THE EFFECTS OF SPACE-DISCRETIZATIONS ON COMPUTING INERTIAL MANIFOLDS

by

Konstantinos E. Korontinis
B.Sc., Simon Fraser University, Burnaby, 1989

THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the Department
of
Mathematics and Statistics

SIMON FRASER UNIVERSITY
May, 1992

Approval

Name : Konstantinos E. Korontinis

Degree : Master of Science

Title of Thesis : The Effects of Space-discretizations on Computing Inertial Manifolds

Examining committee :

Chairman: *Dr. A. Lachlan*

Dr. M.R. Trummer
Senior Supervisor

Dr. R.D. Russell

Dr. T. Tang

Dr. E. Van Vleck

External Examiner
Department of Mathematics and Statistics
Simon Fraser University

Date Approved : May 21, 1992

PARTIAL COPYRIGHT LICENSE

I hereby grant to Simon Fraser University the right to lend my thesis, project or extended essay (the title of which is shown below) to users of the Simon Fraser University Library, and to make partial or single copies only for such users or in response to a request from the library of any other university, or other educational institution, on its own behalf or for one of its users. I further agree that permission for multiple copying of this work for scholarly purposes may be granted by me or the Dean of Graduate Studies. It is understood that copying or publication of this work for financial gain shall not be allowed without my written permission.

Title of Thesis/Project/Extended Essay

The effects of space-discretizations on computing inertial manifolds.

Author:

(signature)

KOKKINIS, Konstantinos

(name)

June 1st 1992

(date)

Abstract

In this thesis we investigate various aspects of the problem of computing inertial manifolds for dissipative Partial Differential Equations (PDEs). In particular, we investigate cases where only an approximate spectral decomposition of the dominant differential operator is known. We perform numerical experiments in one space dimension by discretizing the space variable and performing all of the computations on a fine grid. Furthermore, we approximate the first few of the smallest eigenvalues and eigenvectors of the discretized operator using the Lanczos algorithm. This is the initial step towards computing inertial manifolds for systems involving more than one space dimension and irregular domains.

The basic ideas and techniques are exemplified for the Kuramoto-Sivashinsky (KS) and the Reaction-Diffusion (RD) equations. These equations are chosen because they are fairly simple but their dynamics are sufficiently complicated, and they are a classical case for which a considerable amount of computational experience has been documented.

Our results are compared to those for other methods where the exact eigenvalues and eigenvectors are used. This comparison is not meant to test the competitiveness of the discrete method since, for the case tested, the other methods are expected to be better. Rather, it is a test study to show that the discrete method can be used as a general method for computing inertial manifolds.

Dedication

To Litsa, whose continuous support has been invaluable.

Acknowledgments

The completion of this thesis would not have been possible without the support of many people.

I am deeply grateful to my senior supervisor Dr. M. R. Trummer for his continuous academic support, patience and friendship from the beginning of my graduate studies. I also owe special thanks to him for trusting my abilities during the initial period of this program.

I am equally grateful to Dr. R. D. Russell whose long time friendship and encouragement played an important role in my decision to pursue graduate studies.

I also wish to express my appreciation to Dr. E. Van Vleck for his questions and comments, especially on the early drafts of this work.

Many thanks to my family and to my friends Thanasis Passias, Antonio Cabal and Roger Coroas, whose consistent moral support helped me tremendously.

I would also like to express my thanks to Sylvia Holmes and Tasoula Berggren for their assistance, and to the Department of Mathematics and Statistics of Simon Fraser University for financial support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Recently there has been considerable interest, especially in the dynamical systems community, in the computation of inertial manifolds for dissipative partial differential equations (PDEs). This process is carried out by first splitting the solution space H into two subspaces PH and QH, where P is an orthogonal projection on H with finite dimensional range and $Q = I - P$. To date, the methods used to compute inertial manifolds have dealt with cases where these projections are based on eigenfunctions that are known. This restricts the application of these methods mostly to one-dimensional problems or regular domains. Because many problems arising in practice do not fit in this framework, a method that can be adapted to higher dimensional problems or to one dimensional problems with more complicated boundary conditions is needed.

In this thesis we make a first step in this direction by developing a method that we believe can be adapted to situations involving more than one space dimension and irregular domains. In the discrete method we apply two ideas: First we discretize the space variable creating a computational grid and also, we approximate, using the Lanczos iteration, the eigenvalues of the discretized operator.

For comparison purposes, we test our method on one space dimensional equations for which exact eigenfunctions are known. These equations are the Kuramoto-Sivashinsky (KS) and the Reaction-Diffusion (RD) equations. We compare our results to two methods developed by Russell, Sloan and Trummer in [RST1], by investigating how accurately and how efficiently the discrete method describes the qualitative behaviour of the KS equation as it is demonstrated in its bifurcation diagram.

It should be pointed out that the discrete method is not designed to be competitive in cases where the exact eigenfunctions are known, but rather, to be applied to problems over irregular domains where the eigenfunctions of the operator can only be approximated. Therefore, we do expect a loss of efficiency. However, it is interesting to note that experiments indicate that the discrete method might be somewhat more robust than the semi-discrete methods used in [RST1] and [JKT1]. It is also worthwhile mentioning that the treatment of nonlinear terms is much simpler for the discrete method.

In the second chapter we discuss some of the general inertial manifold theory and give some basic definitions. In chapter three we describe briefly the existing methods to approximate inertial manifolds, while in chapter four we describe the discrete method as it is applied to the KS equation. We also describe the Lanczos algorithm, which is the method used to approximate the first m smallest eigenvalues and corresponding eigenvectors. In chapter five we discuss the numerical results that we obtained after applying the discrete method and we compare our method to the methods developed in the paper [RST1].

Chapter 2

General Theory

2.1 The General Problem

Consider the evolution equation of the form

$$u_t + Au + F(u) = 0 \tag{2.1}$$

where A is a self-adjoint positive operator with compact resolvent defined on a Hilbert space H, and F(u) is the nonlinear part defined on the domain of A.

A is usually a (dissipative) spatial differential operator (like the Laplacian, or the biharmonic operator), making (2.1) a partial differential equation (PDE).

Let $u(t) = S(t)u_0$ denote the solution to (2.1) at time t satisfying the initial condition $u(t_0) = u_0$.

Assume that the above evolution equation is dissipative, i. e., its trajectories enter and eventually remain in an absorbing ball contained in the appropriate phase space. That is, the solution map $u_0 \to u(t) = S(t)u_0$ satisfies the following property:

> There exists a compact, convex set $Y \subset H$ such that, for any bounded set $X \subset H$ there exists $t_0 = t_0(X)$ with $S(t)u_0 \in Y$ for all $t \geq t_0, u_0 \in X$.

The operator A has a complete orthogonal set of eigenfunctions $w_1, w_2, \ldots$, with corresponding eigenvalues $0 < \lambda_1 \leq \lambda_2 \leq \ldots$.

Define P to be the spectral projection onto the span of the first n eigenfunctions and $Q := I - P$ be that onto the remaining ones, i. e.,

$$Pu = p := \sum_{j=1}^{n} \alpha_j(t) w_j := \sum_{j=1}^{n} \frac{(u, w_j)}{(w_j, w_j)} w_j \tag{2.2}$$

and

$$Qu = q := \sum_{j=1+n}^{\infty} \alpha_j(t) w_j := \sum_{j=1+n}^{\infty} \frac{(u, w_j)}{(w_j, w_j)} w_j \tag{2.3}$$

where (,) denotes the scalar product on H.

2.2 Inertial Manifolds

Definition : A subset $\Omega \subset H$ is said to be an **inertial manifold** for equation (2.1) if Ω satisfies the following conditions:

1. Ω is a finite dimensional Lipschitz manifold in H
2. Ω is positively invariant, i. e., if $u_0 \in \Omega$ then $S(t)u_0 \in \Omega$ for all $t > 0$.
3. Ω is exponentially attracting, i. e., there is a $\mu > 0$ such that for every $u_0 \in H$ there is a constant $K = K(u_0)$ such that $dist(S(t)u_0, \Omega) \leq Ke^{-\mu t}$, $t \geq 0$.

Existence of an inertial manifold is very important since it implies that the dynamics of the original equation (2.1) are completely described by a finite dimensional ordinary differential equation (ODE), *with no error.*

A brief description of existence theories for inertial manifolds will come in a later section; for now we concentrate on how to obtain the reduced system of ODEs.

Assuming that the eigenvalues of A grow sufficiently fast to satisfy a gap condition [LS], it is possible to obtain an upper bound on n such that Ω is the graph of a smooth function

$$\Phi : PH \rightarrow QH.$$

Recall that PH and QH are subspaces of the solution space H with P and Q being the orthogonal projections on H as defined in the previous section.

Under the above assumptions the graph of the function Φ is an n-dimensional manifold in H.

For $u \in H$ we set $p = Pu$, $q = Qu$, and using the commutivity relations $PA = AP$ and $QA = AQ$ we can write the PDE (2.1) equivalently as the following system of ODEs

$$\dot{p} + APp + PF(p+q) = 0 \tag{2.4}$$

and

$$\dot{q} + AQq + QF(p+q) = 0. \tag{2.5}$$

By expressing the q variables in terms of the p variables through the relation $q = \Phi(p)$ we obtain the following reduced system of ODEs

$$\dot{p} + APp + PF(p + \Phi(p)) = 0, \tag{2.6}$$

which can now be used to determine the long-term dynamics of the original equation with no error. The system (2.6) is called an **inertial form** of (2.1).

2.3 Approximate Inertial Manifolds

Note that the way P and Q are defined in (2.2) and (2.3), P is an orthogonal projection on H with finite dimensional range, while Q has infinite dimensional range. In practice Q is approximated by truncating the series (2.3) and therefore equation (2.3) is replaced by

$$Qu = q := \sum_{j=1+n}^{m} \alpha_j(t) w_j := \sum_{j=1+n}^{m} \frac{(u, w_j)}{(w_j, w_j)} w_j \tag{2.7}$$

Because of this truncation, the relation between p and q is replaced by an approximation, say $\phi_a(p)$ which now maps PH into the finite dimensional space QH, and the corresponding graph Ω_a is now an approximate inertial manifold (AIM) of (2.1).

From now on it is assumed that the truncation has taken place, thus the subscript is dropped and Ω denotes an AIM.

In order to determine the long-term dynamics of the dissipative PDE (2.1), we need to compute solutions of the ODE system (2.6) and this requires computing the function ϕ. There are a number of ways to do this, some of which will be discussed below:

Galerkin approximations. The *classical* (or flat) *Galerkin* approximation involves setting the function $q = \phi(p) \equiv 0$. Then the system (2.6) becomes

$$\dot{p} + APp + PF(p) = 0 \tag{2.8}$$

which now is a system of ODEs dependent only on the variable p and can be solved using the standard techniques.

The *modified (or nonlinear) Galerkin* approximations involve a nontrivial approximation to ϕ, say $\phi_a(p)$, in equation (2.4) to get

$$\dot{p} + APp + PF(p + \phi_a(p)) = 0. \tag{2.9}$$

There are many ways to obtain approximations to ϕ which can be divided into two groups:

1. Solve a PDE for $\phi(p)$ using both equations (2.4) and (2.5)

2. Use (2.5) to find an approximation to $\phi(p)$ and then use it in (2.4).

In this thesis only methods which belong to the second group will be discussed.

The *Euler-Galerkin* approximation involves starting with the flat manifold, i. e.,

$$\Omega_0 = \{u = p + q : q = 0\} \quad \subset \quad H. \tag{2.10}$$

Now, under appropriate assumptions, it can be shown, [FST], that as $t \to \infty$ the semi-flow induced by the solution of (2.1) takes the initial manifold Ω_0 to a true inertial manifold Ω through the relation $\Omega_t = S(t)\Omega_0$. Since Ω is exponentially attracting, after a relatively short time interval τ, Ω_τ, the image under the semi-flow of Ω_0, can

be shown to be very close to Ω, [FST]. The key to this approach is to approximate the true evolution with the implicit Euler method for small time τ.

Other methods to approximate inertial manifolds include elliptic regularization and parabolic regularization. For more details on these and on the above described methods the reader can look in [LS] and in the references given therein.

2.4 Existence theories for inertial manifolds

There are three classes of existence theories for inertial manifolds.

The **Lyapunov-Perron** method is based on the variation of constants formula. This method is very useful for deriving properties of inertial manifolds and for proving existence but it does not appear to be a good method for approximating one. The reason for this is that it uses backward integration of the equation which is the unstable direction.

The second class of theories uses the **Hadamard** (or graph transform) method. The basic idea behind this method is to start with an initial approximation to the manifold, say Ω_0. Then, by letting the dynamics of the evolutionary equation act on Ω_0 we get a set Ω_t at each time $t > 0$. Under suitable hypotheses, it can be proven that each Ω_t is representable as the graph of some function, the limit

$$\lim_{t \to \infty} \Omega_t = \Omega$$

exists and that Ω is the inertial manifold.

