})_{(j,i)}$ is at row position $l+i+(j-1)n_U$.
– $(Y_U)_{(r,i)}$ is at column position $h+r+(i-1)(n-n_U)$.

Therefore we update

$$J(l+i+(j-1)n_U,\ h+r+(i-1)(n-n_U)) := (D_{31})_{(j,r)},$$

whenever $1 \leq j \leq n-n_U$, $1 \leq i \leq n_U$ and $1 \leq r \leq n-n_U$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_3

for **j** $= 1 : n - n_U$
  for **i** $= 1 : n_U$
   idx1 $= l + i + (j-1)n_U$;
   idx2 $= h + 1 + (i-1)(n-n_U)$;
   idx3 $= h + n - n_U + (i-1)(n-n_U)$;
   J(idx1, idx2 : idx3) $= J$(idx1, idx2 : idx3) $+ D_{31}(j, 1 : n-n_U)$;
  end
 end

Now we consider $D_{32}Y_U^{'}$ that is a $(n-n_U) \times n_U$ matrix with general form

$$(Y_U^{'}D_{32})_{(j,i)} = -\sum_{l=1}^{n_U}(Y_U^{'})_{(j,l)}(D_{31})_{(l,i)}$$

This contributes to the derivative with respect to $(Y_U)_{(r,s)}$ if $r = j$ with the term $(D_{32})_{(s,i)}$. $(Y_U)_{(j,s)}$ is $(D_{32})_{(s,i)}$. Now if $l = n(N-1)$ and $h = Nn$, then

– $(Y_U^{'}D_{32})_{(j,i)}$ is at row position $l+i+(j-1)n_U$.
– $(Y_U)_{(j,s)}$ is at column position $h+j+(s-1)(n-n_U)$.

Therefore we update

$$J(l+i+(j-1)n_U,\ h+j+(s-1)(n-n_U) :=$$
$$J(l+i+(j-1)n_U,\ h+j+(s-1)(n-n_U)+(D_{32}(s,i))',$$

whenever $1 \le j \le n-n_U$, $1 \le i \le n_U$ and $1 \le s \le n_U$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_{32}

for **j** $= 1 : n - n_U$
 for **s** $= 1 : n_U$
 idx1 $= l + 1 + (j-1) * n_U$;
 idx2 $= l + n_U + (j-1) * n_U$;
 idx3 $= h + j + (s-1) * (n - n_U)$;
 J(idx1 : idx2, idx3) $= J$(idx1 : idx2, idx3) $+ (D_{32})(s, 1 : n_U))'$;
 end
end

- We compute the derivatives of $F(Y_U)$ in (7.32) with respect to the components of x_1. If $1 \le i \le n$, then the derivative of $Q^T(0)A(s)Q(0)$ w.r.t $x_{1,i}$ is given by $D_i = Q^T(0)hess(:,:,i)Q(0)$ where $hess$ is the Hessian of $f^{(J)}$. Then $D_{1i} = D_i(1 : n_U,\ 1 : n_U)$, $D_{2i} = D_i(1 : n_U,\ n_U + 1 : n)$, $D_{3i} = D_i(n_U + 1 : n,\ 1 : n_U)$ and $D_{4i} = D_i(n_U + 1 : n,\ n_U + 1 : n)$ are derivatives of $\hat{R_{11}}$, $\hat{R_{12}}$, E_{21} and $\hat{R_{22}}$, w.r.t $x_{1,i}$, respectively. Derivatives of $F(Y_U)$ w.r.t $x_{1,i}$ are hence given by

$$(F(Y_U))_{x_{1,i}} = D_{1i}Y_U - Y_U D_{2i} + D_{3i} - Y_U D_{4i} Y_U$$

All nonzero derivatives arise from the fact that the derivative of $(F(Y_U))_{(j,s)}$ w.r.t to $x_{1,i}$ is

$$(D_{1i})_{(j,:)}(Y_U)_{(:,s)} - (Y_U)_(j,:)(D_{2i})_{(:,s)} + (D_{3i})_{(j,s)} - (Y_U)_{(j,:)}(D_{4i}Y_U)_{(:,s)}$$

Now if $l = n(N-1)$, then

– $(F(Y_U))_{(j,s)}$ is at row position $l + s + (j-1)n_U$.
– $x_{1,i}$ is at column position i.

Therefore we update

$$J(l+s+(j-1)n_U,\ i) :=$$
$$(D_{1i})_{(j,:)}(Y_U)_{(:,s)} - (Y_U)_(j,:)(D_{2i})_{(:,s)} + (D_{3i})_{(j,s)} - (Y_U)_{(j,:)}(D_{4i}Y_U)_{(:,s)},$$

whenever where $j = 1, \ldots, n - n_U,\ \ s = 1, \ldots, n_U,\ i = 1, \ldots, n$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to $F(Y_U)$ w.r.t the components of x_1

```
for i = 1 : n
D = Q^T(0) * hess(:, :, i) * Q(0);
D1 = D(1 : n_U, 1 : n_U);
D2 = D(1 : n_U, n_U + 1 : n);
D3 = D(n_U + 1 : n, 1 : n_U);
D4 = D(n_U + 1 : n, n_U + 1 : n);
 for j = 1 : n - n_U
  for s = 1 : n_U
     idx = l + s + (j - 1) * n_U
     J(idx, i) = D4(j, :) * Y_U(:, s) - Y_U(j, :) * D1(:, s) + D3(j, s);
     for k = 1 : n_U
       J(idx, i) = J(idx, i) - Y_U(j, k) * (D2(k, :) * Y_U(:, s));
     end
  end
 end
end
```

- We now compute the derivatives of $F(Y_U)$ in (7.32) with respect to the control parameter, α_a. The derivative of $Q^T(0)A(s)Q(0)$ w.r.t the control parameter is given by $D = Q^T(0)hessp(:, :, \text{ActiveParams})Q(0)$ where $hessp$ is the Hessian of $f^{(J)}$ w.r.t α_a. Then $D_1 = D(1 : n_U,\ 1 : n_U)$,

$D_2 = D(1 : n_U,\ n_U + 1 : n)$, $D_3 = D(n_U + 1 : n,\ 1 : n_U)$ and $D_4 = D(n_U + 1 : n,\ n_U + 1 : n)$ are derivatives of $\hat{R_{11}}$, $\hat{R_{12}}$, E_{21} and $\hat{R_{22}}$, w.r.t α_a, respectively.

Derivatives of $F(Y_U)$ w.r.t the control parameter are hence given by

$$(F(Y_U))_{\alpha_a} = D_1 Y_U - Y_U D_2 + D_3 - Y_U D_4 Y_U$$

All nonzero derivatives arise from the fact that the derivative of $(F(Y_U))_{(j,s)}$ w.r.t to α_a is

$$(D_1)_{(j,:)}(Y_U)_{(:,s)} - (Y_U)_(j,:)(D_2)_{(:,s)} + (D_3)_{(j,s)} - (Y_U)_{(j,:)}(D_4 Y_U)_{(:,s)}$$

Now if $l = n(N-1)$, then

- $(F(Y_U))_{(j,s)}$ is at row position $l + s + (j-1)n_U$.
- α_a is at column position $k + 1$.

Therefore we update

$$J(l + s + (j-1)n_U,\ k+1) :=$$
$$(D_1)_{(j,:)}(Y_U)_{(:,s)} - (Y_U)_(j,:)(D_2)_{(:,s)} + (D_3)_{(j,s)} - (Y_U)_{(j,:)}(D_4 Y_U)_{(:,s)},$$

whenever where $j = 1, \ldots, n - n_U,\ \ s = 1, \ldots, n_U$

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to $F(Y_U)$ w.r.t the components of α_a

$D = Q^T(0) * hess(:,:,\text{ActiveParams}) * Q(0);$
$D_1 = D(1 : n_U,\ 1 : n_U);$
$D_2 = D(1 : n_U,\ n_U + 1 : n);$
$D_3 = D(n_U + 1 : n,\ 1 : n_U);$
$D_4 = D(n_U + 1 : n,\ n_U + 1 : n);$
for **j** $= 1 : n - n_U$
for **s** $= 1 : n_U$
idx $= l + s + (j - 1) * n_U;$
$J(\text{idx},\ k + 1) = D_4(j,:) * Y_U(:,s) - Y_U(j,:) * D_1(:,s) + D_3(j,s);$
for **k** $= 1 : n_U$
$J(\text{idx},\ k + 1) = J(\text{idx},\ i) - Y_U(j,k) * (D_2(k,:) * Y_U(:,s));$
end
end
end

- We now look at x_N and compute the nonzero entries of J corresponding to (7.35). For simplicity of the computations we divide (7.35) into 3 terms $D_1 = \hat{R}_{22}Y_S$, $D_2 = -Y_S\hat{R}_{11}$, $D_3 = Y_S\hat{R}_{12}Y_S$.

  We first consider D_1 whose derivatives with respect to the components of Y_S are written into a block matrix of size $((n - n_S) \times n_S) \times ((n - n_S) \times n_S)$. D_1 is an $(n - n_S) \times n_S$ matrix with general form $(D_1)_{(j,i)} = \sum_{l=1}^{n-n_S} (\hat{R}_{22})_{(j,l)}(Y_S)_{(l,i)},\ \ j = 1, \ldots, n - n_S,\ \ i = 1, \ldots, n_S$. Hence all nonzero derivatives arise from the fact that the derivative of $(D_1)_{(j,i)}$ with respect to $(Y_S)_{(s,i)}$ is $(\hat{R}_{22})_{(j,s)}$. Now if $l = n(N - 1) + n_U(n - n_U)$ and $h = Nn + n_U(n - n_U)$, then

  - $(D_1)_{(j,i)}$ is at row position $l + i + (j - 1)n_S$.
  - $Y_{(s,i)}$ is at column position $h + s + (i - 1)(n - n_S)$.

  Therefore we update

$$J(l + i + (j - 1)n_S, h + s + (i - 1)(n - n_S)) := (\hat{R}_{22})_{(j,s)},$$

whenever $1 \leq j \leq n - n_S$, $1 \leq i \leq n_S$ and $1 \leq s \leq n - n_S$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_1

```
for j = 1 : n − n_S
 for i = 1 : n_S
  idx1 = h + 1 + (i − 1) ∗ (n − n_S);
  idx2 = h + (n − n_S) + (i − 1) ∗ (n − n_S);
  J(l + i + (j − 1) ∗ n_S, idx1:idx2) = R̂_22(j, 1 : n − n_S);
 end
end
```

- D_2 is an $(n - n_S) \times n_U$ matrix with general form

$$(D_2)_{(j,i)} = -\sum_{l=1}^{n_S} (Y_S)_{(j,l)} (\hat{R}_{11})_{(l,i)}, \;\; j = 1, \ldots, n - n_S, \;\; i = 1, \ldots, n_S$$

Hence all nonzero derivatives arise from the fact that the derivative of $(D_2)_{(j,i)}$ with respect to $(Y_S)_{(j,s)}$ is $(\hat{R}_{11})_{(s,i)}$. Now if $l = n(N-1) + n_U(n - n_U)$ and $h = Nn + n_U(n - n_U)$, then

  - $(D_2)_{(j,i)}$ is at row position $l + i + (j-1)n_S$.
  - $(Y_U)_{(j,s)}$ is at column position $h + j + (s-1)(n - n_S)$.

Therefore we update

$$\begin{aligned} &J(l + i + (j-1)n_S, h + j + (s-1)(n - n_S)) := \\ &J(l + i + (j-1)n_S, h + j + (s-1)(n - n_S)) - (\hat{R}_{11})_{(s,i)}, \end{aligned}$$

whenever $1 \leq j \leq n - n_S$, $1 \leq i \leq n_S$ and $1 \leq s \leq n_S$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_2

for **j** $= 1 : n - n_S$
 for **s** $= 1 : n_S$
  idx1 $= l + 1 + (j-1) * n_S$;
  idx2 $= l + n_U + (j-1) * n_S$;
  idx3 $= h + j + (s-1) * (n - n_S)$;
  J(idx1 : idx2, idx3) $= J$(idx1 : idx2, idx3) $- (\hat{R}_{11}(s,\ 1 : n_S))'$;
 end
end

- $D_3 = -Y_S \hat{R_{12}} Y_S$ is an $(n-n_S)\times n_S$ matrix. We introduce $D_{31} = -Y_S \hat{R_{12}}$ and $D_{32} = -\hat{R_{12}} Y_S$. With this notation, we have

$$D_3^{'} = D_{31} Y_S^{'} + Y_S^{'} D_{32}, \tag{7.40}$$

First we consider $D_{31} Y_S^{'}$ that is a $(n - n_S) \times n_S$ matrix with the general form

$$(D_{31} Y_S^{'})_{(j,i)} = \sum_{l=1}^{n-n_S} (D_{31})_{(j,l)} (Y_S^{'})_{(l,i)}$$

This contributes to the derivative with respect to $(Y_S)_{(r,s)}$ if $i = s$ with the term $(D_{31})_{(j,r)}$. Now if $l = n(N-1) + n_U(n-n_U)$ and $h = Nn + n_U(n-n_U)$, then

  - $(D_{31} Y_S^{'})_{(j,i)}$ is at row position $l + i + (j-1)n_S$.
  - $(Y_S)_{(r,i)}$ is at column position $h + r + (i-1)(n-n_S)$.

