Reconsidering Backward Error Analysis for Ordinary Differential Equations

(Spine Title: Reconsidering Backward Error Analysis for ODE)

(Thesis Format: Monograph)

by

Robert H. C. Moir

Faculty of Science
Department of Applied Mathematics

Submitted in partial fulfillment
of the requirements for the degree of
Master of Science

School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

CERTIFICATE OF EXAMINATION

THE UNIVERSITY OF WESTERN ONTARIO
SCHOOL OF GRADUATE
AND POSTDOCTORAL STUDIES

Chief Advisor

Robert Corless

Examining Board

Dhavide Aruliah

Chris Smeenk

Stephen Watt

The thesis by
Robert H. C. Moir

entitled
Reconsidering Backward Error Analysis for
Ordinary Differential Equations

is accepted in partial fulfillment of the
requirements for the degree of
Master of Science

Date September 21, 2010

David Jeffrey

Chairman of Examining Board

Abstract

The idea of backward error analysis is to assess the quality of a numerical solution by regarding it as the exact solution of a nearby problem. This method of error analysis was developed by Wilkinson in the context of numerical linear algebra, and has been extended widely to other areas of numerical analysis. The subject of this work is a reconsideration of the use of backward error analysis for the numerical solution of ordinary differential equations, focusing mainly on initial value problems. The three main types, *viz.* defect control, shadowing, and the method of modified equations, are surveyed and algorithms for implementing these methods are considered. The asymptotic relationship between the local error, which is normally used to control the step-size of variable step-size numerical methods for initial value problems, and the defect, the difference between the specified problem and the problem exactly solved by the numerical method, is considered. Finally, the advantages of using backward error analysis when using ordinary differential equations to model real world phenomena, including chaotic systems, are considered. In the light of the omnipresence of physical and modeling error, the conditions under which a numerical solution can be regarded as the exact solution to *just as valid* a problem as the one originally posed are discussed.

Contents

Chapter 1

Introduction

The concept of *backward error analysis* (BEA) was first fully developed in the work of J. H. Wilkinson, the basic idea being to assess the quality of a numerical solution to a specified problem by regarding the numerical solution as the *exact* solution of a nearby problem. Although this type of analysis was used in earlier work, particularly (Von Neumann & Goldstine, 1947), and discussed explicitly by Givens (1954), it is Wilkinson that is given the credit (Fox, 1987) for being the true innovator and developer of the method. Wilkinson's first detailed discussion of the method appears in (Wilkinson, 1963).

The original uses of BEA were in numerical linear algebra (see Wilkinson, 1971, for a discussion), but the method has found wide application in numerical analysis, in areas such as function evaluation, solving of polynomial equations, polynomial interpolation and the numerical solution of ordinary and partial differential equations. The focus of the present work is the use of BEA in the numerical solution of ordinary differential equations, with an emphasis on initial value problems (IVP). Thus, we focus on error analysis for differential equations of the form

$$\frac{dy}{dt} \equiv \dot{y}(t) = f(y,t), \qquad y(t_0) = y_0, \tag{1.1}$$

where $y \in \mathbb{R}^n$, $t \in \mathbb{R}$ and $f\colon \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ ($f\colon \mathbb{R}^n \to \mathbb{R}^n$ for autonomous problems), but we will consider results for other kinds of ODE, including problems where the initial condition is replaced by boundary conditions or some algebraic condition, *i.e.*, boundary value problems (BVP) and differential-algebraic equations (DAE), and delay differential equations (DDE).[1]

The use of BEA in this area can be thought to trace back to Cauchy, with his proof of error bounds for the local truncation error for linear interpolants derived from Euler's method (Birkhoff & Rota, 1989, 207). This is, to the best of my knowledge, the first use of the *defect* to compute a bound on the *global error* $\|y(t) - u(t)\|$, where $y(t)$ is the exact solution of the ODE (1.1) and $u(t)$ is an interpolant derived from the numerical solution $y_n = u(t_n)$.

Definition 1.1 (Global Error) The *global error* of an interpolant $u(t)$ of a numerical solution to the ODE (1.1) is the difference

$$E(t) = y(t) - u(t)$$

between the exact solution $y(t)$ to (1.1) and the interpolant. In some cases 'global error' is also used to refer to the norm $\|y(t) - u(t)\|$ of the difference between the two.

[1] Although DDE are strictly speaking a different class of problem, since they are infinite dimensional, they still involve only ordinary derivatives and so are included in this list.

Definition 1.2 (The Defect) The *defect*, or *residual*, is the quantity

$$\delta(t) := \dot{u}(t) - f(u,t). \tag{1.2}$$

Since this implies that $\dot{u}(t) = f(u,t) + \delta(t)$, the defect is the amount by which the numerical solution fails to satisfy the differential equation (1.1).

The notion of the defect is connected to the idea of BEA: because the defect is the amount by which the numerical solution fails to satisfy the differential equation, it is also the difference between the original equation (1.1) and the equation solved exactly by the numerical method (see section 2.1). Although the defect has a long history, the first use of BEA proper took place when Wilkinson's ideas were consciously extended into the numerical solution of ODE. Chapter 2 provides a survey of the development of BEA for ODE. A survey of particular numerical methods for BEA is provided in chapter 3.