The last class of existence theories is based on the method of **elliptic regularization** developed by Sacker [S1],[S2]. The extension of the above method to infinite dimensional dynamical systems is presented in [LS].

For more details on the above theories the reader is referred to [LS].

Chapter 3

Approximate Inertial Manifolds

In this chapter we discuss some of the methods which have been used to approximate inertial manifolds. Recall from chapter 2 the system of ODEs

$$\dot{p} + APp + PF(p+q) = 0 \tag{3.1}$$

and

$$\dot{q} + AQq + QF(p+q) = 0. \tag{3.2}$$

Our goal is to determine the relation $q = \phi(p)$ from equation (3.2) for q and then use this relation to solve the system

$$\dot{p} + APp + PF(p+\phi(p)) = 0. \tag{3.3}$$

3.1 Euler - Galerkin

This method is based on the Hadamard approach to prove existence of inertial manifolds. The basic idea behind this approach is to start with the flat manifold as the initial manifold, i. e.,

$$\Omega_0 = \{u = p+q : q = 0\} \subset H \tag{3.4}$$

and set $\Omega_t = S(t)\Omega_0$.

Then Ω_t is a subset of H and under appropriate assumptions, [CFNT1,2], it can be represented as the graph of a function

$$\Omega_t = Graph\Psi^t \tag{3.5}$$

where $\Psi^t : PH \rightarrow QH$ is a Lipschitz continuous function. Furthermore it can be shown that the limit

$$\lim_{t\rightarrow\infty} \Psi^t = \Phi$$

exists, and $\Omega = Graph\Phi$ is an inertial manifold [M-PS],[CFNT1], [FST].

More specifically, under certain assumptions and for some relatively short time interval τ , one can show that

$$\| \Psi^\tau - \Phi \|_\infty < \epsilon \tag{3.6}$$

for every $\epsilon > 0$, [LS].

The Euler-Galerkin method for approximating inertial manifolds uses the implicit Euler method to approximate Ψ^τ. In the paper [FJKST], which first introduced several of these general methods for computing inertial manifolds, the Backward Euler method with one iteration is used, taking $q = 0$ as the initial approximation.

Let (p_0, q_0) be a given initial condition and let $(p(\tau), q(\tau))$ denote the corresponding solution of the system (3.1) and (3.2) at $t = \tau$. The implicit Backward Euler approximation $(p_a(\tau), q_a(\tau))$ is given by letting $p_a(t)$ be the solution of

$$\dot{p} + APp + PF(p) = 0 \tag{3.7}$$

with $p(0) = p_0$, and then setting

$$q_a(\tau) = q_0 - \tau[AQq_a(\tau) + QF(p_a(\tau) + q_a(\tau))]. \tag{3.8}$$

(It should be noted that in the [FJKST] paper equation (3.7) was not solved but a constant value for $p_a(t)$ was assigned instead. That is, in equation (3.8) the value $p_a(\tau) = p_0$ was used. The same approach has been used thus far with the other

methods as well. A point of interest would be to see whether there would be improved results for the Backward Euler method if $p_a(t)$ was also being updated at each step).

Since the mapping $p_0 \to p = p_a(\tau)$ is a homeomorphism of PH, and since $q_0 = 0$ we get from equation (3.8)

$$q_a(\tau) + \tau AQq_a(\tau) = -\tau QF(p_a(\tau) + q_a(\tau)) \tag{3.9}$$

or

$$(I + \tau AQ)q_a(\tau) = -\tau QF(p_a(\tau) + q_a(\tau)) \tag{3.10}$$

or

$$q_a(\tau) = -\tau(I + \tau AQ)^{-1}QF(p_a(\tau) + q_a(\tau)). \tag{3.11}$$

Applying one iteration with starting value $q_a(\tau) = 0$ we get

$$q_a(\tau) = -\tau(I + \tau AQ)^{-1}QF(p_a(\tau)). \tag{3.12}$$

Thus, the approximation by the Euler - Galerkin method is $q = \Phi_a(p_0) := q_a(\tau)$; using $p_a(\tau) = p_0 = p$ this simplifies to

$$q = \phi_a(p) = -\tau(I + \tau AQ)^{-1}QF(p). \tag{3.13}$$

For the above equation the choice of the stepsize τ is not obvious. While for λ_{n+1} sufficiently large the choice $\tau = 1/\lambda_{n+1}$ can be given some theoretical justification, an illustration of the practical difficulties one can face can be found in [RST1].

3.2 The Steady AIM

Another method for approximating $\phi(p)$ due to Titi [T] is letting $\dot{q} = 0$ in equation (3.2) to obtain:

$$G(q) := AQq + QF(p + q) = 0. \tag{3.14}$$

The above equation defines the *steady approximate inertial manifold*, Ω_s, with graph $q = \phi_s(p)$.

The idea in this method is to solve the semi-explicit differential algebraic equations, (DAE), (3.3) and (3.14) to obtain the AIM. Using equation (3.14) we find an approximation to $q = \phi_s(p)$ and then we use this approximation in equation (3.3).

The above approximation gives useful results in practice as it is shown in [JKT1,2]. One reason for this is that the AIM obtained from solving the equation (3.14) contains all exact steady-state solutions involving only the first m modes, so the bifurcation diagram for the AIM gives a good representation of the steady-state solutions for the inertial manifold if one exists for these m modes [RST1].

One method used to solve equation (3.14) is the Picard iteration

$$q^{(l+1)} = -(AQ)^{-1}QF(p + q^{(l)}) \tag{3.15}$$

with starting value $q^0 = 0$, [JKT1].

Using two iterations in (3.15) gives rise to the so-called *pseudo-steady solution* and the corresponding *pseudo-steady* AIM, $\Omega_{\dot{s}}$, the graph of $q = \phi_{\dot{s}}(p)$. The latter method is used in [JKT1,2] with the specific choice $m = 4n$, where $m - n$ is the number of the q modes while n is the number of the p modes, for a quadratic nonlinearity $F(u)$. The corresponding choice for a nonlinearity of degree r is $m = nr^2$.

We can also use Newton's method to solve equation (3.14). Taking the derivative with respect to q in equation (3.14) we have

$$\frac{\partial G}{\partial q} = AQ + Q\frac{\partial F}{\partial q}(p + q). \tag{3.16}$$

The Newton iteration step is

$$[A + J]q_{new} = -QF(p + q_{old}) + Jq_{old} \tag{3.17}$$

where $J = Q\frac{\partial F}{\partial q}(p + q_{old})$.

The matrix on the left hand side of equation (3.17) is the trailing principal submatrix of dimension $m - n$ of the Jacobian of the full nonlinear system corresponding to the system of equations (3.1) and (3.2) with $\dot{p} = 0$, $\dot{q} = 0$, [RST1].

The applicability of this method depends on the structure of the Jacobian matrix. For some equations, such as the KS equation, the Jacobian has a very simple structure

and thus it is easy and cheap to calculate. But this is not always the case. For more information on the conditions required for such a property, the reader is referred to the paper [RST2].

Chapter 4

The Discrete Method

In this chapter we apply the discrete method to specific examples, namely the Reaction-Diffusion (RD) and the Kuramoto-Sivashinsky (KS) equations. We describe its implementation for the calculation of the inertial manifold and we also discuss the Lanczos method for approximating the eigenvalues and eigenfunctions of the operator A.

Most of the methods used to investigate inertial manifolds, including the ones discussed in the previous chapter, have thus far dealt with cases where the solution space is partitioned using projections based on eigenfunctions which are known exactly. This is not generally the case, especially when more than one space dimension and irregular domains are involved. To overcome the above obstacle, one needs a method where an approximation to the eigenfunctions is used. The discrete method is a first step in the development of such a method, since we have used an approximation of the eigenfunctions of the operator A. Also, the use of a discretization makes easier the calculation of the nonlinearity in the dissipative evolutionary equation at hand.

The discrete method is applied in two stages. In the first stage we approximate the first few smallest eigenvalues and corresponding eigenfunctions of A (or more precisely, a discrete approximation of A). These discrete approximations are then used in a fully

discrete scheme in the second stage to calculate the inertial manifold.

Note that for dissipative systems under the appropriate assumptions, the energy contained in the modes with high wave numbers is exponentially dissipated and asymptotically very small [JKT1]. This implies that only a few of the smallest eigenvalues are needed.

For the first stage of our method we use the Lanczos algorithm to approximate the eigenvalues and eigenfunctions of A. The reason for choosing the above algorithm is that we can calculate the first few small eigenvalues of A without having to compute all of them. This is very important especially when N, the number of the grid points, is large and the application of methods that calculate the full spectrum of the eigenvalues of A becomes very expensive and therefore impractical.

To test our method we chose the (KS) and the (RD) equations for two reasons. First they are fairly simple equations but their dynamics are sufficiently complicated and second, there has been a considerable amount of computational experience collected for both equations.

4.1 Examples considered

The equations that we applied our method to are:

The Kuramoto-Sivashinsky (KS) equation

$$u_t + 4u_{xxxx} + \theta(u_{xx} + uu_x) = 0 \tag{4.1}$$

subject to the periodic boundary conditions

$$u(x,t) = u(x + 2\pi, t) \tag{4.2}$$

and the restriction of the solution space to odd functions

$$u(x,t) = -u(2\pi - x, t).$$

The Reaction-Diffusion (RD) equation (or Chafeey-Infante equation)

$$u_t - \nu u_{xx} + u^3 - u = 0 \tag{4.3}$$

subject to the Dirichlet boundary conditions

$$u(0,t) = u(\pi,t) = 0. \tag{4.4}$$

Comparing the above equations to the general evolutionary equation

$$u_t + Au + F(u) = 0 \tag{4.5}$$

we see that for the (KS) equation

$$Au = 4u_{xxxx}$$

and

$$F(u) = \theta(u_{xx} + uu_x)$$

under the conditions

$$u(x,t) = u(x+2\pi,t),\ u(x,t) = -u(2\pi - x,t)$$

while, for the (RD) equation

$$Au = -\nu u_{xx}$$

and

$$F(u) = u^3 - u$$

under the condition

$$u(0,t) = u(\pi,t) = 0.$$

Next we show in detail how we apply the fully discrete method to the (KS) equation. First we explain our method for the flat Galerkin approach and then for the nonlinear Galerkin approach. In both cases we will assume that we have calculated the eigenvalues and eigenfunctions of the operator A through Lanczos iteration. Details on this first stage of our method will be given in a later section.

The general idea of applying the discrete method to the (KS) equation is the following:

Start off with a uniform grid with grid spacing $h = \frac{\pi}{N+1}$, and denote the eigenvalues and eigenvectors of the discretized operator A by λ_j and w_j, respectively, where the eigenvectors are of the form

$$w_j = (w_j^{(1)}, \cdots, w_j^{(N)})^T \in \mathbb{R}^N.$$

The solution $u(x,t)$ is now also approximated by a vector $u = (u_1, \ldots, u_N)^T$, where

$$u_i \approx u(ih,t) = \sum_{j=1}^{n} \alpha_j(t) w_j^{(i)}.$$

Discretizing the terms u_{xx} and u_x by centered differences we have

$$u_{xx}(ih) \approx (u_{xx})_i = \frac{1}{h^2}(u_{i-1} - 2u_i + u_{i+1}) \tag{4.6}$$

and

$$u_x(ih) \approx (u_x)_i = \frac{1}{2h}(u_{i+1} - u_{i-1}). \tag{4.7}$$

Let $v_i := (u_{xx})_i + u_i(u_x)_i$ denote the i^{th} component of the vector $\mathbf{v}$ approximating the function $u_{xx} + uu_x$. Taking the projection P we find

$$PF(p) = Pv = \sum_{j=1}^{n} \gamma_j w_j \tag{4.8}$$

where $\gamma = \gamma(\alpha)$.

Taking the derivative of the solution with respect to time t we obtain

$$u_t = \sum_{j=1}^{n} (\alpha_j)_t w_j.$$

Substituting in the original equation (KS) we get

$$\sum_{j=1}^{n} (\alpha_j)_t w_j + \sum_{j=1}^{n} \lambda_j \alpha_j w_j + \theta \sum_{j=1}^{n} \gamma_j w_j = 0$$

or

$$\sum_{j=1}^{n} [(\alpha_j)_t + \lambda_j \alpha_j + \theta \gamma_j] w_j = 0. \tag{4.9}$$

Thus, the system of the ODEs for the coefficients α_j is

$$\alpha_t = H(\alpha) \tag{4.10}$$

with

$$H_j(\alpha) = -\lambda_j \alpha_j - \theta \gamma_j. \tag{4.11}$$

It should be noted that we do not take advantage of the special form of the nonlinearity in the (KS) equation. The calculation of the inertial manifold with the fully discrete scheme proceeds as follows:

Denote by W the eigenvector matrix with jth column the eigenvector w_j corresponding to the jth eigenvalue, $j = 1, \cdots, n$.