Therefore we update

$$J(l + i + (j-1)n_S,\ h + r + (i-1)(n-n_S)) := (D_{31})_{(j,r)},$$

whenever $1 \le j \le n - n_S$, $1 \le i \le n_S$ and $1 \le r \le n - n_S$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_3

for j $= 1 : n - n_S$
for i $= 1 : n_S$
idx1 $= l + i + (j-1)n_S$;
idx2 $= h + 1 + (i-1)(n - n_S)$;
idx3 $= h + n - n_S + (i-1)(n - n_S)$;
J(idx1, idx2 : idx3) $= J$(idx1, idx2 : idx3) $+ D_{31}(j, 1 : n - n_S)$;
end
end

Now we consider $D_{32}Y_S^{'}$ that is a $(n - n_S) \times n_S$ matrix with general form

$$(Y_S^{'}D_{32})_{(j,i)} = -\sum_{l=1}^{n_S}(Y_S^{'})_{(j,l)}(D_{31})_{(l,i)}$$

This contributes to the derivative with respect to $(Y_S)_{(r,s)}$ if $r = j$ with the term $(D_{32})_{(s,i)}$. $(Y_S)_{(j,s)}$ is $(D_{32})_{(s,i)}$. Now if $l = n(N-1) + n_U(n - n_U)$ and $h = Nn + n_U(n - n_U)$, then

- $(Y_S^{'}D_{32})_{(j,i)}$ is at row position $l + i + (j-1)n_S$.
- $(Y_S)_{(j,s)}$ is at column position $h + j + (s-1)(n - n_S)$.

Therefore we update

$$J(l + i + (j-1)n_S,\ h + j + (s-1)(n - n_S) :=$$
$$J(l + i + (j-1)n_S,\ h + j + (s-1)(n - n_S) + (D_{32}(s,i))',$$

whenever $1 \le j \le n - n_S$, $1 \le i \le n_S$ and $1 \le s \le n_S$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to D_{32}

for **j** $= 1 : n - n_S$
 for **s** $= 1 : n_S$
  idx1 $= l + 1 + (j-1) * n_S$;
  idx2 $= l + n_S + (j-1) * n_S$;
  idx3 $= h + j + (s-1) * (n - n_S)$;
  $J(\text{idx1} : \text{idx2},\ \text{idx3}) = J(\text{idx1} : \text{idx2},\ \text{idx3}) + (D_{32})(s, 1 : n_S))'$;
 end
end

- We now compute the derivatives of $F(Y_S)$ in (7.35) with respect to the components of x_N. If $1 \leq i \leq n$, then the derivative of $Q^T(0)A(s)Q(0)$ w.r.t $x_{(N,i)}$ is given by $D_i = Q^T(0)hess(:,:,i)Q(0)$ where $hess$ is the Hessian of $f^{(J)}$. Then $D_{1i} = D_i(1 : n_U,\ 1 : n_U)$, $D_{2i} = D_i(1 : n_U,\ n_U + 1 : n)$, $D_{3i} = D_i(n_S+1 : n,\ 1 : n_S)$ and $D_{4i} = D_i(n_S+1 : n,\ n_S+1 : n)$ are derivatives of $\hat{R_{11}}$, $\hat{R_{12}}$, E_{21} and $\hat{R_{22}}$, w.r.t $x_{(N,i)}$, respectively.

  Derivatives of $F(Y_S)$ w.r.t $x_{(N,i)}$ are hence given by

$$(F(Y_S))_{x_{1,i}} = D_{1i}Y_S - Y_S D_{2i} + D_{3i} - Y_S D_{4i} Y_S$$

  All nonzero derivatives arise from the fact that the derivative of $(F(Y_S))_{(j,s)}$ w.r.t to $x_{(N,i)}$ is

$$(D_{1i})_{(j,:)}(Y_S)_{(:,s)} - (Y_S)_(j,:)(D_{2i})_{(:,s)} + (D_{3i})_{(j,s)} - (Y_S)_{(j,:)}(D_{4i}Y_S)_{(:,s)}$$

  Now if $l = n(N-1) + n_U(n - n_U)$, then

  – $(F(Y_S))_{(j,s)}$ is at row position $l + s + (j-1)n_S$.
  – $x(N,i)$ is at column position $N(N-1) + i$.

  Therefore we update

$$J(l + s + (j-1)n_S,\ N * (N-1) + i) :=$$
$$(D_{1i})_{(j,:)}(Y_S)_{(:,s)} - (Y_S)_(j,:)(D_{2i})_{(:,s)} + (D_{3i})_{(j,s)} - (Y_S)_{(j,:)}(D_{4i}Y_S)_{(:,s)},$$

whenever where $j = 1, \ldots, n - n_S,\ \ s = 1, \ldots, n_S,\ i = 1, \ldots, n$.

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to $F(Y_S)$ w.r.t the components of x_N

for i $= 1 : n$
$D = Q^T(0) * hess(:, :, i) * Q(0);$
$D_1 = D(1 : n_S,\ 1 : n_S);$
$D_2 = D(1 : n_S,\ n_S + 1 : n);$
$D_3 = D(n_S + 1 : n,\ 1 : n_S);$
$D_4 = D(n_S + 1 : n,\ n_S + 1 : n);$
for j $= 1 : n - n_S$
for s $= 1 : n_S$
idx $= l + s + (j - 1) * n_S;$
$J(\text{idx},\ N * (N - 1) + i) = D_4(j, :) * Y_S(:, s) - Y_S(j, :) * D_1(:, s) + D_3(j, s);$
for k $= 1 : n_U$
$J(\text{idx},\ N * (N - 1) + i) = J(\text{idx},\ i) - Y_S(j, k) * (D_2(k, :) * Y_S(:, s));$
end
end
end
end

- We now compute the derivatives of $F(Y_S)$ in (7.35) with respect to the control parameter, α_a. The derivative of $Q^T(0)A(s)Q(0)$ w.r.t the control parameter is given by $D = Q^T(0)hessp(:, :, \text{ActiveParams})Q(0)$ where $hessp$ is the Hessian of $f^{(J)}$ w.r.t α_a. Then $D_1 = D(1 : n_S,\ 1 : n_S)$, $D_2 = D(1 : n_S,\ n_S + 1 : n)$, $D_3 = D(n_S + 1 : n,\ 1 : n_S)$ and $D_4 = D(n_S + 1 : n,\ n_S + 1 : n)$ are derivatives of $\hat{R_{11}}$, $\hat{R_{12}}$, E_{21} and $\hat{R_{22}}$, w.r.t α_a, respectively.

  Derivatives of $F(Y_S)$ w.r.t the control parameter are hence given by

$$(F(Y_S))_{\alpha_a} = D_1 Y_S - Y_S D_2 + D_3 - Y_S D_4 Y_S$$

  All nonzero derivatives arise from the fact that the derivative of $(F(Y_S))_{(j,s)}$ w.r.t to α_a is

$$(D_1)_{(j,:)}(Y_S)_{(:,s)} - (Y_S)_(j, :)(D_2)_{(:,s)} + (D_3)_{(j,s)} - (Y_S)_{(j,:)}(D_4 Y_S)_{(:,s)}$$

Now if $l = n(N-1)$, then

- $(F(Y_S))_{(j,s)}$ is at row position $l + s + (j-1)n_S$.
- α_a is at column position $k+1$.

Therefore we update

$$J(l+s+(j-1)n_S,\ k+1) :=$$
$$(D_1)_{(j,:)}(Y_S)_{(:,s)} - (Y_S)_(j,:)(D_2)_{(:,s)} + (D_3)_{(j,s)} - (Y_S)_{(j,:)}(D_4 Y_S)_{(:,s)},$$

whenever where $j = 1, \ldots, n - n_S,\ \ s = 1, \ldots, n_S$

The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J corresponding to $F(Y_S)$ w.r.t the components of α_a

```
D = Q^T(0) * hess(:,:,ActiveParams) * Q(0);
D1 = D(1 : n_S, 1 : n_S);
D2 = D(1 : n_S, n_S + 1 : n);
D3 = D(n_S + 1 : n, 1 : n_S);
D4 = D(n_S + 1 : n, n_S + 1 : n);
 for j = 1 : n - n_S
  for s = 1 : n_S
    idx = l + s + (j - 1) * n_S;
    J(idx, k + 1) = D4(j,:) * Y_S(:,s) - Y_S(j,:) * D1(:,s) + D3(j,s);
    for k = 1 : n_S
      J(idx, k + 1) = J(idx, i) - Y_S(j,k) * (D2(k,:) * Y_S(:,s));
    end
  end
end
```

- We consider conditions (7.7) which consists of $(n - n_U)$ equations of the form

$$B_i = \sum_{k=1}^{n} (x_2 - x_1)_{(1,k)} (q^U_{(n_U+i)})_k,\ \ i = 1, \ldots, n - n_U = 0$$

– All nonzero derivatives are of the form B_i w.r.t $x_{1,s}$ is $-(q_{(n_U+i)})_s$. Now if $l = n(N-1) + n_U(n-n_U) + n_S(n-n_S)$, then

  * $B_{(1,i)}$ is at row position $l+i$.
  * $x_{1,s}$ is at column position s.

  Therefore we update

$$J(l+i, s) := -(q^U_{(n_U+i)})_{(s)},$$

  whenever $1 \le i \le n - n_U$ and $1 \le s \le n$.

  The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J

```
for i = 1 : n − n_U
    J(l + i, 1 : n) = −(q^U_(n_U+i))′;
end
```

– All nonzero derivatives are of the form B_i w.r.t $x(2,s)$ is $(q^U_{(n_U+i)})_{(s)}$. Now if $l = n(N-1) + n_U(n-n_U) + n_S(n-n_S)$, then

  * B_i is at row position $l+i$.
  * $x_{2,s}$ is at column position $n+s$.

  Therefore we update

$$J(l+i, n+s) := (q^U_{(n_U+i)})(s),$$

  whenever $1 \le i \le n - n_U$ and $1 \le s \le n$.

  The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J

```
for i = 1 : n − n_U
    J(l + i, n + 1 : 2 ∗ n) = (q^U_(n_U+i))′;
end
```

– We now compute derivatives of (7.7) w.r.t the components of Q_U. The equations (7.7) have the following form

$$B_i = (x_2 - x_1)^T q^U_{n_U+i} = 0, \text{for } \; i = 1, \ldots, n - n_U \qquad (7.41)$$

Here $q^U_{n_U+i}$ are precisely the columns of $Q_U(0) \begin{bmatrix} -Y_U^T \\ I \end{bmatrix}$. Therefore it is best to introduce the new vector $H = (x_2 - x_1)^T Q(0)$. With this notation the i-th equation becomes

$$-\sum_{j=1}^{n_U} H_j (Y_U^T)_{ji} + \text{terms without components of } Y_U = 0$$

Or

$$-\sum_{j=1}^{n_U} H_j (Y_U^T)_{ji} + \ldots$$

This means that the i-th equation has derivatives with respect to $(Y_U)_{ji}$ equal to $-H_j$ $(i = 1 \ldots, n - n_U, \;\; j = 1, \ldots, n_U)$.
Let $l = n(N-1) + n_U(n - n_U) + n_S(n - n_S), \; h = nN$. Then the i-th equation is at column position $l + i$. The variable $(Y_U)_{ji}$ is at column position $h + j + (n - n_U)(i - 1)$. So we have to set

$$J(l + i, \;\; h + j + (n - n_U)(i - 1)) := -H_j, \;\; \text{for the relevant } i, j.$$

Or equivalently, in MATLAB code we have

for **i** $= 1 : n - n_U$
  idx1 $= h + 1 + (n - n_U) * (i - 1)$;
  idx2 $= h + hetds.n_U + (n - n_U) * (i - 1)$;
    $J(l + i, \;\; \text{idx1} : \text{idx2}) = -(H)'$;
end

- We consider conditions (7.8) which consists of $(n - n_S)$ equations of the form

$$B_i = \sum_{k=1}^{n} (x_{N-1} - x_N)_{(1,k)} (q^S_{(n_S+i)})_{(k)} = 0, \;\; i = 1, \ldots, n - n_S.$$

– All nonzero derivatives are of the form B_i w.r.t $x(N-1,s)$ is $(q^S_{(n_S+i)})_s$. Now if $l = n(N-1) + n_U(n-n_U) + n_S(n-n_S) + n - n_U$, then

  * B_i is at row position $l+i$.
  * $x_{N-1,s}$ is at column position $n(N-2)+s$.