To explain the method, since A is symmetric we assume that the eigenvectors are normalized such that $W^T W = I$. Note that in the program we have used a different normalization, $W^T W = \sigma^2 I$ where $\sigma = \sqrt{\frac{N+1}{2}}$; this was done to have w_j as a discrete approximation to the function $\sin(jx)$ to allow for comparisons with the previously used spectral approximations.

Define the solution vector $\mathbf{u}$ as

$$\mathbf{u} = W \cdot \alpha.$$

Calculate the vector $\mathbf{v}$ at each step by the relation

$$v_i = u_i (u_x)_i + (u_{xx})_i.$$

The projection $P\mathbf{v}$ of $\mathbf{v}$ onto the range of W can then be computed by

$$\gamma = W^T \cdot \mathbf{v},$$

and thus we can now calculate the right hand side of the ODE system given by (4.11).

In the nonlinear case we have to calculate the function $q = \phi(p)$, where the graph of ϕ is the pseudo-steady AIM, and then solve the equation

$$\dot{p} + Ap + PF(p + \phi(p)) = 0.$$

To calculate $q = \phi(p)$ we use the Picard iteration

$$q^{(l+1)} = -A^{-1}QF(p + q^{(l)})$$

for a given p and some initial $q = q_0$.

Set

$$W = [W_p\ W_q]$$

and

$$\alpha = (\alpha_p, \alpha_q)^T$$

where $W_p = [w_1 \cdots w_n]$, $W_q = [w_{n+1} \cdots w_m]$ with each w_j being a vector with N components, and $\alpha_p = (\alpha_1, \ldots, \alpha_n)^T$, $\alpha_q = (\alpha_{n+1}, \ldots, \alpha_m)^T$.

So the notation that we are using is the following: The subscript p indicates the first n columns of the matrix W which corresponds to $p = Pu$. The subscript q indicates the remaining $m - n$ columns, i. e., the columns from $n+1$ to m, corresponding to $q = Qu$.

Define the solution vector $\mathbf{u}$ by

$$\mathbf{u} = W \cdot \alpha = W_p \cdot \alpha_p + W_q \cdot \alpha_q = \mathbf{u}_p + \mathbf{u}_q.$$

Note that α_p, and thus $\mathbf{u_p}$, does not change during the calculation of the AIM.

For some given α_p and some initial α_q (which can be zero), we start the Picard iteration loop:

Calculate the function $v = u_{xx} + uu_x$ through a finite difference scheme to get the vector $\mathbf{v}$.

Next calculate the coefficient vector β from the relation

$$\beta = W_q^T \cdot \mathbf{v},$$

and thus, the q components of α using the relation

$$\alpha_j = -\frac{\theta}{\lambda_j}\beta_j,$$

for $j = n+1, \ldots, m$.

Next calculate the new solution vector $\mathbf{u}_q = W_q \cdot \alpha_q$ and update the solution

$$\mathbf{u} = \mathbf{u}_p + \mathbf{u}_q.$$

End the Picard loop.

Now calculate the vector $\mathbf{v} = W \cdot \gamma$ for the updated solution and obtain the first p components of γ from the relation

$$\gamma = W_p^T \cdot \mathbf{v}.$$

Finally the ODE system in term of the coefficients α is

$$H_j = -\lambda_j \alpha_j - \theta \gamma_j$$

for $j = 1, \ldots, n$.

The above system solves the equation

$$\dot{p} + Ap + PF(p + \phi(p)) = 0$$

which describes the inertial form of the original PDE.

To summarize, a general application of the discrete method is as follows:

step 1

- Calculate the eigenvalues and eigenfunctions of the matrix operator Ȧ using the Lanczos method and pass the information to step 2.

step 2

- Calculate the solution vector $\mathbf{u} = W \cdot \alpha$.
- Discretize the nonlinear term and then calculate the vector $\mathbf{v}$ to obtain the coefficient vector $\gamma = W^T \cdot \mathbf{v}$; $W \cdot \gamma$ is the projection $P\mathbf{v}$ of $\mathbf{v}$ onto the range of W.
- If calculating the nonlinear manifold find the relation $q = \phi(p)$ using a number of Picard iterations.
- Obtain the ODE system in terms of the coefficients α, that is, $\alpha_t = H(\alpha)$.

4.2 The Lanczos Method

In this section we describe the Lanczos method, a technique that can be used to solve large, sparse, symmetric eigenproblems of the form $Ax = \lambda x$. The method involves partial tridiagonalizations of the given matrix A. The advantages over other methods, such as the Householder approach, are that no intermediate full submatrices are generated and convergence to the extremal eigenvalues of A happens without needing to complete the tridiagonalization.

Suppose $A \in \mathbb{R}^{n\times n}$ is large, sparse and symmetric and assume that we want a few of its largest or smallest eigenvalues. The Lanczos method generates a sequence of tridiagonal matrices T_j with the property that the extremal eigenvalues of $T_j \in \mathbb{R}^{j\times j}$ are progressively better estimates of A's extremal eigenvalues.

There are many ways to derive the Lanczos algorithm. The way we are going to proceed is by considering the optimization of the Rayleigh quotient:

$$r(x) = \frac{x^T Ax}{x^T x} \quad , x \neq 0. \tag{4.12}$$

The maximum and minimum values of r(x) are $\lambda_n(A)$ and $\lambda_1(A)$ respectively [GV]. Suppose $\{q_i\} \subseteq \mathbb{R}^n$ is a sequence of orthonormal vectors and define the scalars M_j and m_j by

$$M_j = \lambda_n(Q_j^T AQ_j) = \max_{y\neq 0} \frac{y^T(Q_j^T AQ_j)y}{y^T y} = \max_{\|y\|_2=1} r(Q_j y) \leq \lambda_n(A) \tag{4.13}$$

$$m_j = \lambda_1(Q_j^T AQ_j) = \min_{y\neq 0} \frac{y^T(Q_j^T AQ_j)y}{y^T y} = \min_{\|y\|_2=1} r(Q_j y) \geq \lambda_1(A) \tag{4.14}$$

where $Q_j = [q_1, \ldots, q_j]$.

The algorithm is derived by considering how to generate the q_j so that M_j and m_j are increasingly better estimates of $\lambda_n(A)$ and $\lambda_1(A)$.

Suppose $v_j, u_j \in span\{q_1, \ldots, q_j\}$ are such that $M_j = r(u_j)$ and $m_j = r(v_j)$.

Since $r(x)$ increases most rapidly in the direction of the gradient

$$\nabla r(x) = \frac{2}{x^T x}(Ax - r(x)x) \tag{4.15}$$

we can have $M_{j+1} > M_j$ if q_{j+1} is determined so that

$$\nabla r(u_j) \in span\{q_1, \ldots, q_{j+1}\} \tag{4.16}$$

and similarily, since $r(x)$ decreases most rapidly in the direction of $-\nabla r(x)$ we want

$$\nabla r(v_j) \in span\{q_1, \ldots, q_{j+1}\} \tag{4.17}$$

[GV].

We need to find a q_{j+1} that satisfies the two requirements above. Since $\nabla r(x) \in span\{x, Ax\}$, (4.16) and (4.17) are satisfied if

$$span\{q_1, \ldots, q_j\} = span\{q_1, Aq_1, \ldots, A^{j-1}q_1\} \equiv \mathcal{K}(A, q_1, j). \tag{4.18}$$

Choose q_{j+1} so $\mathcal{K}(A, q_1, j+1) = span\{q_1, \ldots, q_{j+1}\}$. Thus, the problem now becomes computing orthonormal bases for the **Krylov subspaces** $\mathcal{K}(A, q_1, j)$. To do this we look at the connection between the tridiagonalization of A and the QR decomposition of the **Krylov matrix**

$$\mathcal{K}(A, q_1, n) = [q_1, Aq_1, A^2q_1, \ldots, A^{n-1}q_1]. \tag{4.19}$$

More specifically, if $T = Q^TAQ$ is tridiagonal with $Qe_1 = q_1$ then,

$$\mathcal{K}(A, q_1, n) = Q[e_1, Te_1, T^2e_1, \ldots, T^{n-1}e_1]$$

is the QR factorization of $\mathcal{K}(A, q_1, n)$.

Thus the q_j can be generated by tridiagonalizing A with an orthogonal matrix whose first column is q_1. Therefore, to compute the elements of the tridiagonal matrix $T = Q^TAQ$ directly we have:

Set

$$Q = [q_1, \ldots, q_n]$$

and

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \beta_{n-1} \\ 0 & \cdots & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

Multiply both sides of the equation

$$T = Q^T AQ$$

by Q to get

$$QT = AQ.$$

Equating columns on both sides we have

$$\beta_{j-1}q_{j-1} + \alpha_j q_j + \beta_j q_{j+1} = Aq_j \tag{4.20}$$

with $\beta_0 q_0 \equiv 0$ for $j = 1, \ldots, n-1$.

The orthonormality of the q_j implies

$$\alpha_j = q_j^T A q_j.$$

Also if

$$r_j = (A - \alpha_j I)q_j - \beta_{j-1}q_{j-1}$$

is nonzero, then $q_{j+1} = r_j/\beta_j$, where $\beta_j = \pm \parallel r_j \parallel_2$.

Organizing the above information we get the **Lanczos iteration:**

$r_0 = q_1$ starting vector
$\beta_0 = 1$
$q_0 = 0$
$j = 0$
while $\beta_j \neq 0$
  $q_{j+1} = r_j/\beta_j$
  $j = j + 1$
  $\alpha_j = q_j^T A q_j$
  $r_j = (A - \alpha_j I)q_j - \beta_{j-1}q_{j-1}$
  $\beta_j = \parallel r_j \parallel_2$
end.

The q vectors are called the **Lanczos vectors.**

Several mathematical properties of the method are summarized in the following theorem, the proof of which is given in [GV].

Theorem: Let $A \in \mathbb{R}^{n\times n}$ be symmetric and assume $q_1 \in \mathbb{R}^n$ has unit two-norm. Then the Lanczos iteration runs until $j = m$, where $m = rank(\mathcal{K}(A, q_1, n))$. Moreover, for $j = 1, \ldots, m$ we have

$$AQ_j = Q_j T_j + r_j e_j^T$$

where

$$T_j = \begin{bmatrix} \alpha_1 & \beta_1 & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \beta_{j-1} \\ 0 & \cdots & \beta_{j-1} & \alpha_n \end{bmatrix}$$

and

$$Q_j = [q_1, \ldots, q_j]$$

has orthonormal columns that span $\mathcal{K}(A, q_1, j)\square$.

If the Lanczos vectors are not orthonormal then they do not generate the correct T matrix. Also, if $\beta_j = 0$ for some j then the iteration breaks down but not without giving us some important information. If $\beta_j = 0$ then we have calculated an exact invariant subspace and the eigenvalues of T are the eigenvalues of A, [W].

We can write the tridiagonal eigenvalue problem generated by the Lanczos method in the form

$$T_j s = \theta s \tag{4.21}$$

where the θ's are called *Ritz values*. The Ritz values will approximate the eigenvalues of A, especially the extremal ones. Approximations to the eigenvectors of A, called *Ritz vectors*, can be calculated using the equation

$$y_i = Q_j s_i$$

where y_i is the Ritz vector corresponding to s_i. These Ritz vectors satisfy

$$Ay_i = y_i\theta_i + r_j s_j(i) \tag{4.22}$$

[GV].

In the examples we have used, and specifically for the KS equation , the matrix operator is as follows:

Discretize the term u_{xxxx} by centered differences to obtain

$$u_{xxxx}(ih) \approx (u_{xxxx})_i = \frac{1}{h^4}(u_{i+2} - 4u_{i+1} + 6u_i - 4u_{i-1} + u_{i-2}), \tag{4.23}$$

where $h = \frac{\pi}{N+1}$.

Applying the boundary conditions on the above discretization we have:

$$u^{(1)}_{xxxx} = \frac{1}{h^4}(u_3 - 4u_2 + 6u_1 - 4u_0 + u_{-1})$$

$$u^{(2)}_{xxxx} = \frac{1}{h^4}(u_4 - 4u_3 + 6u_2 - 4u_1 + u_0)$$

$$\vdots$$

$$u^{(N-1)}_{xxxx} = \frac{1}{h^4}(u_{N+1} - 4u_N + 6u_{N-1} - 4u_{N-2} + u_{N-3})$$

$$u^{(N)}_{xxxx} = \frac{1}{h^4}(u_{N+2} - 4u_{N+1} + 6u_N - 4u_{N-1} + u_{N-2}).$$

Since u is periodic and odd we have $u_{-1} = -u_1$, $u_0 = u_{N+1} = 0$ and $u_{n+2} = -u_N$. Applying this we get

$$u^{(1)}_{xxxx} = \frac{1}{h^4}(5u_1 - 4u_2 + u_3)$$

$$u^{(2)}_{xxxx} = \frac{1}{h^4}(-4u_1 + 6u_2 - 4u_3 + u_4)$$

$$u^{(3)}_{xxxx} = \frac{1}{h^4}(u_1 - 4u_2 + 6u_3 - 4u_4 + u_5)$$

$$\vdots$$

$$u^{(N-1)}_{xxxx} = \frac{1}{h^4}(u_{N-3} - 4u_{N-2} + 6u_{N-1} - 4u_N)$$

$$u^{(N)}_{xxxx} = \frac{1}{h^4}(u_{N-2} - 4u_{N-1} + 5u_N).$$

Therefore, the discrete version of the operator A becomes an $N \times N$ matrix which we again call A:

$$A = \frac{4}{h^4}\begin{bmatrix} 5 & -4 & 1 & 0 & 0 & 0 & \cdots & 0 \\ -4 & 6 & -4 & 1 & 0 & 0 & \cdots & 0 \\ 1 & -4 & 6 & -4 & 1 & 0 & \cdots & 0 \\ 0 & 1 & -4 & 6 & -4 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \cdots & \vdots \\ 0 & \cdots & \cdots & 0 & 0 & 1 & -4 & 5 \end{bmatrix}.$$

When we applied Lanczos algorithm to the matrix A the initial results were not very good. Because the relative spacing of the eigenvalues of A is fairly even throughout the spectrum, the Lanczos algorithm approximated eigenvalues over the whole spectrum of A. Since our intention is to approximate a few of the smallest eigenvalues of A, we applied the algorithm to the inverse of A. The eigenvalues of A^{-1} are distributed in a different way. The relative spacing of the largest eigenvalues of A^{-1} is large, whereas the smaller eigenvalues of A^{-1} are clustered close to 0. Therefore, the Lanczos method approximates the large eigenvalues of A^{-1}, i. e., the small ones of A, very well. (Note that A^{-1} is no longer sparse).

In our program we did not actually calculate A^{-1} directly, but we used the Cholesky decomposition. According to the Cholesky decomposition theorem if A is positive definite then, it can be decomposed into a product

$$A = GG^T$$

where G is lower-triangular, in exactly one way. Since A is positive definite, we can use the above decomposition to obtain the matrix-vector product required in Lanczos algorithm. The banded structure of A carries over to G.

Recall the two steps of the Lanczos algorithm

$$\alpha_j = q_j^T A q_j$$

and

$$r_j = Aq_j - \alpha_j q_j - \beta_{j-1} q_{j-1}.$$

Applying A^{-1} instead of A we have

$$\alpha_j = q_j^T A^{-1} q_j$$

and

$$r_j = A^{-1} q_j - \alpha_j q_j - \beta_{j-1} q_{j-1}.$$

The idea is to substitute the matrix-vector product $A^{-1}q$ by a vector x satisfying $x = A^{-1}q$ or equivalently

$$Ax = q \tag{4.24}$$

and then use forward and backward substitution to solve the system of equations (4.24). Analytically this is done the following way:

Let $A = GG^T$ be the Cholesky decomposition of A. The system (4.24) then becomes

$$GG^T x = q.$$

Let $y = G^T x$ to get

$$Gy = q.$$

Since G is lower-triangular we can solve for y by forward substitution. Once we have y, we can solve the upper-triangular system $G^T x = y$ for x by backward substitution. After completing the process we have effectively obtained $x = A^{-1}q$.

One of the biggest drawbacks with the Lanczos algorithm is the problem of "ghost" or "spurious" eigenvalues. These are multiple eigenvalues of T_j that correspond to simple eigenvalues of A. They arise when orthogonality of the q_j vectors is lost, and the iteration restarts itself. Thus, while in theory the process should terminate after $n-1$ steps, in practice it continues until we stop it. Some people are using this idea to calculate not only a few of the extremal eigenvalues but also, the complete spectrum. It has been observed that the huge tridiagonal matrix T_j has eigenvalues that approximate all eigenvalues of A. The matrix T_j ($j \gg n$) has many more eigenvalues than A where, the extra eigenvalues are concentrated in tight clusters that approximate the eigenvalues of A [W].

The way we tried to identify the "spurious" eigenvalues generated by Lanczos algorithm was to calculate the difference

$$\| Aw - \lambda w \|_2$$

and keep the eigenvalues and eigenvectors that satisfied a specific tolerance. The results were slightly inconsistent and further investigation is required. Interestingly enough, even when we used spurious eigenvalues and eigenvectors in our inertial manifold calculations, the bifurcation diagrams were often hardly affected by the inclusion of spurious eigenpairs, (see figure 6a).

For more information on how to separate the ghost eigenvalues from the actual ones see [CW], [KP1,2] and [PR].

Chapter 5

Numerical Results

In this chapter we apply the discrete method to the (KS) equation and compare our results with those obtained by other methods. Rather than presenting detailed computations of the AIMs themselves, we concentrate on presenting bifurcation diagrams obtained via our new discrete method and the previously used "semidiscrete" methods. All of the computations use an improved version of AUTO, a FORTRAN code to compute bifurcation diagrams for ODEs. AUTO has been developed by Doedel [D], and modified by Liu [LL]. The runs are made on a SUN SPARC station in double precision (roughly 16 decimal digits).

The bifurcation diagrams show plots of the bifurcation parameter θ versus the L^2-*norm* of the solution. Solid lines correspond to stable steady-state solutions, broken lines correspond to unstable ones. White squares denote steady-state bifurcation points, black squares are Hopf bifurcation points, while period doubling points are marked with two triangles.

The methods that we compare the discrete method to are the ones developed in the paper [RST1] and we will denote them by RST-Newton and RST-Picard. RST-Newton uses the Newton method to solve the equation

$$G(q) := Aq + QF(p+q) = 0 \tag{5.1}$$

while RST-Picard solves the above equation using a fixed number of Picard iterations (for more information on these methods please refer to chapter 3).

We look at how accurate the results given by the discrete method are when compared to the two methods mentioned above, and we also look at the CPU time as a measure of the efficiency of the discrete method. We did the computations over the parameter range $\theta \in [0, 70]$, using the default parameter values set in AUTO.

We start our computations by looking at the steady-state solutions and then we proceed by examining the periodic solutions.

5.1 Steady-State Solutions

We start by computing the "flat" manifold using the flat Galerkin method with 12 modes. The results are shown in figures 1*a* and 1*b*. Shown in figure 1*a* is the "exact" bifurcation diagram obtained by using the exact eigenvalues and eigenvectors, while in figure 1*b* we have the bifurcation diagram obtained using the discrete method.

As indicated in the two figures, the bifurcation diagrams are qualitatively correct up to $\theta = 60$. For the values of $\theta \in [60, 70]$ the two diagrams differ at two points. The bifurcation point (BP) located at $(64.57, 2.56)$ is not found when the discrete method is applied and also, the unstable steady-state solution branch emanating from the BP located at $(64.22, 1.81)$ is not computed when we applied the flat-Galerkin method developed in [RST1]. The CPU time used to calculate the diagrams is shown in table 5.1, where $CPU_u :=$ number of seconds of CPU time devoted to the user's process. The CPU time shown is only a rough indication of the relative performance of each method. The reason for this is that often, in order to compute the bifurcation diagrams we have to restart the computations at different bifurcation points and switch branches. When this happens, AUTO, along with the new information, recalculates a lot of the data obtained in previous runs, thus giving an "inflated" CPU time. Since every method was restarted a different number of times, we decided to report the relative performance of each method given by the CPU time calculated from the run

that produced most of the data.

	Discrete method	RST-methods
CPU_u	729.9	238.8

Table 5.1: CPU times for the Flat-Galerkin methods.

In figures $2a$ and $2b$ we show the discrete method vs the RST-Newton and RST-Picard methods using $p+q=5+5$ modes and a fixed number of Picard iterations, $its=6$. (The bifurcation diagrams for these two methods for the case where $p+q=5+5$ modes are identical).

The most noticable difference between these diagrams is again the absence of the bifurcation point located at $(64.57, 2.56)$. In fact, in all the bifurcation diagrams computed with the discrete method this BP is absent. Also, in the diagrams created by the RST-methods we could not compute the unstable steady-state solution branch emanating from the bifurcation point located at $(64.22, 1.81)$. It should be noted however, that only default parameter values were used in AUTO. Often bifurcation points not found when using these parameter values can be found when re-adjusting the parameters values. The computation of this branch is shown in the diagrams in the paper [RST1]. Comparison of the CPU times is given in table 5.2.

	Discrete	RST-Newton	RST-Picard
CPU_u	1,152.4	683.6	106.1

Table 5.2: CPU time for the Non-linear Galerkin with $p+q=5+5$ modes and $its=6$ Picard iterations.

In figures $3a$ and $3b$ we show bifurcation diagrams obtained from the discrete method using $p+q=4+4$ and $p+q=6+6$ modes respectively, with $its=6$ Picard iterations. The CPU time for these calculations is given in table 5.3; the data suggest that the computational effort for computing diagrams with $n+n$ modes increases approximately linearly with n.

In all of the results above it is shown that the bifurcation diagrams computed by the discrete method are very good approximations to the diagrams computed using

	6 + 6 modes	4 + 4 modes
Discrete method	1,416.7	740.4

Table 5.3: CPU time for the Discrete method with $its = 6$ iterations.

the methods RST-Newton and RST-Picard. The bifurcation points themselves are very well approximated, with the worst case still having two digits of accuracy. For example, while the "exact" trivial solution has pitchfork bifurcations at the values of $\theta = 4, 16, 36, 64$, (these are the exact eigenvalues), in the discrete case these values are $\theta = 3.99979, 15.99684, 35.98397, 63.94933$ (these are the eigenvalues of the discretized operator, accurate to $O(N^{-2})$). The accuracy of these points increases as we increase N, the number of grid points.

So far we have shown that the discrete method produces good results, but we have not discussed its efficiency. Since in the discrete method we discretize the problem in space and we have computations on a reasonably fine grid, one would expect to have a less efficient method when compared to analytic methods which use the exact eigenvalues and eigenvectors.

Looking at the results in tables 5.1 and 5.2 , we see that the discrete method uses more computer time than the other two methods. Fortunately, the computer time used is not very large, especially when computing steady-state solutions, to make the application of the discrete method impractical. The difference among these methods becomes more significant when we compute periodic solutions which is discussed next.

5.2 Periodic Solutions

In this section we compute periodic solutions branching from Hopf bifurcation points (HP). In the computations we concentrate on two Hopf bifurcation points, located near $(34.2, 2.3)$ and $(30.3, 6.2)$ in the $(\theta, L^2 - norm)$ plane. We again use the default parameter values in AUTO. The results are limited to comparison of the non-linear Galerkin methods with $p + q = 5 + 5$ modes and $its = 6$ Picard iterations.

The solution branches off of the two Hopf bifurcation points are shown in figure 4*a*, while finer detail of the periodic solutions generated from the lower Hopf bifurcation point is shown in figure 4*b*. One period doubling is found and the corresponding periodic solution branch is also computed leading into another period doubling. This second period doubling has not been verified by the RST-Newton or RST-Picard methods.

Following the solution branch off of the top Hopf bifurcation point and following the branch of that point, we find a period doubling. The corresponding periodic solution branch is also computed and leads to another period doubling (not shown in this figure here, but calculated in previous computations with $p + q = 6 + 6$ and $p + q = 4 + 4$ modes). These results are consistent with the results found by the application of the RST-Newton and RST-Picard methods as they are shown in [RST1]. (With the parameter values used we could not reproduce the finer details calculated when we follow the branches off the bifurcation point located at $(32.79, 4.90)$).

It is important to note that the results can quickly become sensitive and bifurcation points can be missed or "spurious" ones can appear, and a more detailed study is required in order to take advantage of changing parameter values in AUTO, such as, setting smaller tolerances or increasing the number of subintervals for spline collocation. For more information and more details on AUTO please look at the description of the method by Doedel, [D]. The CPU time needed for these calculations is shown in table 5.4.

	Discrete	RST-Newton	RST-Picard
CPU_u	5,999.7	1,778.8	389.6

Table 5.4: CPU time for periodic solutions with $p + q = 5 + 5$ modes and $its = 6$ Picard iterations.

As we can see from table 5.4, the amount of computer time increases dramatically when we compute periodic solutions, especially when we follow branches emanating from period doubling points. The increase of CPU time that occurs when we apply the discrete method leads us to try and find ways to make the discrete method more

efficient. One such way is, when solving equation (5.1), instead of using $q = 0$ as an initial approximation in each step, to use the solution q from the previous parameter value θ as an initial approximation to q. This leads us to the next section where we discuss the discrete method using a variable number of Picard iterations.