  Therefore we update

$$J(l+i, n(N-2)+s) := (q^S_{(n_S+i)})(s),$$

  whenever $1 \le i \le n - n_S$ and $1 \le s \le n$.
  The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J

```
for i = 1 : n − n_S
    J(l + i, 1 : n) = (q^S_(n_U+i))';
end
```

– All nonzero derivatives are of the form B_i w.r.t $x(N,s)$ is $-(q^S_{(n_S+i)})_{(s)}$. Now if $l = n(N-1) + n_U(n-n_U) + n_S(n-n_S)$, then

  * B_i is at row position $l+i$.
  * $x_{2,s}$ is at column position $n(N-1)+s$.

  Therefore we update

$$J(l+i, n(N-1)+s) := (q^S_{(n_S+i)})(s),$$

  whenever $1 \le i \le n - n_S$ and $1 \le s \le n$.
  The following MATLAB command lines compute the nonzero entries of the Jacobian matrix J

```
for i = 1 : n − n_S
    J(l + i, n(N − 1) + 1 : N ∗ n) = (q^S_(n_S+i))';
end
```

– We now compute derivatives of (7.8) w.r.t the components of Q_S. The equations (7.8) have the following form

$$B_i = (x_{N-1} - x_N)^T q^S_{n_S+i} = 0, \text{for } i = 1, \ldots, n - n_S \qquad (7.42)$$

Here $q^S_{n_S+i}$ are precisely the columns of $Q_S(0) \begin{bmatrix} -Y_S^T \\ I \end{bmatrix}$. Therefore it is best to introduce the new vector $H = (x_{N-1} - x_N)^T Q(0)$. With this notation the i-th equation becomes

$$-\sum_{j=1}^{n_S} H_j (Y_S^T)_{ji} + \text{terms without components of } Y_S = 0$$

Or

$$-\sum_{j=1}^{n_S} H_j (Y_S^T)_{ji} + \ldots$$

This means that the i-th equation has derivatives with respect to $(Y^S)_{ji}$ equal to $-H_j$ $(i = 1 \ldots, n - n_S,\ \ j = 1, \ldots, n_S)$.
Let $l = n(N-1) + n_U(n - n_U) + n_S(n - n_S) + n - n_U,\ h = nN + n_U(n - n_U)$. Then the i-th equation is at column position $l + i$. The variable $(Y^S)_{ji}$ is at column position $h + j + (n - n_S)(i-1)$. So we have to set

$$J(l + i,\ \ h + j + (n - n_S)(i-1)) := -H_j, \ \text{ for the relevant } i, j.$$

In MATLAB code we have

```
for i = 1 : n − n_S
  idx1 = h + 1 + (n − n_S) ∗ (i − 1);
  idx2 = h + n_S + (n − n_S) ∗ (i − 1);
    J(l + i, idx1 : idx2) = −(H)′;
end
```

7.2 Continuation of homoclinic connections

Assume that the eigenvalues of $(f^{(J)}(x_1,\alpha)_x$ are ordered as follows:

$$|\lambda_1| \leq \ldots \leq |\lambda_k| < 1 < |\lambda_{k+1}| \leq \ldots \leq |\lambda_n|$$

The procedure to continue a homoclinic connection to x_1 is similar to the procedure used in §7.1. The algorithm now requires the evaluation of two projections associated with the eigenspaces of $(f^{(J)}(x_1,\alpha)_x$. These projections are constructed using the real Schur factorizations.

$$(f^{(J)}(x_1,\alpha))_x = Q_1 R_1 Q_1^T, \;\; (f^{(J)}(x_1,\alpha))_x = Q_2 R_2 Q_2^T$$

where Q_1, Q_2, R_1 and R_2 are $n \times n$-matrices.

The first factorization has been chosen so that the first k columns $q_1^S, \ldots, q_k^S$ of Q_1 form an orthonormal basis of the right invariant subspace S_1 of $(f^{(J)}(x_1,\alpha)_x$, corresponding to $\lambda_1, \ldots, \lambda_k$ and the remaining $n-k$ columns $q_{k+1}^U, \ldots, q_n^U$ of Q_1 form an orthonormal basis of the orthogonal complement $S_1^\perp$. Similary the first $l = n-k$ columns $q_1^U, \ldots, q_l^U$ of Q_2 form an orthonormal basis of the right invariant subspace U_1 of $(f^{(J)}(x_1,\alpha))_x$, corresponding to $\lambda_{k+1}, \ldots, \lambda_n$ and the remaining $n-l = k$ columns $q_{l+1}^U, \ldots, q_n^U$ of Q_2 form an orthonormal basis of the orthogonal complement $U_1^\perp$.

The problem of *homoclinic* connection is to find a connection $\{x_m\}_{m=1,\ldots,N}$ with

- Stationary state condition

$$f^{(J)}(x_1,\alpha) - x_1 = 0, \tag{7.43}$$

- The iteration conditions

$$f^{(J)}(x_m,\alpha) - x_{m+1} = 0, \;\; m = 2,3,\ldots,N-2, \tag{7.44}$$

- The left boundary conditions

$$\langle (x_2 - x_1).q_{k+i}^U(\alpha)\rangle = 0, \;\; i = 1,\ldots,n-k \tag{7.45}$$

- The right boundary conditions

$$\langle (x_{N-1} - x_1), q^S_{l+i}(\alpha) \rangle = 0, \;\; i = 1, \ldots, n-l \tag{7.46}$$

A regular zero of a system of equations (7.43), (7.44), (7.45) and (7.46) corresponds to a transversal homoclinic orbit. Thus, a zero for this system can be continued in one parameter.

In the continuation process the conditions in (7.45) and (7.46) imply that we need to access the stable and unstable eigenspaces of the map (7.1) at the fixed points x_1 at each step of the continuation. Using the same procedure, as in the computation of the unstable subspace for x_1, we can obtain relations analogous to (7.13), (7.26) and (7.32) to compute the stable invariant subspace and its orthogonal complement for x_N. Solving (7.32) for Y_U of size $(n-k) \times k$, enables us to compute the stable invariant subspace for x_1 and its complement. If $Q_U(0)$ is the orthogonal matrix from the initial or adaptation step, related to the unstable invariant subspace, then a basis for the orthogonal subspace in the new step $Q_U^{\perp}$, is given by

$$Q_U(s) = Q_U(0) \begin{bmatrix} I \\ Y_U \end{bmatrix}. \tag{7.47}$$

A basis for the orthogonal complement of the subspace in the new step $Q_U^{\perp}$, is given by

$$Q_U^{\perp}(s) = Q_U(0) \begin{bmatrix} -Y_U^T \\ I \end{bmatrix}. \tag{7.48}$$

As in §7.1, we can also continue the stable eigenspace and its orthogonal complement.

7.3 Invariant manifolds of planar maps

Invariant manifolds give information about the global structure of phase space. For example, a codimension 1 manifold may separate several basins of attraction. Invariant manifolds are also used to simplify dynamical systems. The phase portrait near the manifold may be trivial, so restricting the dynamical system to the manifold effectively reduces the dimension of the system.

Our main motivation for computing stable and unstable manifolds of a saddle point is the role that they play in the computation of connecting orbits. Intersections of stable and unstable manifolds may form homoclinic or heteroclinic tangles. Stable and unstable manifolds are global objects that cannot normally be found analytically and, hence, must be computed numerically. These manifolds must be grown from local knowledge, for example from linear information near a fixed point.

We concentrate here on the simplest case that these manifolds are one-dimensional. Most algorithms use the idea of computing the manifold by starting from a local approximation near the saddle point. The map that arises in a particular application does not necessarily have an explicit inverse or may not even be invertible, meaning that there may be several branches of inverses. Consequently, the standard algorithms requiring the inverse cannot be used to compute stable manifolds of saddle points in this case.

First we present an algorithm to compute the stable manifold of a saddle point of a planar map, without requiring any knowledge of its inverse map, either explicitly or approximately. In particular, the algorithm can also be used in the case where the map is noninvertible, so that multiple pre-images may exist.

We recall some definitions, mostly to fix the notation. We consider (7.1) when $n = 2$ and assume that f has a fixed point $x_0 = f^{(J)}(x_0)$ and that f is differentiable in a neighborhood of x_0, but may not have a unique inverse. The fixed point x_0 of f is a saddle if the Jacobian matrix $D(f^{(J)})(x_0)$ has one stable eigenvalue λ_s and one unstable eigenvalue λ_u. The stable manifold theorem [79] guarantees that there exist local stable and unstable manifolds $W^s_{loc}(x_0)$ and $W^u_{loc}(x_0)$ tangent at x_0 to the stable and unstable eigenspaces $E^s(x_0)$ and $E^u(x_0)$, respectively. The stable manifold $W^s(x_0)$ of x_0 is defined as the set of points that converge to x_0 under forward iteration of f,

$$W^s(x_0) = \left\{ x \in \mathbb{R}^2 : f^{(J)}(x) \to x_0 \text{ as } J \to \infty \right\}. \tag{7.49}$$

Similarly, the unstable manifold $W^u(x_0)$ of x_0 consists of points that converge to x_0 under backward iteration of the map f. In terms of forward iterates, this is

defined as

$$W^u(x_0) = \left\{ x \in \mathbb{R}^2 : \exists \{q_k\}_{k=0}^{\infty}, q_0 = x \text{ and } f^{(J)}(q_{k+1}) = q_k, \text{ and } \lim_{k\to\infty} q_k = x_0 \right\}. \tag{7.50}$$

The global stable manifold $W^s(x_0)$ is similarly defined as the union of the successive pre-images of $W^s_{loc}(x_0)$. However, if multiple inverses exist, then all pre-images, even if disjoint from the main branch, are part of the stable manifold. Hence the stable manifold may or may not be simply connected in phase space.

7.3.1 Computing a stable manifold

To compute the one-dimensional stable manifold of a planar map at a saddle point, we use the algorithm described in [32]. We briefly explain the search circle (SC) algorithm. SC algorithm uses the idea of growing a one-dimensional manifold in steps by adding new points according to the local curvature properties of the manifold and finds a new point close to the last computed point that maps under f to a piece of the manifold that was already computed. SC produces a piecewise linear approximation of $W^s(x_0)$ by computing an ordered list of points $M = \{p_0, p_1, \ldots, p_n\}$ at varying distance from each other. The first point p_1 is taken a small distance $\delta > 0$ from $p_0 = x_0$ along $E^s(x_0)$. The distance between consecutive points is adjusted according to the curvature of the manifold. To ensure an acceptable resolution of the curve according to pre-specified accuracy parameters, we monitor α_k, the angle between p_{k-1}, p_k and p_{k+1}, and the product $\alpha_k \Delta_k$, where $\Delta_k = \|p_{k+1} - p_k\|$. The α_k is approximated by

$$\alpha_k = 2sin^{-1}\left(\frac{\|\bar{p} - p_{k-1}\|}{2\|p_k - p_{k-1}\|}\right) \approx \frac{\|\bar{p} - p_{k-1}\|}{\|p_k - p_{k-1}\|} \tag{7.51}$$

where

$$\bar{p} = p_k + \frac{\|p_k - p_{k-1}\|}{\|p_k - p_{k+1}\|}(p_k - p_{k+1}) \tag{7.52}$$

is the point on the line through p_k and p_{k+1} that lies at the same distance from p_k as p_{k-1}. We check the conditions

$$\alpha_k < \alpha_{max} \tag{7.53}$$

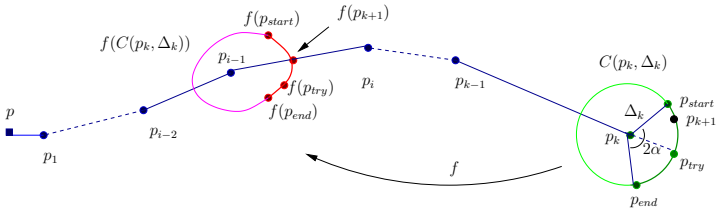


Figure 7.1: A graphical illustration of the SC algorithm. A new point P_{k+1} is found on the circle $C(p_k, \Delta_K)$ centered at p_k with radius Δ_k such that $f(p_{k+1})$ lies on a previously computed part of $W^S_{x_0}$.

$$\alpha_k \Delta_k < (\alpha\Delta)_{max} \tag{7.54}$$

Condition (7.53) ensures that the resolution of the curve is maintained and condition (7.54) controls the local interpolation error. The new point p_{k+1} is accepted if it satisfies the above criteria. If one of the criteria is not satisfied, then we replace Δ_k by $\frac{1}{2}\Delta_k$ and repeat the procedure to find a new candidate for p_{k+1}. We set $\Delta_{k+1} = 2\Delta_k$ if both $\alpha_k > \alpha_{min}$ and $\alpha_k\Delta_k > (\alpha\Delta)_{min}$ for a user-specified choice of parameters α_{min} and $(\alpha\Delta)_{min}$. This ensures that the number of points used to approximate the manifold is in some sense optimized for the required accuracy constraints, see [59] for more details.