5.3 Variable Number of Picard Iterations

In this section we discuss the results obtained after applying the discrete method to the (KS) equation using the non-linear Galerkin approach with $p + q = 5 + 5$ modes, with the following variation:

When we solve equation (5.1), instead of using at each step the initial approximation $q = 0$ to start the Picard iteration, we use the solution q obtained at the previous parameter value θ. We let the Picard iteration continue either until it reaches a maximum number of iterations previously set, or until the relative difference satisfies a given tolerance TOL, i. e., until

$$\frac{\| q_{new} - q_{old} \|}{\| q_{new} \|} \leq TOL.$$

Using the previous value of q instead of $q = 0$ as an initial approximation reduces the number of Picard iterations needed, especially when we follow the same branch. The reason for this is that the values of q change very slowly when we are at the same branch, so the previous q offers a much better initial approximation for the Picard iteration for the new value of θ than $q = 0$. This way we need very few Picard iterations to satisfy a given tolerance. The only time we might use the maximum number of Picard iterations is when we switch branches and the values of q differ a lot.

In our computations we use as a tolerance the value $TOL = 1.E - 10$. This value was obtained by trial and error. We tried various tolerance values applied to different number of modes. For instance, when we used $p + q = 6 + 6$ modes we got fairly good results with the tolerance set at $1.E - 8$, while when we used $p + q = 4 + 4$ modes

	$Tol = 1.E-10$	$Tol = 1.E-8$	$Tol = 1.E-6$
$p+q = 5+5$ modes	1,549.2	1,111.1	425.4
$p+q = 6+6$ modes	1,598.7	1,419.3	790.2

Table 5.5: CPU time for the Discrete method with variable Picard iterations.

a tolerance of the order $1.E-10$ was needed to get some of the results. The CPU time was also dependent upon the tolerance used. For tolerances of the order 10^{-4} or 10^{-6} the CPU time was reduced considerably but at the same time the results were not very satisfying. Further study is required in order to find the connection between the parameter values set in AUTO and the value used for the tolerance in the Picard iteration.

In figures 5*a* and 5*b* we show some of the calculations performed on the non-linear Galerkin case with $p+q = 5+5$ modes and $TOL = 1.E-10$. In figure 5*b* we show finer detail of the periodic solutions generated from the top Hopf bifurcation point and of the two period doublings calculated when we followed the corresponding branches.

The calculations of the periodic solutions generated from the bottom Hopf bifurcation point give the same results that we got when we applied the discrete method to the non-linear Galerkin case with $p+q = 5+5$ modes and $its = 6$ Picard iterations (not shown here). The CPU time used to calculate the two cases is shown in table 5.5.

Notice that if we compare the values obtained for the CPU time from tables 5.2 and 5.5, the discrete method with variable Picard iterations seems to be more expensive than the one using a fixed number of Picard iterations. It should be pointed out that when we used the discrete method with a variable number of Picard iterations we got all the information without having to restart AUTO, while when we used the fixed Picard iterations we needed two more additional runs to complete the bifurcation diagram. In a similar fashion, looking at table 5.5, the CPU time needed to do the computations with the $p+q = 6+6$ modes seems to be very comparable to the $p+q = 5+5$ case, but additional runs were required in order to complete the bifurcation diagram.

It should be noted that the Picard method with a *variable* number of iterations can also be applied to the analytic methods used to compute inertial manifolds. However, the savings will not be as dramatic as they are for the discrete method.

In figure 6a we show the bifurcation diagram obtained using $p + q = 6 + 6$ modes and $its = 6$ fixed Picard iterations for the non-linear Galerkin method without eliminating all the "spurious" eigenvalues and eigenvectors, while in figure 6b we show the bifurcation diagram obtained when we use $N = 64$ grid points for the $p + q = 5 + 5$ modes and $its = 6$ Picard iterations. As we can see the qualitative behaviour of the solution has not been altered dramatically, except for the fact that some of the bifurcation points were not computed with the AUTO default parameter values. (We were able to compute some of the missing bifurcation points when we changed some of the parameter values in AUTO).

Finally, in figure 7 we show the bifurcation diagrams that we obtained when we applied the discrete method to the Reaction-Diffusion equation. In figure 7a we show the flat Galerkin approach using 12 modes ,while in figure 7b we show the nonlinear case using $p + q = 5 + 5$ modes with $its = 6$ Picard iterations. Because the RD equation describes a gradient system we can only have steady-state solutions thus, no Hopf bifurcation points are detected.

5.4 Conclusions

During the last few years the study of inertial manifolds for dissipative PDEs has been an active area of research. To date, investigations of inertial manifolds have dealt with cases where the solution space is partitioned using projections based on eigenfunctions which can be evaluated exactly and also where the numerical techniques for their computation have been limited to a fixed number of cycles of a Picard iteration.

In this thesis a first step has been taken towards a different direction. We have developed a method to compute approximate inertial manifolds under the assumption that the exact eigenfunctions are not known. Instead, the space variable is discretized

and all of our computations are carried out on a grid. Furthermore, we have shown how to approximate the first few eigenvalues and eigenvectors of the discrete operator efficiently using the Lanczos iteration. Using this approach, we showed that AUTO still produces accurate bifurcation diagrams. The discrete method is not designed to be competitive with methods developed to compute inertial manifolds when the exact eigenfunctions are available, but rather, to be applied in more general cases where the eigenfunctions can only be approximated.

Before we can apply our method to cases involving more than one space dimension and/or irregular domains, we believe that further investigation is needed in order to computationally improve some aspects of the method. For instance, more research is needed to find a more efficient and reliable way to identify the "spurious" eigenvalues and eigenvectors produced by the Lanczos algorithm. It should be noted that an implementation in higher dimension is by no means a straightforward generalization of our approach. For example, a non-uniform grid can lead to non-symmetric matrices. Also, more careful study is required in order to find the best parameter values in AUTO that combined with the tolerance value given for the Picard iteration will give the best possible results. Early experiments with the above ideas have led us to believe that we can optimize the efficiency of the discrete method and therefore, use it to compute inertial manifolds for more complicated dissipative partial differential equations.