A graphical illustration of the SC algorithm is given in Figure 7.1.

A pseudo-code representation, as described in [32], of how the branch is grown, is given below:

Grow-Manifold

(Fixed point: p_0, first point along $E^s(p_0)$: p_1, target arclength: A)

$M = \{p_0, p_1\}$;

$p_{left} = p_0$;

$p_{right} = p_1$;

$arclength = \|p_1 - p_0\|$;

while ($arclength < A$)

  p_k and p_{k-1} last and next to the last point in M;

  $(p_{candidate}, p_{i-1}, p_i) =$ **Search_Circle**(M, p_{left}, p_{right});

($[p_{i-1},\ p_i]$ is the interval in which $f(p_{candidate})$ lies)
$\alpha_k = \angle(p_{k-1}, p_k, p_{candidate})$
if $((\alpha_k < \alpha_{max}\ \ and\ \ \Delta_k\alpha_k < (\Delta\alpha)_{max})$ or $\Delta_k < \Delta_{min})$;
(Accept point)
$P_{k+1} = p_{candidate}$;
append P_{k+1} to M
$arclength = arclenght + \Delta_k$;
$p_{left} = p_{i-1},\ \ p_{right} = p_i$;
if $((\alpha_k < \alpha_{min}\ \ and\ \ \Delta_k\alpha_k < (\Delta\alpha)_{min}))$
(Increase Δ_k for the next step)
$\Delta_k = 2\Delta_k$;
end if
else
(Accuracy conditions not satisfied. Reject point and decrease Δ_k.)
end if
end while
return M;
end

The **Search_Circle** algorithm is given in pseudo code, as described in [32]:

Search_Circle (M, p_{left}, p_{right})
do
$(p_{candidate}, \tau) =$ **Find_Point_On_Line**(p_{left}, p_{right});
(If $\tau < 0$ or $\tau > 1$, point is on line, but not on segment).
if ($\tau < 0$) (move backward)
$(p_{left}, p_{right}) = (p_{left-1}, p_{left})$;
else if ($\tau > 0$) (move forward)
$(p_{left}, p_{right}) = (p_{right}, p_{right+1})$;
end if
while $(\tau \notin [0,\ 1])$
return $(p_{candidate}, p_{left}, p_{right})$;
end.

Find_Point_On_Line (p_{left}, p_{right})
$L(\tau) = line[p_{left}, p_{right}] = \{(1-\tau)p_{left} + \tau p_{right} | \tau \in R\}$;
$\theta_{start} = -\alpha_{max}; \theta_{end} = \alpha_{max}$;

p_{start} = point on circle at angle θ_{start};
p_{end} = point on circle at angle θ_{end};
$\bar{p}$ = point on circle at angle 0;
$\vec{V_{start}} = f(p_{start}) - p_{right}$;
$\vec{V_{end}} = f(p_{end}) - p_{right}$;
$\vec{W}$ = normal vector to $p_{left} - p_{right}$;
if $(\langle \vec{V_{start}}, \vec{W} \rangle * \langle \vec{V_{end}}, \vec{W} \rangle) > 0$
($f(p_{start})$ and $f(p_{end})$ do not lie on opposite sides of L.)
Increase α_{max}
end if
do
(Bisection to find point on $L(\tau)$)
$\theta_{try} = \frac{(\theta_{start} + \theta_{end})}{2}$;
p_{try} = point on circle at angle θ_{try} from $\bar{p} - p_k$;
$\vec{V} = f(p_{try}) - p_{right}$;
if$(\langle \vec{V_{end}}, \vec{W} \rangle * \langle \vec{V}, \vec{W} \rangle) > 0$
($f(p_{try})$ is on same side as and $f(p_{end})$.)
$\theta_{end} = \theta_{try}$;
else
($f(p_{try})$ is on same side as and $f(p_{endstart})$.)
$\theta_{start} = \theta_{try}$;
end if
while $(|\langle \vec{V_{end}}, \vec{W} \rangle| < \epsilon_B)$
Normal distance between L and $f(p_{try}) < \epsilon_B$,
Accept as candidate.
return (p_{try}, τ);
end.

7.3.2 Computing an unstable manifold

We use the algorithm for computing the global one-dimensional unstable manifold of a saddle point of a map as described in [32]. To keep the exposition simple, we consider a planar diffeomorphism and suppose f is orientation preserving, otherwise we consider its second iterate. Let x_0 be a saddle point of f. The

unstable manifold of x_0 is defined as

$$W^u(x_0) = \left\{ x \in R^2 : f^{(-J)}(x) \to x_0 \text{ as } J \to \infty \right\}. \tag{7.55}$$

Note that, since f is a diffeomorphism, the stable manifold $W^s(x_0)$ is simply the unstable manifold of f^{-1} at x_0. The unstable manifold theorem [79] guarantees the existence of the local unstable manifold

$$W^u_{loc}(x_0) = \left\{ x \in W^u(x_0) : f^{(-n)}(x) \in U \text{ for all } n \in N \right\}. \tag{7.56}$$

in a suitable neighborhood U of x_0. Furthermore, it states that $W^u_{loc}(x_0)$ is tangent to the unstable eigenspace $W^u(x_0)$ of λ_u.
Similar to the algorithm that is used for computing the stable manifold, the idea is to grow the manifold independently of the dynamics in steps as a list of ordered points. At each step a new point is added at a prescribed distance Δ_k from the last point. New points are found as f-images of suitable points from the part we already computed. The algorithm starts with a linear approximation of the local manifold and grows the manifold up to a prespecified arclength l with a speed depending on the curvature of the manifold.

We now briefly describe a single step of the algorithm and suppose that the manifold $M = \{p_0, p_1, \ldots, p_k\}$ is already computed, where $p_0 = x_0$ and the point p_1 is at a small distance δ from x_0 in the unstable eigenspace $E^u(x_0)$. The next point p_{k+1} should have the property that the line segment $[p_k, p_{k+1}]$ accurately approximates $E^u(x_0)$. In order to achieve a good approximation, the distance between p_k and p_{k+1}, Δ_k, must be adjusted from step to step according to the curvature of the manifold.

We want to find p_{k+1} in a small annulus around the circle at p_k with radius Δ_k. To this end, we search in $W^u_{loc}(x_0)$ from the line L that is mapped by f to a curve which intersects the circle with center p_k and radius Δ_k. We start with the line segment in $W^u_{loc}(x_0)$ that contains the preimage of p_k and move linearly through $W^u_{loc}(x_0)$. Onece L is found, we use bisection to find a point $q \in L$ such that

$$(1 - \epsilon)\Delta_k < \|f(q) - p_k\| < (1 + \epsilon)\Delta_k$$

The point $p_{k+1} = f(q)$ is a candidate for the next point in M, see Figure 7.2. If Δ_k is acceptable then $p_{k+1} = f(q)$ is added to M, $[p_k, p_{k+1}]$ is added to $W^u_{loc}(x_0)$,

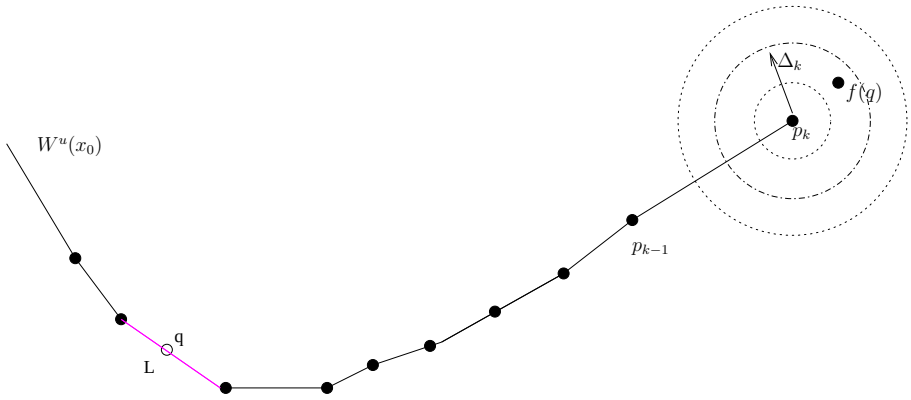


Figure 7.2: The next point $p_{k+1} = f(p_k)$ is chosen at distance Δ_k from p_k.

and step is completed. However, if Δ_k was too large then we reject $f(q)$, half the estimate Δ_k, and repeat the procedure. This algorithm is presented in pseudo-code, as described in [59], as follows:

Globalized1D

Input:	$f, x_0, v,$	$f(x_0) = x_0$, $E^u(x_0)$=span(v)
	$\delta,$	initial distance from x_0
	$\Delta,$	first estimate for Δ_k
	$\alpha_{min}, \alpha_{max}, (\Delta\alpha)_{min}, (\Delta\alpha)_{max}$	tolerances for α_k and $\Delta_k\alpha_k$
	l_{arc}	total arclength to be computed
output	$M,$	list of points

Begin

Add (M, x_0); $p_k = x_0 + \delta\nu$, Add (M, p_k);
$arclength = \delta$; $\Delta_k = \Delta$;
while $arclength < l_{arc}$ do
while $\|f(p_{can}) - p_k\| < \Delta_k$ do
L contains the preimage of p_k
$q = \mathbf{BISECT}(L, \Delta_k)$;
now $(1 - \epsilon)\Delta_k < \|f(q) - p_k\| < (1 + \epsilon)\Delta_k$

$p_{k+1} = f(q)$; $\alpha_k = \angle p_{k-1}, p_k, p_{candidate}$;
if $(\alpha_k < \alpha_{max})$ and $(\Delta_k \alpha_k < (\Delta\alpha)_{max})$ then
if $(\alpha_k < \alpha_{min})$ and $(\Delta_k \alpha_k < (\Delta\alpha)_{min})$ then
$\Delta_k = 2\Delta_k$;
a$rclength = $ a$rclength + \|p_{k+1} - p_k\|$
Add (M, p_{k+1});
else
$\Delta_k = \frac{1}{2}\Delta_k$;
end

7.4 Continuation of heteroclinic and homoclinic tangencies

Let $F(X, \alpha) = 0$ be the defining system of the heteroclininc connection, then a heteroclinic tangency satisfies the following conditions:

$$\begin{cases} F(X,\alpha) &= 0, \\ det(F_X(X,\alpha)) &= 0, \end{cases} \tag{7.57}$$

which is a system of $K_1 = n(N-1)+n_U(n-n_U)+n_S(n-n_S)+2n-n_U-n_S+1$ equations in a $K_2 = nN + n_U(n-n_U) + n_S(n-n_S) + \#ap$-dimensional space with coordinates (X, α). We recall that $X = (x_1, \ldots, x_N, Y_U, Y_S, ap)$.
If $n_U + n_S = n$ and $\#ap = 2$, then (7.57) defines a continuation problem. This system is natural from of a theoretical perspective but may lead to numerical scaling problems. If the Jacobian has eigenvalues of large magnitude, then these eigenvalues contribute to the determinant (which is the product of all eigenvalues) and may make it difficult to satisfy the defining equations to a desired tolerance. The larger the system, the worse this problem becomes. Thus we seek alternate defining equations that avoid calculation of the determinant. Bordered matrices allow us to find a substitute function of the determinant.

We define a curve of heteroclinic tangencies by the following system

$$\begin{cases} F(X,\alpha) &= 0, \\ g(X,\alpha) &= 0, \end{cases} \tag{7.58}$$

where $g(X,\alpha)$ is computed as the last component of the solution vector in the K_1-dimensional bordered system:

$$\begin{pmatrix} F_X(X,\alpha) & b \\ c^T & 0 \end{pmatrix} \begin{pmatrix} v \\ g \end{pmatrix} = \begin{pmatrix} 0_{(K_1-1)} \\ 1 \end{pmatrix}, \tag{7.59}$$

for suitable vectors $b, c \in \mathbb{R}^{K_1-1}$.

If c is close to the nullvector of $F_X(X,\alpha)$ and b is close to the nullvector of $F_X^T(X,\alpha)$, then the matrix

$$M = \begin{pmatrix} F_X(X,\alpha) & b \\ c^T & 0 \end{pmatrix} \tag{7.60}$$

is nonsingular at (X,α) and (7.59) has a unique solution. In practical computations, c and b are approximations of the null vectors of $F_X(X,\alpha)$ and $F_X^T(X,\alpha)$, respectively.

In the continuation of heteroclinic tangencies b and c are computed in the curve initializer *init_HetT_HetT* and stored in the fields *hetTds.b* and *hetTds.c* of the global variable *hetTds*.