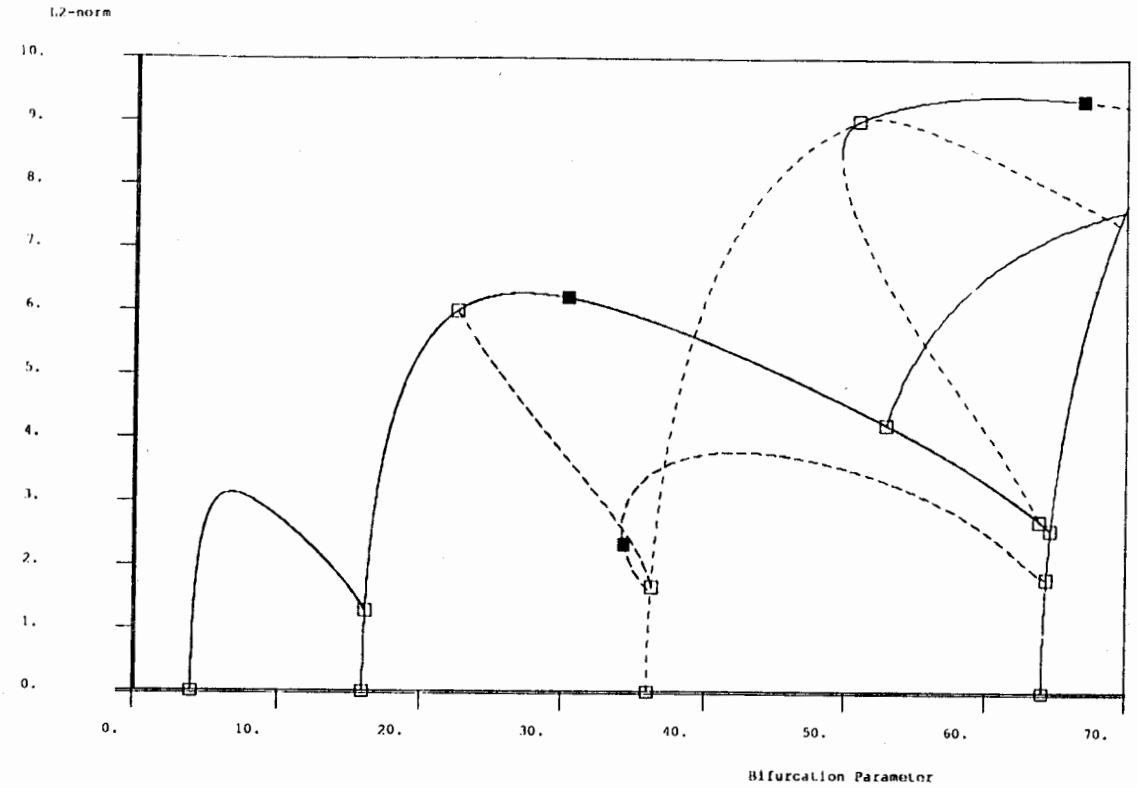

Figure 1a: 12 modes, Flat-Galerkin, RST

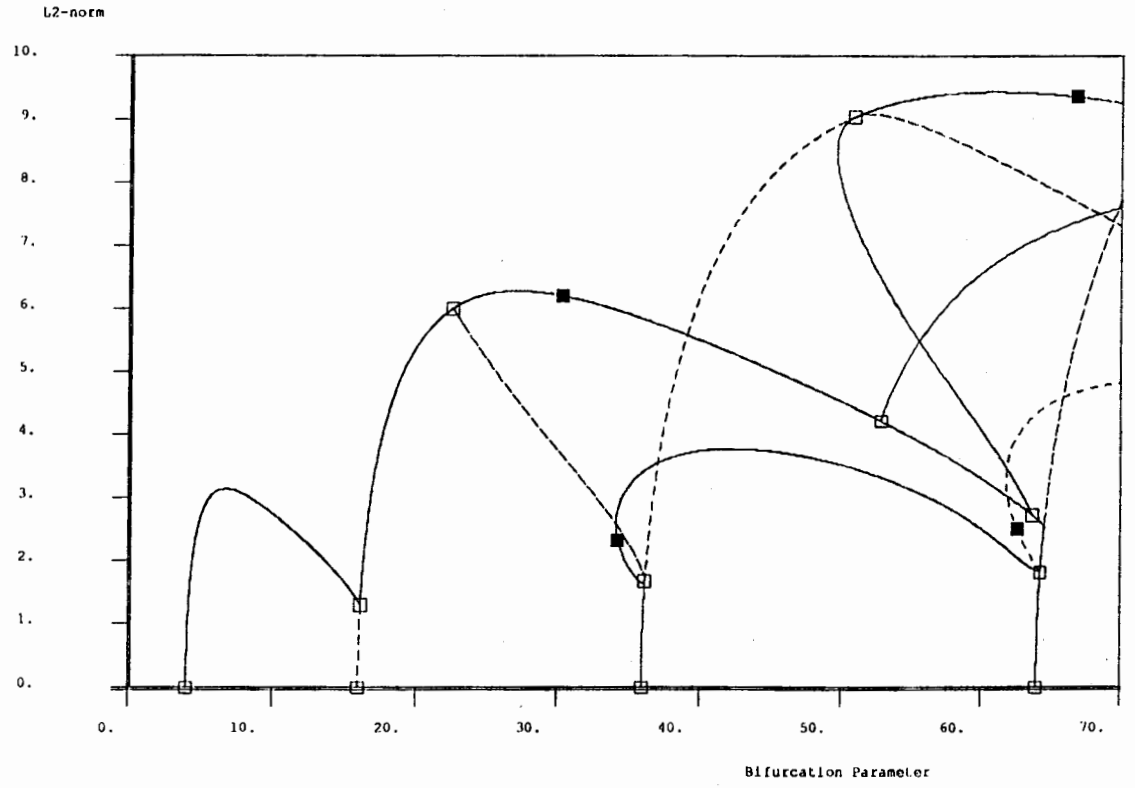

Figure 1b: 12 modes, Flat-Galerkin, Discrete

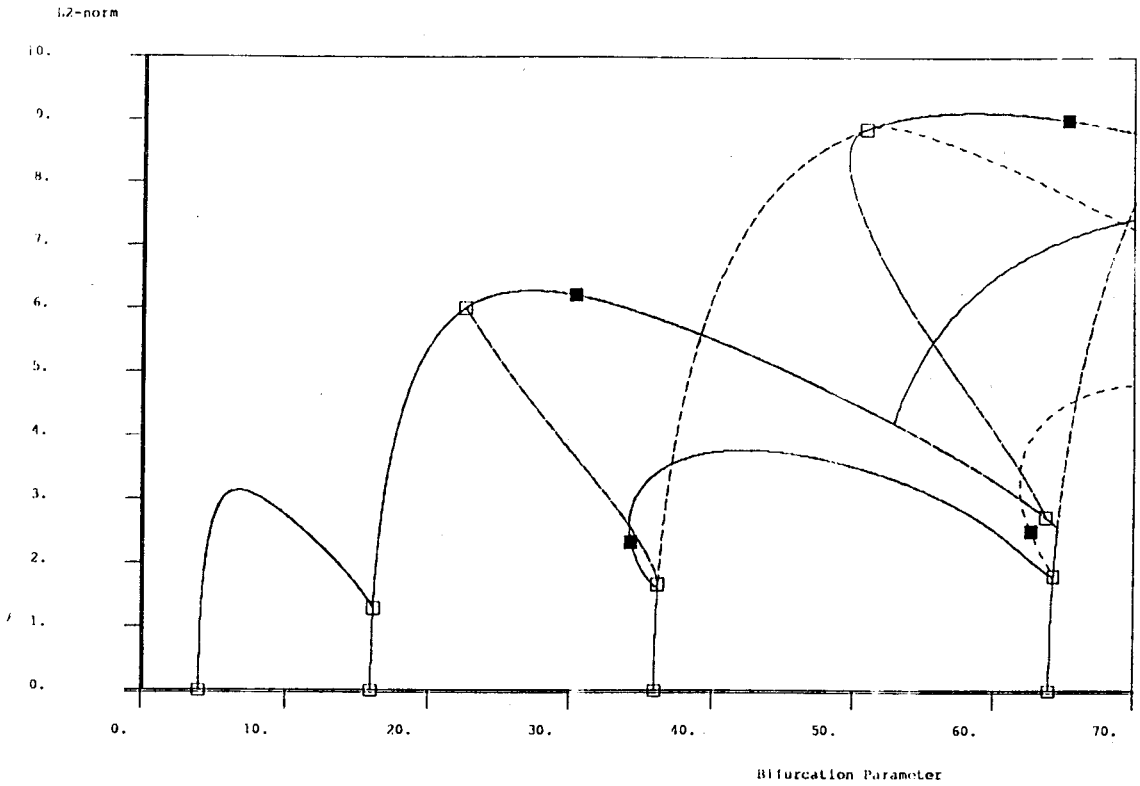

Figure 2a: 5+5, Nonlinear, Discrete, $its = 6$

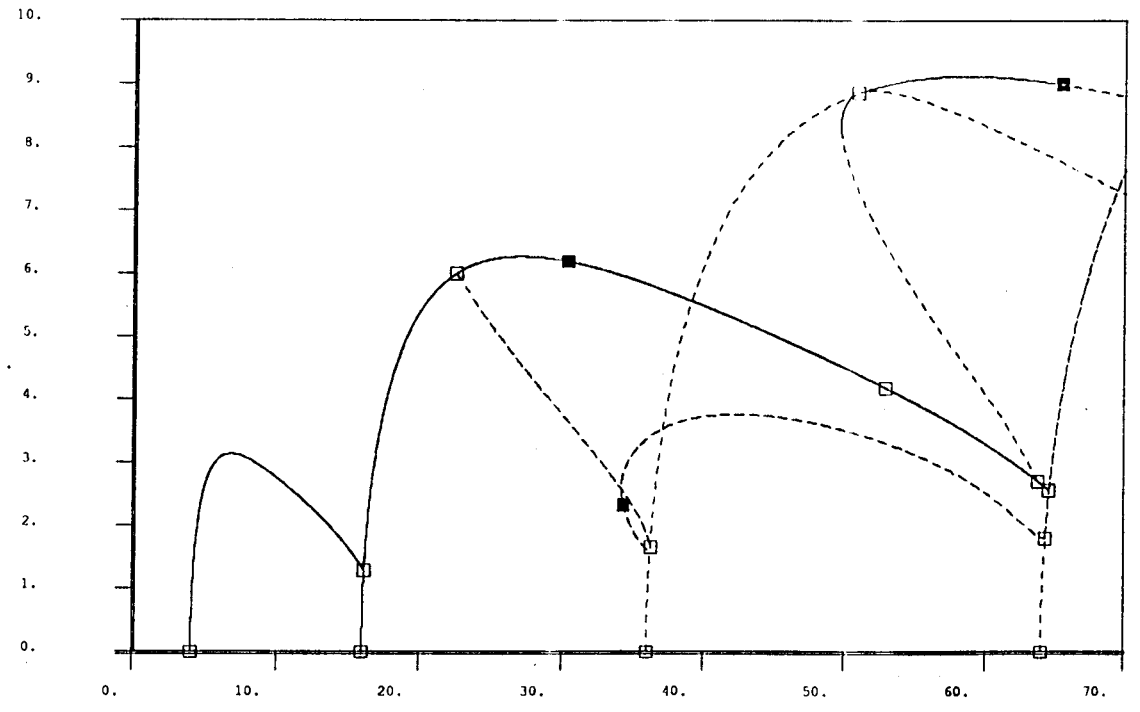

Figure 2b: 5+5, Nonlinear, RST-Newton, $its = 6$

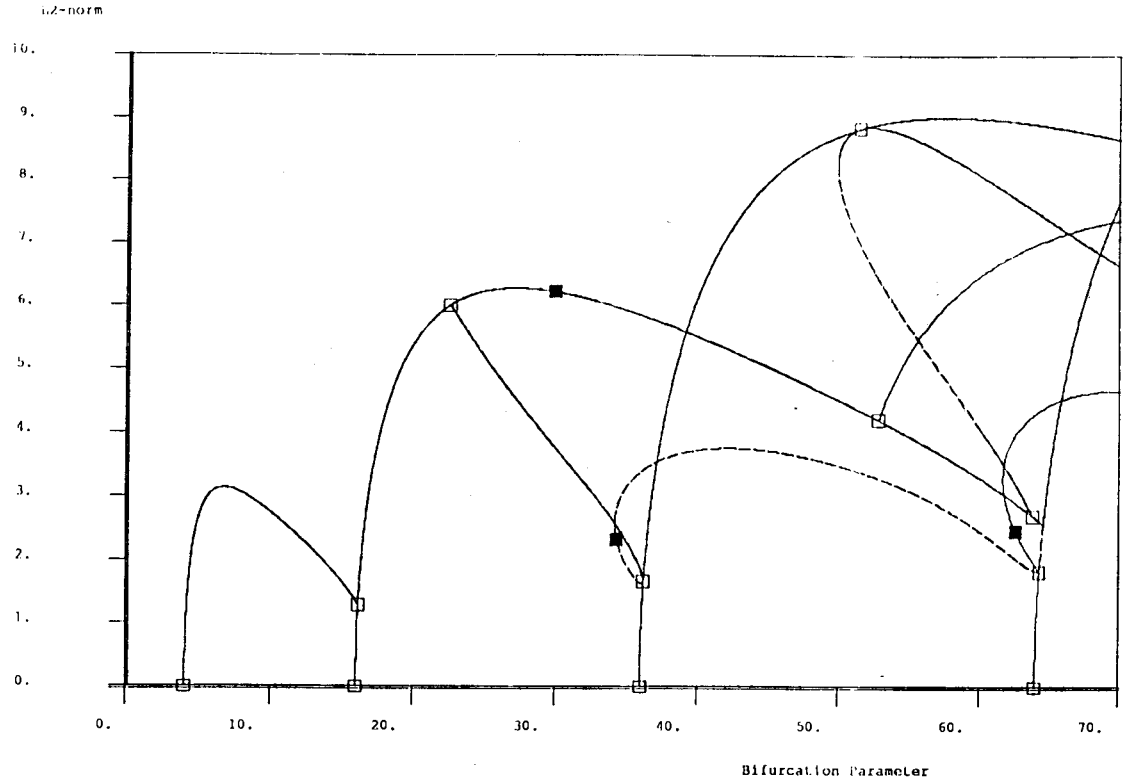

Figure 3a: 4+4, Nonlinear, Discrete, $its = 6$

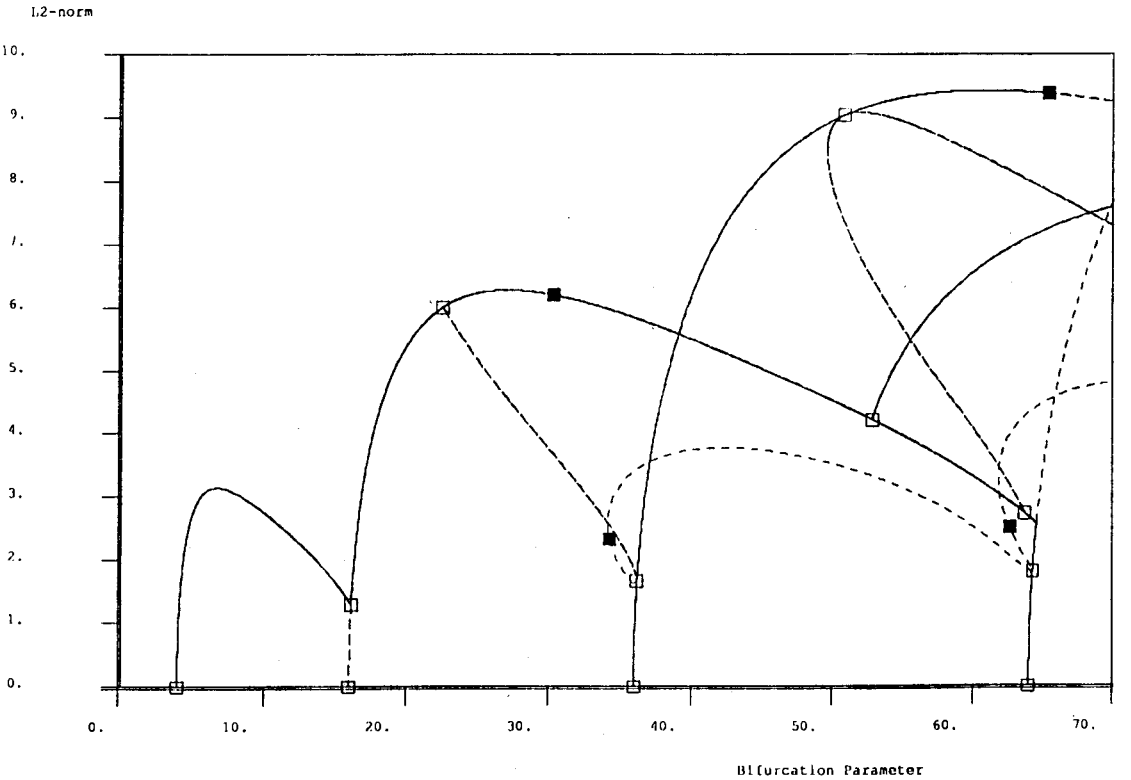