The vectors b and c must be adapted during the continuation of hetereoclinic tangencies to keep the matrix M nonsingular. Towards this end we use the *SVD* decomposition $[U,S,V] = svd(full(F_X(X,\alpha)))$ where U,V are orthogonal matrices and S is a diagonal matrix, and $F_X(X,\alpha) = USV^{'}$. Using the fact that c is normalized right nullvector of F_X we have:

$$F_X(X,\alpha)c = USV^{'}c = 0.$$

By the orthogonality of U, we get $SV^{'}c = 0$. The only possibility for the null vector of S is $V^{'}c = [0,\ldots,0,1]^T$. Sine V is an orthogonal matrix, we finally obtain: $c = V[0,\ldots,0,1]^T$. That means c is the last column of V.

For the left nullvector b, we have

$$b^T F_X(X,\alpha) = b^T USV^{'} = 0.$$

By the orthogonality of V, we get $b^T US = 0$. The only possibility for the null vector of S is $b^T U = [0,\ldots,0,1]$. Since U is an orthogonal matrix, we finally obtain: $b = U[0,\ldots,0,1]^T$. That means b is the last column of U.

By now it is fairly clear that homoclinic tangencies can be computed in essential the same way.

7.5 Examples and applications

We consider the generalized Hénon map (GHM)

$$F : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \mapsto \begin{pmatrix} x_2 \\ \alpha - \beta x_1 - x_2^2 + R x_1 x_2 + S x_2^3 \end{pmatrix}, \tag{7.61}$$

which appears in numerous theoretical studies of homoclinic bifurcations. For $\alpha = 0.3$, $\beta = -1.057$, $R = -1.057$ and $S = 0$, F^2 has two fixed points, namely

$$X_0 = (0.4666170238049, 0.4666170238049)^T$$

and

$$X_1 = (-0.4286170238049, -0.4286170238049)^T$$

with multipliers 2.76261564, 0.24558887 at X_0 and multipliers 3.18306252 and 0.50775797 at X_1.

7.5.1 Heteroclinic connections and tangencies

We use the algorithm as described in §7.3.2, to compute the one-dimensional unstable manifold of the saddle point X_0, by calling the MATLAB function *Umanifold.m*. To this end, first we set the accuracy parameters *amax, amin, dmax, dmin, damax, damin, dk* and *epsb* corresponding to parameters α_{max} , α_{min}, Δ_{max}, Δ_{max}, $(\Delta\alpha)_{max}$, $(\Delta\alpha)_{min}$,Δ_k and ϵ_b, respectively, as described in §7.3.2. We use the routines:

```
epsb=1e-10;Arc=8.6;dk=1e-3;amax=0.6;amin=0.2;
dmin=0.001;damax=0.07;damin=0.0001;dmax=0.2;
p0=[0.46661702380495;0.46661702380495];
p=[0.3;-1.057;-0.5;0];%alpha,beta,R,S
lamb=2.76261564458262;
del=1e-2;
v=[0.51553133957551;-0.85687072415592];
MM=Umanifold(p0,Arc,amax,amin,dmax,dmin,damax,
amin,dk,p,epsb,lamb,del,v);
for i=1:size(M,2) hold on,
plot(M(1,i),M(2,i),'-r.'), end
hold on
plot(0.46661702380495,0.46661702380495,'--sg')
```

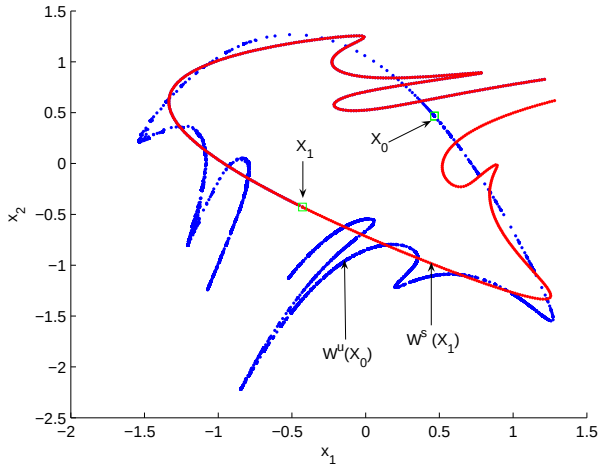


Figure 7.3: The transversal intersection of unstable and stable manifolds of F at X_0 and X_1, respectively.

where p_0 represents the saddle point X_0, p is the vector of parameters (α, β, R, S), λ is the multiplier of F corresponding to X_0 and Arc is the total length of the unstable manifold to be computed.
The plot of the unstable manifold is given in Figure 7.3.

We now use the algorithm for computing the stable manifold as described in §7.3.1, to compute the stable manifold of X_1. The routine is given by:

```
epsb=1e-7;Arc=5;dk=e-4;amax=0.5;amin=0.2;
dmin=0.01;damax=0.7;damin=1e-3;dmax=1e-2;
p0=[-0.42861702380495;-0.42861702380495];
p=[0.3;-1.057;-0.5;0];
p1=p0-1e-3*[-0.81439328674458;0.58031334165721];
MM=Smanifold(p0,p1,Arc,amax,amin,dmax,dmin,damax,
damin,dk,p,epsb);
```

The plot of the stable manifold is given in Figure 7.3.

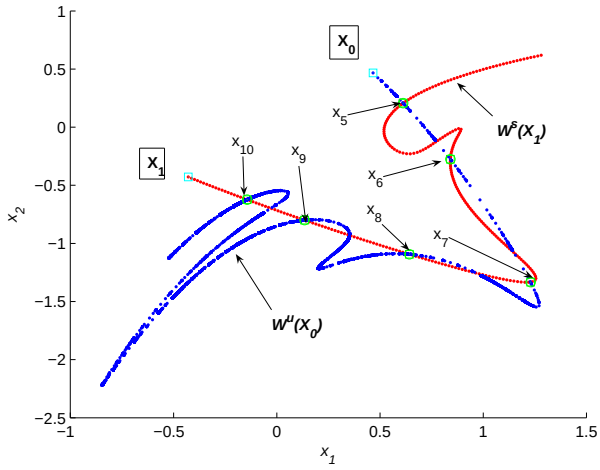


Figure 7.4: The transversal intersection of the unstable and stable manifolds of F at X_0 and X_1, respectively.

The transversal intersection of the unstable and stable manifolds, depicted in Figure 7.4, provides an initial approximation of a heteroclinic connection.

We continue the heteroclinic orbit in CL_MATCONTM, using the transversal intersection of invariant manifolds at X_0 and X_1, as an initial approximation. As indicated in Figure 7.4, the set of intersection points $\{x_5, x_6, \ldots, x_{10}\}$ is an initial approximation. However, to get a more accurate heteroclinc orbit we extend the initial approximation set by adding more points. To this end, we use iterations of F and F^{-1} and project the resulting points on $W^u(x_0)$ and $W^s(x_1)$, respectively.

We start from x_5 and compute the point $F^{-1}(x_5)$. By projecting the resulting point on $W^u(x_0)$, we compute x_4 as an approximation of a new intersection point of $W^u(x_0)$ and $W^s(x_1)$. We then apply F^{-1} on x_4 and by projecting the new point on $W^u(x_0)$ we compute x_3. We proceed with the same steps to compute the points x_2 and x_1.

We now use the same procedure by applying F on x_{10}. By projecting the

resulting point on $W^s(x_1)$ we compute x_{11} as an approximation of an intersection point of $W^u(x_0)$ and $W^s(x_1)$. We repeat the same steps to compute the points $x_{12}, x_{13}, \ldots, x_{16}$.

The resulting initial approximation is given by

$$C = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}\}$$

where

$$\begin{array}{rclrcl}
x_1 &=& (0.4666171; 0.4666171), & x_2 &=& (0.519502; 0.376394) \\
x_3 &=& (0.483172; 0.439158), & x_4 &=& (0.4731169; 0.456962) \\
x_5 &=& (0.612300; 0.206700), & x_6 &=& (0.841195; -0.276064) \\
x_7 &=& (1.229904; -1.332700), & x_8 &=& (0.641982; -1.093020) \\
x_9 &=& (0.134731; -0.799843), & x_{10} &=& (-0.143457; -0.623386) \\
x_{11} &=& (-0.333799; -0.495162), & x_{12} &=& (-0.380621; -0.462550) \\
x_{13} &=& (-0.404242; -0.445916), & x_{14} &=& (-0.416213; -0.437437) \\
x_{15} &=& (-0.422303; -0.433111), & x_{16} &=& (-0.428617; -0.428617)
\end{array}$$

The code below is the implementation in CL_MATCONTM:

```
C={x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12,x13,
x14,x15,x16};
p=[0.3;-1.057;-0.5;0];ap=[2];
opt = contset;
[x0,v0]=init_HE_HE(@Ghmap,C, p, ap,2);
opt=contset(opt,'MaxNumpoints',30);
opt=contset(opt,'Singularities',1);
opt=contset(opt,'Backward',1);
[xhet,vhet,shet,hhet,fhet]=
cont(@heteroclinic,x0,[],opt);
```

We detect two limit points (LP) on the heteroclinic orbit:

```
first point found
tangent vector to first point found
label = LP  , x = ( 0.450332 0.450332 0.464235
 0.427916 0.486124 0.391578 0.542144 0.295162
 0.680067 0.035058 0.973257 -0.628904  1.192852
```

```
-1.382567 0.417078 -0.981090 -0.036976 -0.709741
-0.254175 -0.571162 -0.355358 -0.504248 -0.402531
-0.472468  -0.424600 -0.457463 -0.434950 -0.450395
-0.439810 -0.447070 -0.444117 -0.444117  -0.004369
-0.005812 -1.009322 )

label = LP  , x = ( 0.471227 0.471227 0.487755
 0.443527 0.517117 0.392578 0.597800 0.245433
 0.806172 -0.186179 1.203153 -1.234826 0.805668
-1.173190 0.258458 -0.870745 -0.069068 -0.666717
-0.241453 -0.552315 -0.330135 -0.491249 -0.375727
-0.459196  -0.399232 -0.442484 -0.411379 -0.433796
-0.417667 -0.429288 -0.424423-0.424423   0.000138
 0.000184 -1.070206 )

elapsed time  = 1.8 secs
npoints curve = 30
```

In the computed LP points the first 32 components indicate the coordinates of the mesh points $x_1, \ldots, x_{16}$, the following 2 indicate Y_U and Y_S in the Riccati equations (7.32) and (7.35), respectively and the last component is the value of the control parameter β. A picture of the computed branch of heteroclinic orbits is given in Figure 7.6.

As a comparison, we also continued the branch of heteroclinic connections using the numeric Jacobian computed by finite differences. The continuation using finite differences lead to the same branch of heteroclinic connections as well as the same LP bifurcations. However, the elapsed time using finite differences was 13.2 (for 30 points). That means, using the symbolic Jacobian speeds up the continuation by a factor 7 times compared with the case when finite differences were used.

For the parameter values of the fold points, i.e. $\alpha = 0.3, \beta = -1.009322, R = -0.5, S = 0$ and $\alpha = 0.3, \beta = -1.070206, R = -0.5, S = 0$, we have a tangential intersection of the invariant manifolds. A tangential intersection of invariant manifolds is shown in Figure 7.5.

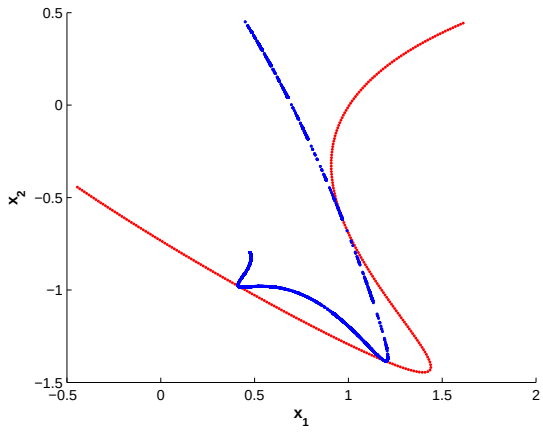


Figure 7.5: Tangential intersection of the invariant manifolds of (7.61) for $\alpha = 0.3; \beta = -1.009322; R = -0.5; S = 0$

Next we continue the limit points in two parameters, starting from the LP on the heteroclinic connections. This curve is given in Figure 7.7.

7.5.2 Homoclinic connections and tangencies

Now we consider the parameter values $\alpha = -0.4$, $\beta = 1.03$, $R = -0.1$ and $S = 0$. F has fixed point $X_0 = (-1.62114638486, -1.62114638486)$ with the multipliers 0.2775591559 and 3.1268482523.

We compute $W^u(X_0)$ and $W^s(X_0)$ and determine their intersection points to be used as initial data for the homoclinic continuation. Figure (7.8) depicts $W^u(X_0)$ and $W^s(X_0)$ along with their intersection points. We continue the homoclinic orbit in CL_MATCONTM, using the transversal intersection of invariant manifolds at X_0, as an initial approximation. The initial approximation is given by

$$C = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\}$$

where

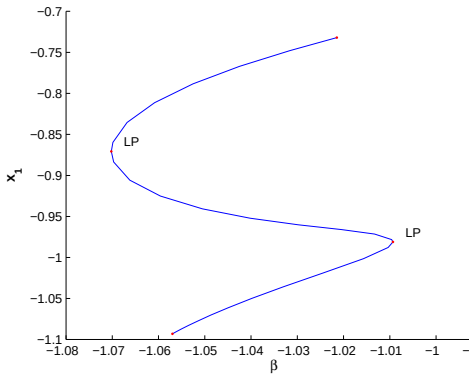


Figure 7.6: Fold points on the branch of heteroclinic connections of (7.61)

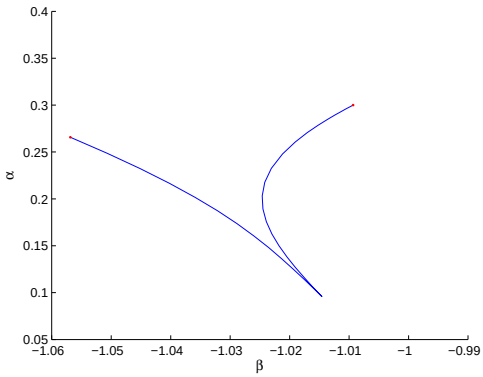


Figure 7.7: Branch of heteroclinic tangencies of (7.61)

$$x_1 = (-1.6211464; -1.62114638), \quad x_2 = (-1.56200000; -1.44300000)$$
$$x_3 = (-1.4430000; -1.09878560), \quad x_4 = (-1.09878560; -0.27959456)$$
$$x_5 = (-0.27959456; 0.62285460), \quad x_6 = (0.62285460; -0.48255079)$$
$$x_7 = (-0.48255079; -1.24433961), \quad x_8 = (-1.24433961; -1.51139945)$$
$$x_9 = (-1.51139945; -1.59072793), \quad x_{10} = (-1.59072793; -1.61409646)$$

We detect two limit points (LP):

```
label = LP  , x = ( -1.704631 -1.704631 -1.668284 -1.584880
-1.584880 -1.324870 -1.324870 -0.606436 -0.606436 0.622163
 0.622163 -0.076364  -0.076364 -1.091525 -1.091525 -1.515017
-1.515017 -1.649326 -1.649326 -1.688864 -0.002296 -0.001094
 1.109749 )

label = LP  , x = ( -1.586188 -1.586188 -1.559729 -1.505309
-1.505309 -1.345718 -1.345718 -0.912761 -0.912761 -0.014305
```

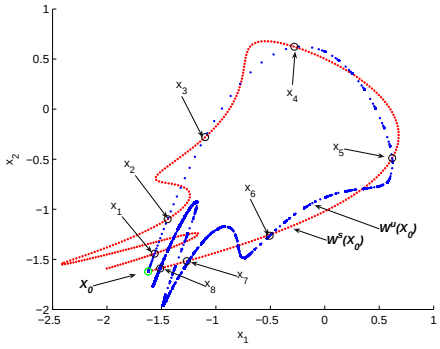


Figure 7.8: Transversal intersection of the invariant manifolds of (7.61) for $\alpha = -0.4$, $\beta = 1.03$, $R = -0.1$ and $S = 0$

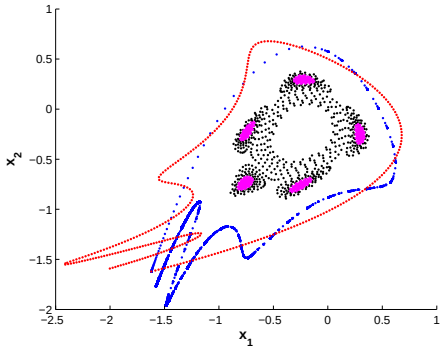


Figure 7.9: Closed invariant curve of (7.61) for $\alpha = -0.4$, $\beta = 1.03$, $R = -0.1$ and $S = 0$, superposed on Figure 7.8

```
-0.014305 0.508497 0.508497 -0.643580 -0.643580 -1.288433
-1.288433 -1.501341 -1.501341 -1.562917 0.000551 0.000252
 0.996984 )

elapsed time  = 1.3 secs
npoints curve = 35
```

In the computed LP points the first 20 components indicate the coordinates of the mesh points $x_1, \ldots, x_{10}$, the following 2 indicate Y_U and Y_S in the Riccatti equations (7.32) and (7.35), respectively and the last component is the value of the control parameter β. The computed branch of homoclinic connections is presented in Figure 7.10.

Now we can continue the curve of limit points in two parameters, starting from the *LP* on the homoclinic connections. This curve is given in Figure 7.11.

In Figure 7.9 we present a closed invariant curve, obtained by simulation, for $\alpha = -0.4; \beta = 1.03; R = -0.1; S = 0$ superposed on Figure 7.8. This invariant curve is destructed at the tangency.

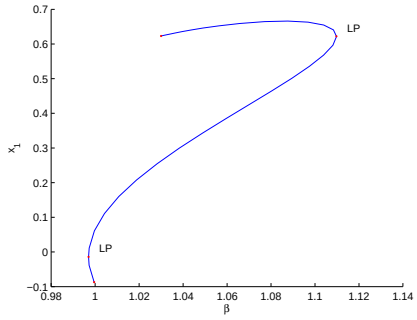


Figure 7.10: Fold points on the branch of homoclinic connections of (7.61)

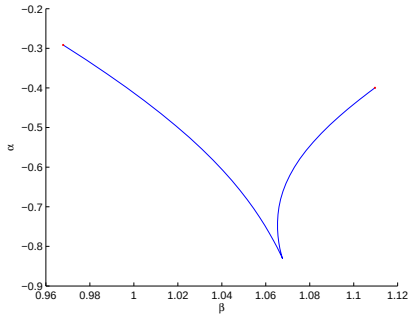


Figure 7.11: Branch of homoclinic tangencies of (7.61)

Chapter 8

Future work

There is a clear need for further development of the work described in the present thesis. An important goal is to provide an interactive environment, i.e. a graphical user interface (GUI), for CL_MATCONTM, in order to take advantage of the GUI capabilities of MATLAB. The GUI for CL_MATCONTM should be similar to that of MATCONT, in many aspects. It would differ in some aspects, especially in branch switching. For an iterated map one key element is to consider the iteration number of the map, e.g. for the branches of different periods that emanate from codim 1 and codim 2 bifurcation points.

As an alternative to symbolic derivatives, we implemented automatic differentiation to compute the critical normal form coefficients that appear in the normal form, in order to compute derivatives accurately when symbolic derivatives are not available, and to speed up the computations. With AD the elapsed time grows linearly whereas it grows nonlinearly when using symbolic derivatives. AD should also be implemented in MATCONT for computing the critical normal form coefficients of codim l and codim 2 bifurcations of an ODE.

In Chapter 7 we implemented numerical algorithms for the continuation of branches of connecting orbits of a map in one parameter, which includes an improved algorithm for the continuation of invariant subspaces. In the continuation we need to compute the Jacobian matrix of the defining system. Depending on the number of initial approximation points, we may have a large number of equations

in the defining system. To speed up the continuation we implemented algorithms to compute the symbolic Jacobian of connecting orbits. It uses a sparse matrix as opposed to the full Jacobian used in the finite differences. However, it should be possible to improve the implementation of the symbolic Jacobian significantly by converting loops into array operations, i.e. vectorizing the code. MATLAB is optimized for performing operations on arrays and provides a rich set of functions and many expressive indexing schemes that make it possible to vectorize code. So we should vectorize the code for the symbolic Jacobian by converting loops into array operations.

The next aim in connecting orbits is to analyze the bifurcation of transversal heteroclinic orbits, when one end point loses its hyperbolicity at a critical parameter, while transversality remains valid. This non-hyperbolic situation arises, for example, when one fixed point undergoes a fold or a flip bifurcation, while the second fixed point stays hyperbolic. In this case, every point of a connecting orbit is lying in the intersection of an unstable manifold, and a center-stable manifold. Since the bifurcation of fixed points is well understood, we are interested in the continuation of the saddle to fold and saddle to flip orbits in a neighborhood of the critical parameter and also detection and location of the corresponding bifurcations along these orbits.

We implemented algorithms for computing one-dimensional stable and unstable manifolds of a saddle point of a planar map. These algorithms should be generalized to compute two-dimensional stable and unstable manifolds in a higher-dimensional state space.

Nederlandse samenvatting

Het uitgangspunt van deze thesis is een afbeelding $f : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n, (x, \alpha) \to f(x, \alpha)$, waarbij we x een toestandsvector noemen en α een parametervector. Een vast punt van de afbeelding is een punt x for waarvoor $f(x) = x$; een cykel met periode J is een $J-$ tal vectoren $\{x_1, x_2, \ldots, x_J\}$ waarvoor $f(x_j) = x_{j+1}$, $j = 1..J-1$, $f(x_J) = x_1$.

Een connectie is een tweezijdig oneindige familie $\{\ldots, x_{j-1}, x_j, x_{j+1}, \ldots\}$ met de eigenschap dat $f(x_j) = x_{j+1}$ voor alle j en dat x_j nadert tot een vast punt $x_{-\infty}$ als j nadert tot $-\infty$, en tot een vast punt x_∞ als j nadert tot ∞.

De connectie heet homoclinisch als $x_{-\infty} = x_\infty$ en heteroclinisch als $x_{-\infty} \neq x_\infty$. Vaste punten en cykels zijn generieke fenomenen voor afbeeldingen; het is wellicht minder evident dat dit ook waar is voor homoclinische en heteroclinische connecties.

De thesis behoort tot het brede domein van dynamische systemen, maar eveneens tot dat van de numerieke algoritmen en van de toegepaste wiskunde. De toepassingen die wij expliciet vermelden behoren tot de economie, speltheorie en biologie maar het potentieel is ruimer.

In Hoofdstuk 1 (Aspecten van CL_MATCONTM) geven we een overzicht van de structuur van het softwarepakket CL_MATCONTM (Command line Matlab Continuation for Maps). Ruwweg gezegd is dit een universeel continuatie - algoritme toegepast op types van krommen die specifiek zijn voor afbeeldingen. Alhoewel er enige gelijkenissen zijn met het corresponderende pakket MATCONT voor gewone differentiaalvergelijkingen, is de hele structuur grondig verschillend.

Een basisingredient voor alle routines is de *mapfile* van de afbeelding, d.w.z. een stel van computationele routines die de basisinformatie verschaffen over hoe

de afbeelding wiskundig gedefinieerd is, of er symbolische afgeleiden beschikbaar zijn, welke functies de gebruiker eventueel wenst te monitoren tijdens de verschillende continuaties, enzoverder.

In Hoofdstuk 2 (Locale bifurcatieanalyse) frissen we de wiskundige achtergrond op waarvoor de numerieke algoritmen ontwikkeld werden. We geven een breed overzicht van het bifurcatiegedrag van discrete dynamische systemen, vaste punten, cykels en heteroclinische en homoclinische banen. Daarbij geven we de normaalvormen van alle codimensie 1 en codimensie 2 bifurcaties van cykels. Dit bevat in wezen geen origineel werk maar wordt hier voor het eerst op die manier samengevat en is fundementeel voor de rest van de thesis.

Hoofdstuk 3 (Continuatie van codimensie 1 bifurcaties; takwisseling) beschrijft de numerieke algoritmen die we gebruikten om de continuatie van codimensie 1 bifurcaties van afbeeldingen te continueren, d.w.z. van limietpunten (folds), periodeverdubbelingspunten (flip, perioddoubling) en torusbifurcaties (Neimark-Sacker). In de laatste bladzijden van dit hoofdstuk geven we grafische representaties van de bifurcatiestructuur van cykels van afbeeldingen tot en met codimensie 2. De detectiegraaf toont welke bifurcatiepunten van hogere codimensie gedetecteerd kunnen worden op krommen met een lagere codimensie. De vertakkingsgraaf toont welke bifurcatiekrommen van lagere codimensie kunnen opgestart worden vanaf punten met een hogere codimensie. De volledige informatie bevat in deze grafen is numeriek geimplementeerd en in vele gevallen gebeurt dit voor de eerste keer in dit werk.

In Hoofdstuk 4 geven we algoritmische en numerieke details. In sectie 4.1 geven we vooreerst de computationele details over het definierend systeem, de initializatie en de adaptatie van de Neimark-Sacker krommen. Dit is het meest gecompliceerde van de drie codimensie 1 gevallen; de twee andere (limietpunten en periodeverdubbeling) zijn wat eenvoudiger.