Figure 3b: 6+6, Nonlinear, Discrete, $its = 6$

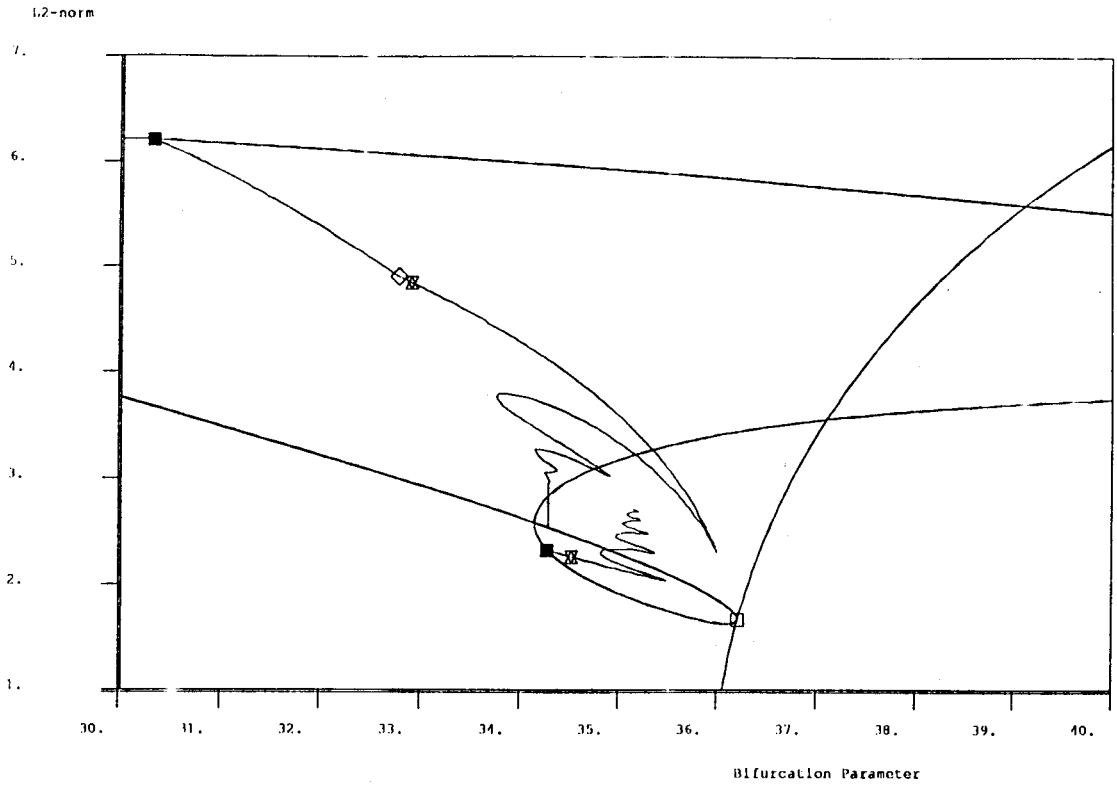

Figure 4a: 5+5, Periodic, Discrete, $its = 6$

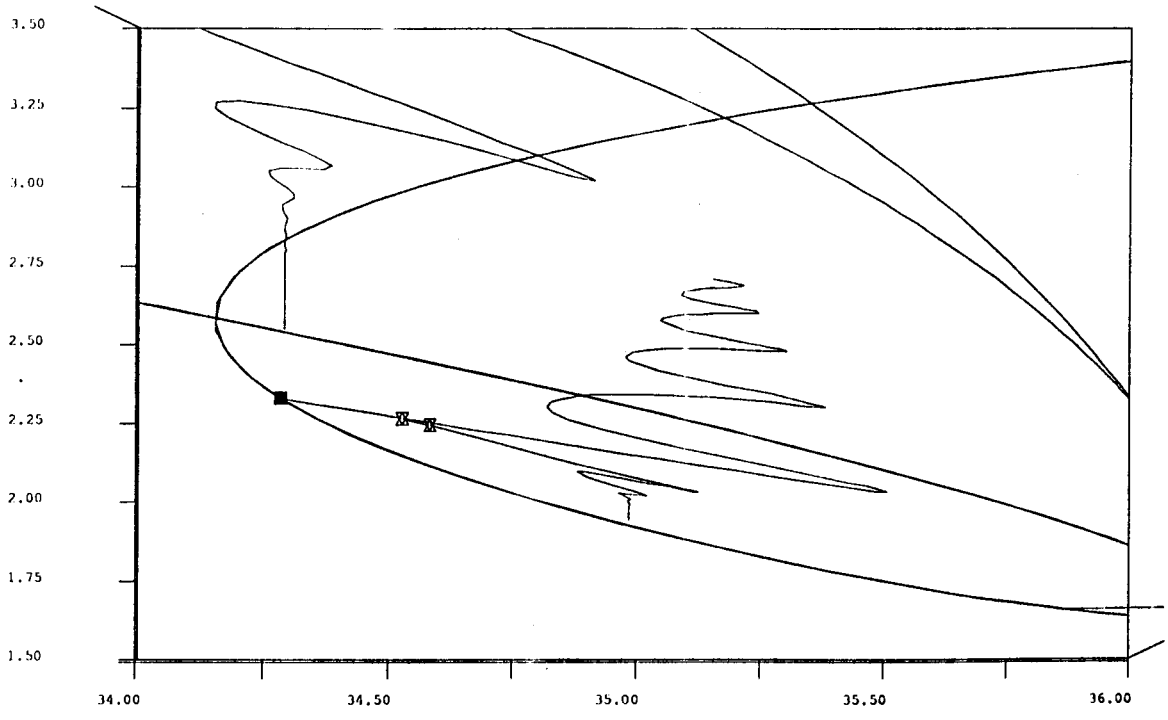

Figure 4b: 5+5, Periodic, Discrete, $its = 6$

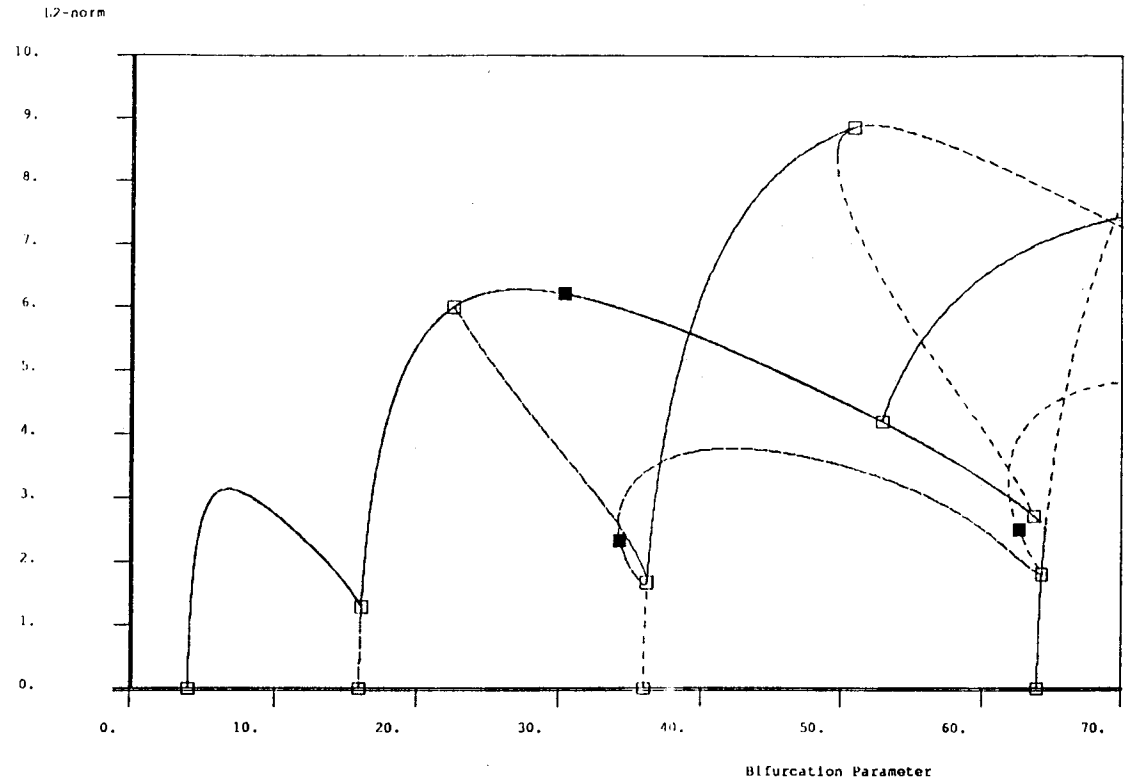

Figure 5a: 5+5, Variable Picard iterations

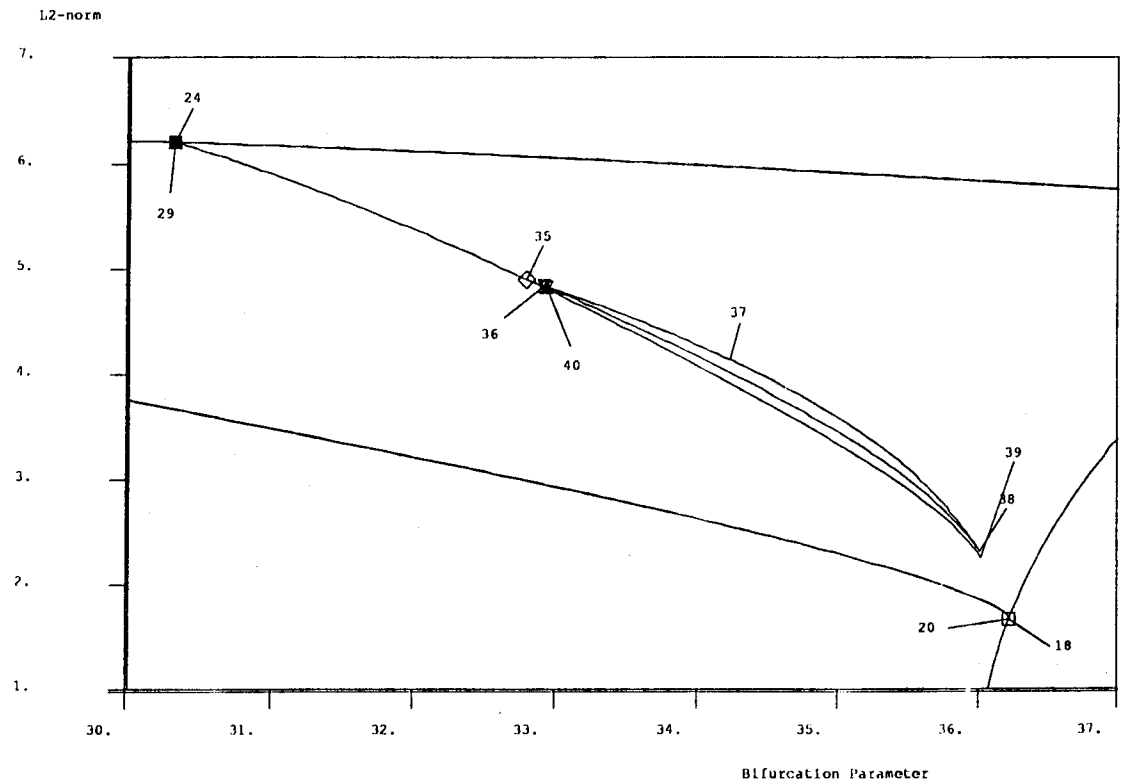

Figure 5b: 5+5, Variable Picard , Periodic

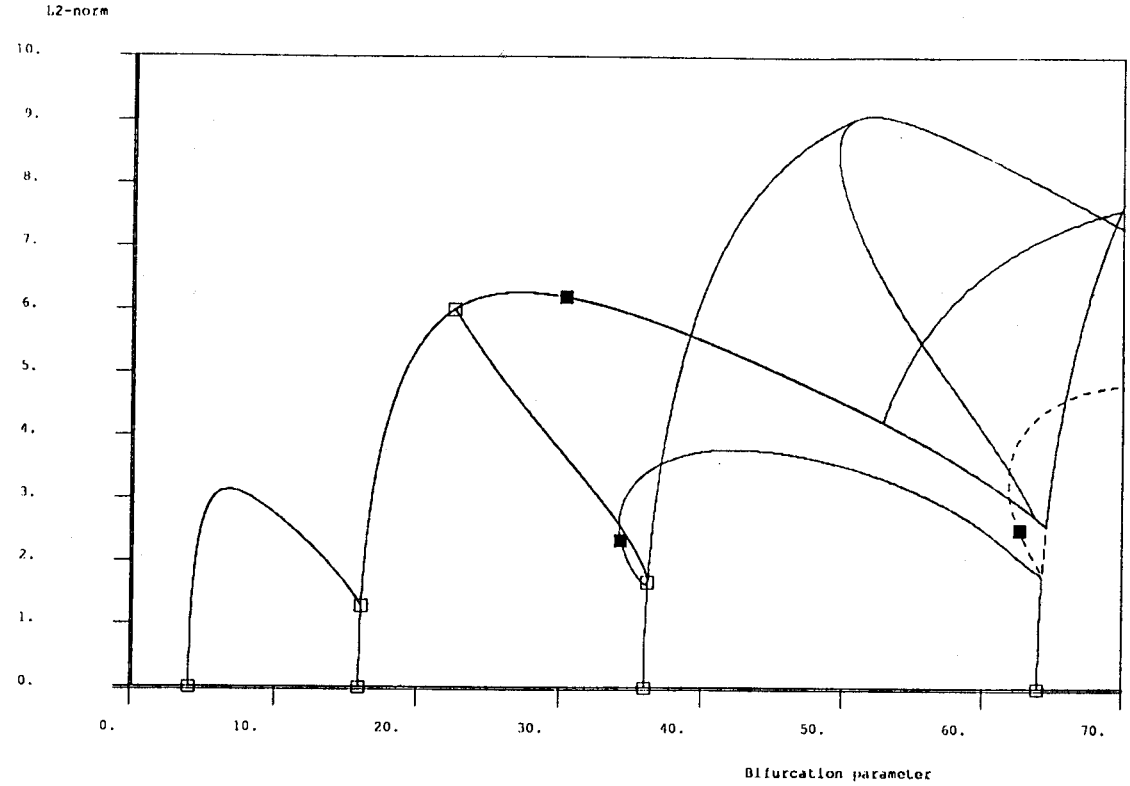

Figure 6a: 6+6, Discrete, including "spurious"