In de secties 4.2-3 geven we recursieve formules voor de eerste en tweede orde afgeleiden van cykels met betrekking tot ofwel de toestandsvariabelen ofwel de parameters. Deze worden systematisch gebruikt in het computationele kader van CL_MATCONTM. Men kan ze berekenen met behulp van symbolische afgeleiden (SD) als deze voorzien zijn in de *mapfile*; anders worden ze benaderd met behulp van eindige differenties (FD). In sectie 4.4 beschrijven we meer bepaald hoe eindige differenties van directionele afgeleiden van orde ten hoogste 5 berekend

kunnen worden. Men moet er wel rekening mee houden dat voor orde hoger dan 3 deze afgeleiden niet erg betrouwbaar zijn. Dit wordt verder duidelijk gemaakt in sectie 4.8.

In secties 4.5-7 beschrijven we de implementatie van automatische differentiatie (AD) in CL_MATCONTM, hetgeen we beschouwen als een van de meest originele bijdragen in deze thesis. Het onderliggend idee is de introductie van een Matlab klasse van Taylor veeltermen (van een zekere orde) van functies van een enkele variabele. We overladen dan de standaardfuncties in Matlab met operaties van deze klasse, hetgeen toelaat multilineaire uitdrukkingen te berekenen van bijvoorbeeld de vorm $(f^{(J)})_{xxxxx}(q,q,q,q,q)$ in het punt x_0 voor de Taylor expansie vam $f^{(J)}(x_0+tq)$ tot orde 5. Hierbij is f de afbeelding gedefinieerd op de vectorvariabele x en geëvalueerd in x_0. Verder is t de onafhankelijk variabele in de Taylorontwikkeling, q is een richtingsvector en J is het iteratiegetal. De polarizatie - identiteiten kunnen dan gebruikt worden om het geval van $(f^{(J)})_{xxxxx}(q_1,q_2,q_3,q_4,q_5)$ (met verschillende vectoren q_1,q_2,q_3,q_4,q_5) te herleiden tot het geval $q_1=q_2=q_3=q_4=q_5$.

Op deze manier kunnen we afgeleiden van hogere orde exact berekenen, zelfs zonder te beschikken over symbolische afgeleiden. We merken op dat afgeleiden van orde tot en met vijf nodig zijn voor het berekenen van de normaalvormcoefficienten en dat de exactheid van deze coefficienten essentieel is voor de beschrijving van het gedrag van het dynamische systeem in de omgeving van het bifurcatiepunt en voor computationele algoritmen zoals takwisseling.

In Sectie 4.8 vergelijken we de drie strategieën voor differentiatie (SD,FD,AD). Het is geen verrassing that SD en AD dezelfde accuraatheid hebben terwijl FD onbetrouwbaar is voor afgeleiden van orde hoger dan 2. Interessanter is dat AD sneller is dan SD for cykels van voldoend hoge orde. In sommige gevallen geven we dus de voorkeur aan AD, zelfs als SD beschikbaar is.

De Hoofdstukken 5 (Toepassingen in de biologie) en 6 (Toepassingen in de economie) vormen het gedeelte "toegepaste wiskunde" van de thesis. We bestuderen vier afbeeldingen, twee uit de biologie en twee uit de economie. We nemen telkens een afbeelding uit de bestaande literatuur in deze vakgebieden en bestuderen het met de computationele methoden die beschikbaar zijn in CL_MATCONTM, na enige voorbereidende analytische studie. We vinden telkens nieuwe resultaten, meestal betreffende de berekening van stabiliteitsgrenzen maar ook voor wat be-

treft de detectie en berekening van dynamisch belangrijke aspecten zoals multistabiliteit van cykels met dezelfde of met verschillende periodes. In enkele gevallen vonden we zelfs echte fouten in de gepubliceerde literatuur.

In Hoofdstuk 7 beschrijven we de implementatie, als deel van CL_MATCONTM, van algoritmen voor de continuatie (in een parameter) van heteroclinische en homoclinische connecties van zadelpunten, detectie van de limietpunten van deze connecties (*tangencies*) en de continuatie (in twee parameters) van deze limietpunten. Afgezien van de implementatie in Matlab bestaat het origineelste deel van dit werk enerzijds uit een nieuwe wiskundige beschrijving van de continuatie van invariante deelruimten (gebaseerd op alleen maar argumenten uit de lineaire algebra) en anderzijds uit een randingsmethode voor de continuatie van de limietpunten.

In Hoofdstuk 8 (Toekomstig werk) beschrijven we aan de gang zijnde of geplande extensies van het werk beschreven in dit proefschrift. Hiertoe behoort het bouwen van een GUI voor CL_MATCONTM (uiterst nuttig), het invoeren van AD methoden in MATCONT (de ODE tegenhanger van CL_MATCONTM), en het vectorizeren van de Matlab code voor homoclinische en heteroclinische connecties en hun limietpunten. Misschien is het vectorizeren ook nuttig voor andere delen van de code, inbegrepen alle bifurcaties van cykels. Afgezien hiervan is er ook behoefte aan verdere wiskundige studie van homoclinische en heteroclinische connecties, in het bijzonder voor wat betreft de situatie waarbij de hyperboliciteit in een of beide eindpunten verloren gaat. Een verwant algoritmisch en numeriek probleem is de continuatie van zadel-tot-limietpunt en van zadel-tot-flip connecties.

Bibliography

[1] H. N. Agiza, Explicit stability zones for Cournot games with 3 and 4 competitors, Chaos, Solitons & Fractals, 9 (1998) 1955-1966.

[2] H. N. Agiza, On the Analysis of Stability, Bifurcation, Chaos and Chaos Control of Kopel Map, Chaos, Solitons & Fractals, 10 (1999) 1909-1916.

[3] H. N. Agiza, G. I. Bischi, M. Kopel, Multistability in a dynamic Cournot game with three oligopolists, Math. Comput. Simulation, 51 (1999) 63-90.

[4] H. N. Agiza, A. A. Elsadany, M. Kopel, Nonlinear dynamics in the Cournot duopoly game with heterogeneous players, Physica, A 320 (2003) 512-524.

[5] H. N. Agiza, A. S. Hegazi, A. A. Elsadany, Complex dynamics and synchronization of a duopoly game with bounded rationality, Math. Comput. Simulation, 58 (2002) 133-146.

[6] E. Ahmed, H. N. Agiza, Dynamics of a Cournot game with n-competitors, Chaos, Solitons & Fractals, 9 (1998) 1513-1517.

[7] E. L. Allgower, K. Georg, Numerical Continuation Methods: An Introduction, Springer-Verlag, 1990.

[8] D. R. Aronson, M. Chory, G. Hall, and R. McGehee, Bifurcations from an invariant circle for two-parameter families of maps on the plane: A computer assisted study, Comm. Math. Phys., 83 (1982), 303–354.

[9] D. R. Anderson, N. G. Myran, D. L. White, Basin of attraction in a Cournot duopoly model of Kopel, J. Diff. Eqns. Appl., Vol. 11, No. 10, (2005) 879-887.

[10] A. Back, J. Guckenheimer, M. Myers, F. Wicklin, and P. Worfolk, DsTool: Computer assisted exploration of dynamical systems, Notices Amer. Math. Soc., 39 (1992)303-309.

[11] M. Begon, J. L. Harper, and C. R. Townsend, Ecology: Individuals, Populations, and Communities, 3rd edition, 1996. Blackwell Science Ltd. Cambridge, MA.

[12] W.-J. Beyn and J.-M. Kleinkauf, The numerical computation of homoclinic orbits for maps. SIAM J. Numer. Anal., 34 (1997), 1207-1236.

[13] W. -J. Beyn, T. Hüls, J.-M. Kleinkauf, and Y.-K. Zou, Numerical analysis of degenerate connecting orbits of maps, Int. J. of Bifurcation and Chaos, 14 (10), (2004) 3385-3407.

[14] W.-J. Beyn, A. Champneys, E. Doedel, W. Govaerts, Yu. A. Kuznetsov and B. Sandstede, Numerical continuation, and computation of normal forms, in Handbook of Dynamical Systems, Vol. 2, B. Fiedler, ed., Elsevier Science, Amsterdam, 2002, pp. 149–219.

[15] G. I. Bischi, A. Mammana, L. Gardini, Multistability and cyclic attractors in duopoly games, Chaos, Solitons & Fractals, 11 (2000) 543-564.

[16] F. Brauer, C. Castillo-Chávez, Mathematical Models in Population Biology and Epidemiology, Springer-Verlag, 2000.

[17] H. Broer, R. Roussarie, and C. Simó, Invariant circles in the Bogdanov-Takens bifurcation for diffeomorphisms, Ergodic Theory Dynamical Systems, 16 (1996) 1147-1172.

[18] H. M. Bücker, Looking for narrow interfaces in automatic differentiation using graph drawing, FGCS., 21 (2005) 1418- 1425.

[19] H. Caswell, Matrix Population Models, Sunderland, Massachusetts: Sinaur Ass. Inc. Publishers 2001.

[20] S. N. Chow, and J. Hale, Methods of Bifurcation Theory. Grundlehren der mathematischen Wissenschaften, Vol. 251. Springer-Verlag, 1982.

[21] T. F. Coleman, and A. Verma, ADMAT: An automatic differentiation toolbox for MATLAB. Tech. rep., Computer Science Department, Cornell University, 1998.

[22] T. F. Coleman, and A. Verma, The efficient computation of sparse Jacobian matrices using automatic differentiation. SIAM J. Sci. Comput., 19, 4, (1998) 1210 - 1233.

[23] A. Cournot, Researches Into the Principles of Wealth, 1963 (English Translation), Irwin Paperback Classics in Economics, (original: Recherches sur les principes mathématiques de la théorie des richesses, 1838).

[24] J. M. Cushing, R.F. Costantino, B. Dennis, R. A. Deshamias, and A. M. Henson, Nonlinear population dynamics: models, experiments and data. J. Theor. Biol., 194 (1998) 1-9.

[25] J. W. Demmel, L. Dieci, and M. J. Friedman, Computing connecting orbits via an improved algorithm for continuing invariant subspaces, Siam J. Sci. Comput., Vol. 22, No. 1, (2000) 81-94.

[26] B. Dennis, R. A. Deshamais, J. M. Cushing, and R. F Costantino, Transition in population dynamics: equilibria to periodic cycles to aperiodic cycles, J. Anim. Ecol., 6b, (1997) 704-729.

[27] L. Dieci, and M. J. Friedman, Continuation of invariant subspaces, Numer. Linear Algebra Appl., 8 (2001) 317-327.

[28] E. J. Doedel, A. R. Champneys, T. F. Fairgrieve, Yu. A. Kuznetsov, B. Sandstede, and X.-J. Wang, AUTO97-AUTO2000: Continuation and Bifurcation Software for Ordinary Differential Equations (with HomCont), User's Guide, Concordia University, Montreal, Canada (1997-2000). Available from `http://indy.cs.concordia.ca`.

[29] A. Dhooge, W. Govaerts, and Yu. A. Kuznetsov, MATCONT: A MATLAB package for numerical bifurcation analysis of ODEs, ACM Transactions on Mathematical Software, 29(2), (2003) pp. 141-164.

[30] A. Dhooge, W. Govaerts, Yu. A. Kuznetsov, W. Mestrom, A. Riet and B. Sautois: CL_MATCONT: A continuation toolbox in MATLAB, (2004) `http://www.matcont.UGent.be`.

[31] J. Edmunds, J. M Cushing, R. F. Costantino, S. M. Henson, B. Dennis and R. A. Desharnais, Park's Tribolium competition experiments: a non-equilibrium species coexistence hypothesis, J. Animal Ecology, 72 (2003) 703-712.

[32] J. P. England, B. Krauskopf and H. M. Osinga, Computing one-dimensional stable manifolds and stable sets of planar maps without the inverse, Siam J. Applied Dynamical Systems, Vol. 3, No. 2, 2004, pp. 161-190.

[33] J. D. Fox, Cannibalism in natural populations. Ann. Rev. Ecol.Syst., 6 (1975) 87-106.

[34] N. K. Gavrilov, and L. P. Shilnikov, On three-dimensional systems close to systems with a structurally unstable homoclinic curve: I, Math. USSR-Sb., 17 (1972) 467-485.

[35] N. K. Gavrilov, and L. P. Shilnikov, On three-dimensional systems close to systems with a structurally unstable homoclinic curve: II, Math. USSR-Sb., 19 (1973) 139-156.

[36] G. F. Gause, The Struggle for Existence, Hafner Press, NewYork, 1934 (reprinted 1964).

[37] G. H. Golub, and C. F. Van Loan, Matrix Computations, 2nd ed., The John Hopkins University press, Baltimore, MD, 1989.

[38] W. Govaerts, Numerical Methods for Bifurcations of Dynamical Equilibria, SIAM, Philadelphia, 2000.

[39] W. Govaerts, and R. Khoshsiar Ghaziani, Stable cycles in a Cournot duopoly model of Kopel, to appear in J. Comp. Appl. Math.