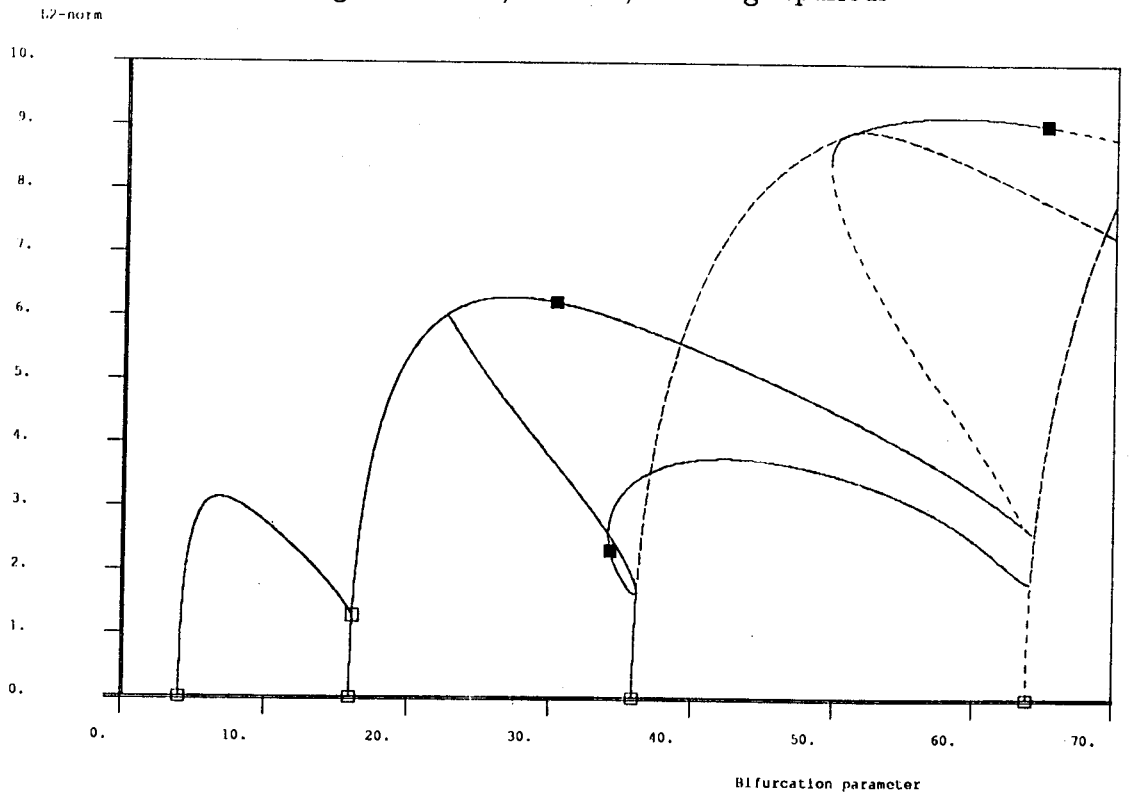

Figure 6b: 5+5, Discrete, $N = 64$

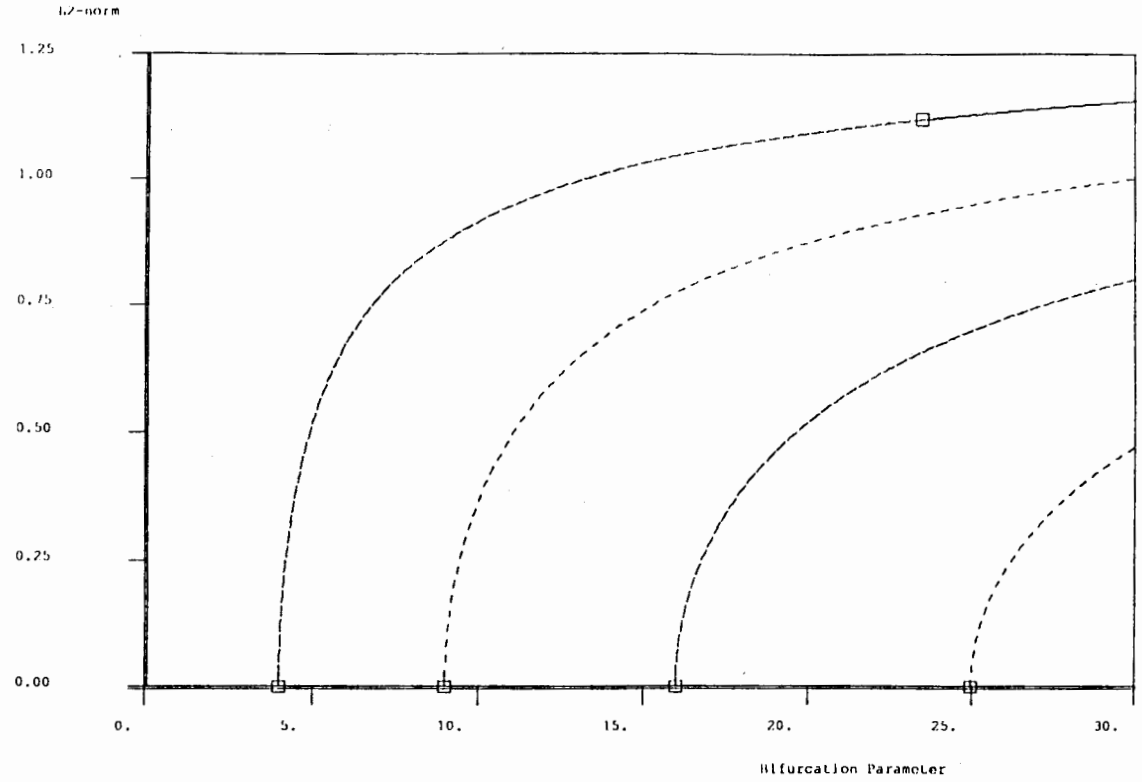

Figure 7a: 12 modes, Flat-Galerkin, RD equation

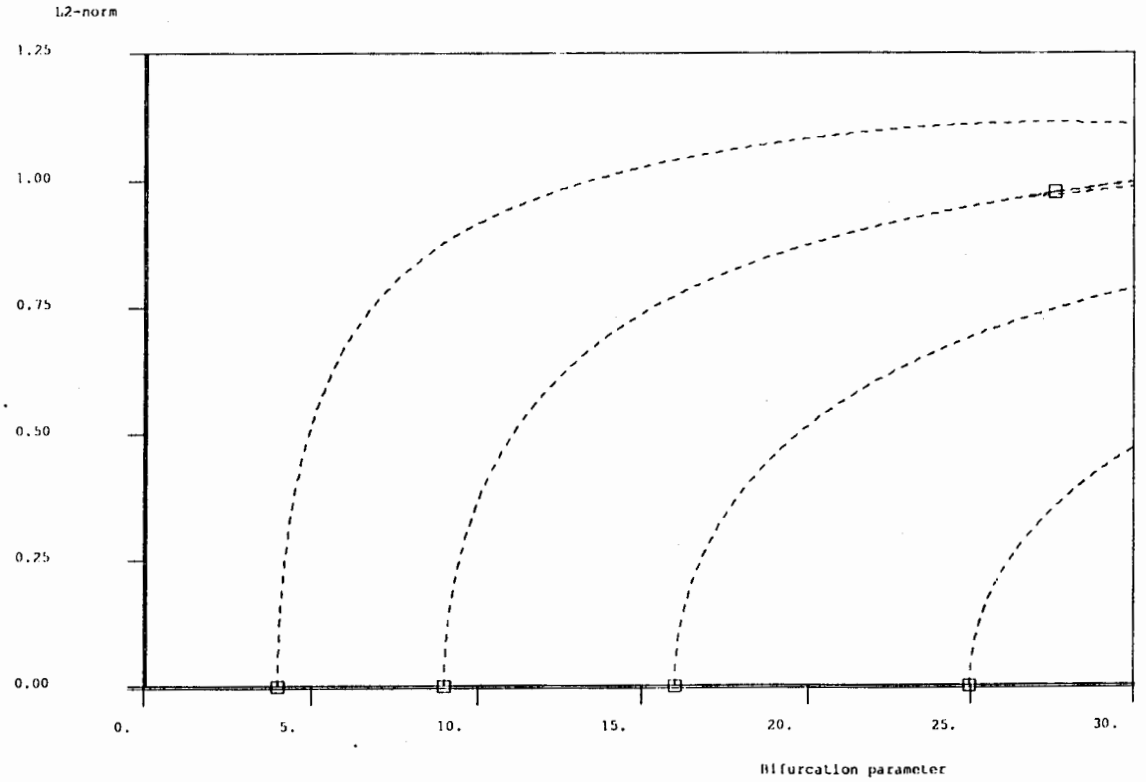

Figure 7b: 5+5, RD equation, Nonlinear, $its = 6$

Bibliography

[CFNT1] P. Constantin, C. Foias, B. Nicolaenko, and R. Temam (1989), *Spectral barriers and inertial manifolds for dissipative partial differential equations*, J. Dynamics & Diff. Eqns. 1, pp 45-73.

[CFNT2] P. Constantin, C. Foias, B. Nicolaenko, and R. Temam (1988), *Integral manifolds and inertial manifolds for dissipative partial differential equations*, Applied Mathematics Sciences, No. 70, (Springer, Berlin).

[CW] J.Cullum and R. A. Willoughby (1978), *Lanczos and the computation in specified intervals of the spectrum of large, sparce real symmetric matrices*, in Sparce Matrix Proc., ed. I. S. Duff and G. W. Stewart, SIAM Publications, Philadelphia, PA.

[D] E. J. Doedel (1980), *AUTO: A program for the automatic bifurcation analysis of autonomous systems*, Cong. Num. 30 (1981), pp. 265-284, (Proc. 10th Manitoba Conf. on Num. Math. and Comp., Univ. of Manitoba, Winnipeg, Canada).

[FJKST] C.Foias, M. S. Jolly, I. G. Kevrekidis, G. R. Sell, and E. S. Titi (1988), *On the computation of inertial manifolds*, Physics Letters A 131, pp 433-436.

[FST] C. Foias, G. R. Sell, and E. S. Titi (1989), *Exponential tracking and approximation of inertial manifolds for dissipative equations*, J. Dynamics & Diff. Eqns. 1, pp 199-224.

[GV] G. H. Golub and C. F. Van Loan (1979), *Matrix Computations*, 2nd ed., John Hopkins Univ. Press, Baltimore.

[JKT1] M. S. Jolly, I. G. Kevrekidis, and E. S. Titi (1990), *Approximate inertial manifolds for the Kuramoto-Sivashinsky equation: Analysis and computations*, Physica D 44, pp 38-60.

[JKT2] M. S. Jolly, I. G. Kevrekidis, and E. S. Titi (1991), *Preserving dissipation in approximate inertial forms*, J. Dynamics & Diff. Eqns. 3, pp 179-197.

[KP1] W. Kahan and B. N. Parlett (1974), *An analysis of Lanczos algorithms for symmetric matrices*, ERL-M467, Univ. of California, Berkeley.

[KP2] W. Kahan and B. N. Parlett (1976), *How far should you go with the Lanczos process?*, in Sparce Matrix Computations, ed. J. Bunch and D. Rose, Academic Press, New York, pp 131-144.

[L] Cornelius Lanczos (1956), *Applied Analysis*, Dover Publications, New York.

[LL] Lixin Liu and R. D. Russell, *Linear System Solvers for Boundary Value ODEs*, to appear in J. Comp. and Appl. Math.

[LS] M. Luskin and G. R. Sell (1989), *Approximation theories for inertial manifolds*, Math. Modelling & Num. Anal. 23, pp 445-461.

[M-PS] J. Mallet-Paret and G. R. Sell (1987), *Inertial manifolds for reaction diffusion equations in higher space dimensions*, IMA Preprint No. 331, June 1987.

[PR] B. N. Parlett and J. K. Reid (1981), *Tracking the progress of the Lanczos algorithm for large symmetric eigenproblems*, IMA J. Num. Anal. 1, pp 135-155.

[RST1] R. D. Russell, D. M. Sloan, and M. R. Trummer, *Some numerical aspects of computing inertial manifolds*, to appear in SIAM J. Sci. Stat. Comput.

[RST2] R. D. Russell, D. M. Sloan, and M. R. Trummer (1992), *On the structure of Jacobians for spectral methods for nonlinear PDEs*, SIAM J. Sci. Stat. Comput. 13, pp 541-549.

[S1] J. Sacker (1964), *On invariant surfaces and bifurcation of periodic solutions of ordinary differential equations*, NYU Preprint No. 333, October 1964.

[S2] J. Sacker (1965), *A new approach to the perturbation theory of invariant surfaces*, Comm. Pure Appl. Math 18, pp 717-732.

[T] E. S. Titi (1990), *On approximate inertial manifolds to the Navier-Stokes equations*, Math. Anal. & Appl. 149.

[W] David S. Watkins (1991), *Fundamentals of Matrix Computations*, John Wiley & sons, New York.