[40] W. Govaerts, and R. Khoshsiar Ghaziani, Numerical Continuation of fixed points of maps in CL_MATCONTM, Proceedings of the Fifth Euromech Nonlinear Dynamics Conference, Eindhoven 2005. Eds. D.H. van Campen, M.D. Lazurko and W.P.J.M. van den Oever, pp. 1740-1749. ISBN 90 386 2667 3.

[41] W. Govaerts, and R. Khoshsiar Ghaziani, Numerical bifurcation analysis of a nonlinear stage structured cannibalism population model, J. Diff. Eqns. Appl., 12,10 (2006) 1069-1085.

[42] W. Govaerts, R. Khoshsiar Ghaziani, Yu. A. Kuznetsov, and H. G. E. Meijer, Numerical methods for two-parameter local bifurcation analysis of maps, SIAM J. Sci. Comp., 29 (2007), 2644-2667.

[43] W. Govaerts, R. Khoshsiar Ghaziani, Yu. A. Kuznetsov, and H. G. E. Meijer, CL_MATCONTM: A toolbox for continuation and bifurcation of cycles of maps, (2006) `http://www.matcont.UGent.be`.

[44] W. Govaerts,Yu. A. Kuznetsov, and B. Sijnave, Bifurcations of maps in the software package CONTENT, in Computer Algebra in Scientific Computing—CASC'99 (Munich), V. G. Ganzha, V.G. E. W. Mayr, and E. V. Vorozhtsov, eds., Springer, Berlin, 1999, pp. 191-206.

[45] A. Griewank, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation. Frontiers in Applied Mathematics, No. 19. SIAM, Philadelphia, PA, 2000.

[46] A. Griewank, D. Juedes, and J. Utke, Algorithm 755: ADOL-C, a package for the automatic differentiation of algorithms written in C/C++. ACM Trans. Math. Softw., 22, 2, (1996) 131 - 167.

[47] A. Griewank, Some bounds on the complexity of gradients. In: Complexity in Nonlinear Optimization, P. Pardalos, Ed. World Scientific Publishing Co., Inc., River Edge, NJ, (1993) 128161.

[48] V. G. Gruzdev, and Ju. I. Neimark, A symbolic description of motion in the neighborhood of a not structurally stable homoclinic structure and of its change in transition to close systems, In: System Dynamics, Vol. 8, Gorkii State University, Gorkii, 1975, pp. 13–33. In Russian.

[49] J. Guckenheimer, and P. Holmes, Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields. Springer-Verlag, 1983. Applied Mathematical Sciences, Volume 42.

[50] M. W. Hirsch, C. C. Pugh, and Shub, M., Invariant manifolds, Springer Verlag, Berlin-New York, 1997, Lecture Notes in Mathematics, Vol.583.

[51] P. Holmes, and J. Guckenheimer, Nonlinear Dynamical Systems and Bifurcations of Vector Fields, Springer-Verlag, 1983.

[52] T. Hüls, Bifurcation of connecting orbits with one nonhyperbolic fixed point for maps, SIADS, Vol. 4, No. 4, 2005, 985-1007.

[53] G. Iooss, and D. Joseph, Elementary stability and bifurcation theory. Springer-Verlag, New York, 1980.

[54] A. I. Khibnik, Yu. A. Kuznetsov, V. V. Levitin, and E. V. Nikolaev, Continuation techniques and interactive software for bifurcation analysis of ODEs and iterated maps, Physica, D 62 (1993) 360–371.

[55] R. Khoshsiar Ghaziani, W. Govaerts, Yu. A. Kuznetsov, and H.G. E Meijer, Numerical continuation of connecting orbits of maps in MATLAB, submitted.

[56] R. Khoshsiar Ghaziani, J. D. Pryce, and W. Govaerts, Automatic differentiation in the computation of normal form coefficients of bifurcations of cycles, submitted.

[57] M. Kopel, Simple and complex adjustment dynamics in Cournot duopoly models, Chaos, Solitons & Fractals, 12 (1996) 2031-2048.

[58] B. Krauskopf, and H. M. Osinga, Growing 1D and quasi 2D unstable manifolds of maps, J. Comp. Physics 146 (1998) 404-419.

[59] B. Krauskopf, and H. M. Osinga, Growing unstable manifolds of planar maps, preprint, 1997.

[60] B. Krauskopf, and H. M. Osinga, Investigating torus bifurcations in the forced Van der Pol oscillator. In E. J. Doedel and L.S. Tuckerman, editors. Numerical methods for bifurcation problems and Large-Scale Dynamical Systems, volume 119 of IMA vol. Math. Appl., pages 199-208. Springer, New York, 2000.

[61] Yu. A. Kuznetsov and V. V. Levitin, CONTENT: A Multiplatform Environment for Analyzing Dynamical Systems, CWI, Amsterdam, 1997. Available via ftp from `ftp.cwi.nl/pub/CONTENT`.

[62] Yu. A. Kuznetsov, Elements of Applied Bifurcation Theory, 3rd edition, Springer-Verlag, New York, 2004.

[63] Yu. A. Kuznetsov and H. G. E. Meijer, Numerical normal forms for codim 2 bifurcations of maps with at most two critical eigenvalues, SIAM J. Sci. Comp., 26 (2005) 1932-1954.

[64] Yu. A. Kuznetsov and H. G. E. Meijer, Remarks on interacting Neimark-Sacker bifurcations, J. Diff. Eqs. Appl., 12 (2006) 1009-1035.

[65] P. H. Leslie, and J. C. Gower, The properties of a stochastic model for two competing species, Biometrika, 45 (1958) 316-330.

[66] P. H. Leslie, T., Park, D. B., Mertz, The effect of varying the initial numbers on the outcome of competition between two Tribolium species, J. of Animal Ecology, 37 (1968) 9-23.

[67] J. E. Linehan, R. S. Gregpy, and D. C. Schneider, Predation risk of age-0 cod(Gadus) relative to depth and substrate in coastal water. J. Exp. Mar. Biol. Ecol., 263 (2001) 25-44.

[68] MATLAB, The Mathworks Inc., `http://www.mathworks.com`.

[69] H. G. E. Meijer, Codimension 2 bifurcations of iterated maps, PhD Thesis, Utrecht University, 2006.

[70] R. A. Myers, W. Blanchard, K. R. Thompson, 1990, Summary of North Atlantic fish recruitment 1942-1987. Can.Tech.Rep.Fish. & Aquat.Sci. 1743.

[71] J. D. Murray, Mathematical Biology, 2nd corrected edition, Springer-Verlag, 1993.

[72] J. Nash, Non-cooperative Games, Annals of Mathematics 54 (1951) 286-295.

[73] Ju. I. Neimark, On some cases of periodic motions depending on parameters, Dokl. Akad. Nauk SSSR., 129 (1959) 736-739. In Russian.

[74] Ju. I. Neimark, Motions close to doubly-asymptotic motion, Soviet Math. Dokl., 8 (1967) 228-231.

[75] S. Newhouse, J. Palis, and F. Takens, Bifurcations and stability of families of diffeomorphisms, Inst. Hautes Etudes Sci. Publ. Math., 57 (1983) 5-71.

[76] H. Nusse and J. Yorke, Dynamics: Numerical explorations, 2nd edition, Springer-Verlag, New York, 1998.

[77] K. Okuguchi, expectations and stability in oligopoly models. Springer-Verlag, 1979.

[78] K. Okuguchi, and F. Szidarovszky , The theory of oligopoly with multi-product firms, 2nd edition,, Springer-Verlag, 1999.

[79] J. Palis, and W. De Melo, Geometric Theory of Dynamical Systems, Springer-Verlag, 1982.

[80] J. Palis and F. Takens, Hyperbolicity and sensitive chaotic dynamics at homoclinic bifurcations, Cambridge University Press, 1995.

[81] T. Park, Experimental studies of interspecies competition. I, Competition between populations of the flour beetles Tribolium confusum Duval and Tribolium castaneum Herbst, Ecological Monographs, 18 (1948) 265-308.

[82] T. Park, Experimental studies of interspecies competition. II, Temperature humidity and competition in two species of Tribolium. Physiological Zoology, 27 (1954) 177-238.

[83] T. Park, Experimental studies of interspecies competition. III, Relation of initial species proportion to the competitive outcome in populations of Tribolium. Physiological Zoology, 30 (1957) 22-40.

[84] T. Park, P. H. Leslie, and D. B. Mertz, 1964, Genetic strains and competition in populations of Tribolium. Physiological Zoology, 37 (1964) 97-162.

[85] G. A. Polis, The evolution and dynamics of intraspecific predation, Ann. Rev. Ecol.Sys., 12 (1981) 225-251.

[86] J. Palis, and F., Takens, Hyperbolicity and Sensitive Chaotic Dynamics at Homoclinic Bifurcations: Fractal Dimensions and Infinitely Many Attractors, Cambridge University Press, Cambridge, 1993.

[87] J. D. Pryce, J. K. Andreid, 1998, ADO1, a Fortran 90 code for automatic differentiation. Tech. Rep. RAL-TR-1998-057, Rutherford Appleton Laboratory, Chilton, UK. ftp://matisa.cc.rl.ac.uk/ pub/reports/prRAL98057.ps.gz.

[88] T. Puu, Nonlinear economic dynamics, fourth edition, Springer-Verlag, 1997.

[89] T. Puu, Attractors, bifurcations, and chaos, Nonlinear phenomena in economics dynamics, second edition, Springer, 2003.

[90] T. Puu, Chaos in duopoly pricing, Chaos, Solitons & Fractals, 1 (1991) 573-581.

[91] T. Puu, The Chaotic duopolists revisited, J. Econ. Behav. Organization, 33 (1998) 385-394.

[92] D. Rand, Exotic phenomena in games duopoly models, J. Math. Econ., 5 (1978) 173-184.

[93] L. C. Rich, and D. R. Hill, Automatic differentiation in MATLAB. App. Num. Math., 9 (1992) 33 - 43.

[94] D. Roose and V. Hlavaçek, A direct method for the computation of Hopf bifurcation points, SIAM J. Appl. Math., 45 (1985) 879-894.

[95] J. B. Rosser, The development of complex oligopoly dynamics theory, in: T. Puu, I. Sushko (Eds.), Text Book Oligopoly Dynamics: Models and Tools, Springer, Berlin, (2002) 31-83.

[96] R. Sacker, On invariant surfaces and bifurcation of periodic solutions of ordinary differential equations, Report IMM-NYU 333, New York University, 1964.

[97] R. Sacker, A new approach to the perturbation theory of invariant surfaces, Comm. Pure Appl. Math., 18 (1965), 717–732.

[98] P. Shil'nikov, On a Poincaré–Birkhoff problem, Math. USSR-Sb. 3 (1967) 353-371.

[99] S. Smale, Diffeomorphisms with many periodic points, In: S. Carins, ed., Differential and Combinatorial Topology, Princeton University Press, Princeton, NJ, 1963, pp. 63–80.

[100] A. Shaun, A. Forth, M. Marcus, and M. Edvall, User guide for MAD - MATLAB automatic differentiation toolbox TOMLAB/MAD, Version 1.1 The Forward Mode. TOMLAB Optimisation Inc., 855 Beech St 12, San Diego, CA 92101, USA, Jan 2004. See `http://tomlab.biz/products/mad`.

[101] A. N. Shoshitaishvili, The bifurcation of topological type of singular points of vector fields that depend on parameters, Trudy Sem. Petrovsk, no. Vyp. 1 (1975) 279-309.

[102] Song, Yuhui Lisa, The Juvenile-Adult Leslie/Gower Competition Model, preprint 2004.

[103] S. Wiggins, Global Bifurcations and Chaos, Springer-Verlag, 1988.

[104] A. Wikan, A. Eide, An analysis of a nonlinear stage-structured cannibalism model with application to the Northeast Arctic cod stock, Bulletin of Mathematical Biology, 66 (2004) 1685-1704.

[105] A. Wikan, and E. Mjølhus, Overcompensatory recruitment and generation delay in discrete age-structured population models. J. Math. Biol, 35 (1996) 195-239.

[106] Z. Xiao-Qiang, Dynamical systems in population biology, Springer-Verlag, 2003.

[107] K. Yagasaki, Numerical detection and continuation of homoclinic points and their bifurcations for maps and periodically forced systems, Internat. J. Bifur. Chaos Appl. Sci. Engrg 8 (1998) 1617-1627.

[108] Y. Zhiping, E. J. Kostelich, and J. A. Yorke, Calculating stable and unstable manifolds, Int. J. Bifur. Chaos, 1 (1991) 605–623.

[109] Y. Zhiping, E. J. Kostelich, and J. A. Yorke, Erratum: "Calculating stable and unstable manifolds", Int. J. Bifur. Chaos , 2 (1992), 215.