The general idea of BEA can be understood by thinking of a mathematically posed problem as a map f from a data space $\mathcal{D}$ to a solution space $\mathcal{S}$. A schematic diagram of this picture is provided in figure 1.1. Given the specified data $x \in \mathcal{D}$ for a specified problem f, the problem maps it to its exact solution $y = f(x) \in \mathcal{S}$. Since the exact solution is usually not available, however, one often uses a numerical method to obtain an approximate solution $\hat{y}$ to the specified problem. This approximate solution $\hat{y}$ will be some distance Δy away from the exact solution y.

Definition 1.3 (Forward Error) Let y be the exact solution to a problem f and $\hat{y}$ be an approximate solution. The difference $\Delta y = \hat{y} - y$ is the *forward error*.

We usually want the forward error to be small since we usually seek the exact solution y to the specified problem f. But we are rarely able to calculate or estimate the forward error directly, since in general we know very little or nothing at all about the exact solution y. Rather than focusing concern on the forward error, the shift in thinking in BEA is to consider how large a perturbation Δx of the data x is required so that the numerical solution $\hat{y}$ is the solution of the specified problem f with the perturbed input data $\hat{x} = x + \Delta x$.

Definition 1.4 (Backward Error) Let $\hat{y}$ be an approximate solution to a problem f with specified data x. Then the *backward error* is the amount Δx that the data x must be varied in order for $\hat{y}$ to be the exact solution of f with data $\hat{x} = x + \Delta x$.

Unlike the forward error, the backward error can often be calculated or estimated. Thus, the strategy is to reflect back consideration of the forward error into a consideration of the backward error. Now, the diagram in figure 1.1 commutes. So, this enables us to view the approximate solution $\hat{y}$ to specified problem as the *exact solution to a modified problem* $\hat{f}$, the diagonal map. The guiding idea of BEA is that if the backward error is small, then the numerical solution is the exact solution to a nearby problem.

In some cases, including the present case of ODE problems, it is appropriate to consider the data space $\mathcal{D}$ as a space of problems (or equations); in such cases the perturbed input $\hat{x}$ is understood to be the modified problem (or equation) that is solved exactly by the numerical method. In the context of ODE problems the map f is the problem ‘solve a system of ODE,’ and the

$$\begin{tikzcd}
 & y = f(x) \arrow[dd, leftrightsquigarrow, "\|\Delta y\|"] \\
x \arrow[ur, "f"] \arrow[dr, dashed, "\hat{f}"] \arrow[d, leftrightsquigarrow, "\|\Delta x\|"'] & \\
\hat{x} = x + \Delta x \arrow[r, "f"'] & \hat{y} = f(\hat{x})
\end{tikzcd}$$

Figure 1.1: Schematic representation of backward error Δx and forward error Δy.

diagonal map $\hat{f}$ is the problem 'solve a perturbed system of ODE.' It is because we are interested in the perturbed system $x + \Delta x$ as a modified model, and the size of the perturbation Δx as a perturbation of the specified model, that we consider the the data space as a space of problems.

One of the main advantages of BEA in general is that it allows one to assess the quality of the solution to a problem without knowing what the exact solution is—one knows that if the backward error is small and that the relevant quantities in the model vary continuously under perturbation, then one has a valid solution.[2] If, in addition, one desires an applicable or a useful solution, then one also requires a knowledge of the *conditioning* of a problem, *i.e.*, how sensitive the solution to a problem is to a variation of the problem itself. In the general picture above, the conditioning of the problem is determined by how sensitive the solution y to a problem f is to variations in the data x.

Definition 1.5 (Conditioning) Let f be a problem with specified data x and solution y. Then, f is *well-conditioned* if small changes in the data x lead only to small changes in the solution y, and f is *ill-conditioned* if small changes in

[2] For a careful definition of the term 'valid' in this context see Stetter (2004).

the data x lead to large changes in the solution y.

The conditioning of a problem is usually characterized by a *condition number*, which characterizes how much perturbations of the data are ampified in their effect on the solution. In the general picture above an example of a condition number κ of a problem f is

$$\kappa = \sup \frac{\text{relative change in solution}}{\text{relative change in data}} = \sup_{x \in D \subseteq \mathcal{D}} \frac{\|f(\hat{x}) - f(x)\| / \|f(x)\|}{\|\hat{x} - x\| / \|x\|}.$$

So we see that the condition number is like a derivative, measuring (the maximum over some data range $D \subseteq \mathcal{D}$ of interest of) the rate of change in the solution for a given change in the data.

As an example, in the case of linear systems $Ax = b$ it is well-known that the condition number is the product of the norms of the matrix A and its inverse, making it easy to estimate or calculate. With an estimate of the condition number κ, one knows that if the backward error (BE), *i.e.*, the residual, is small and the condition number is small, then the forward error (FE) is also small, since

$$\|\text{FE}\| \lessapprox \kappa \cdot \|\text{BE}\|.$$

The situation in the case of ODE is similar. Obtaining a condition number for ODE is more involved, but is possible using results from variational calculus.

The conditioning of ODE problems is known in the differential equations literature using the terminology of stability properties of solutions of differential equations. The notion of stability coming from the dynamical systems context has to do with stability of solutions under perturbations, *e.g.*, how much a solution will change if the initial condition is varied slightly. Thus, stability in the dynamical systems context means something like the notion of conditioning from numerical analysis. This kind of stability may be called *dynamical* stability. This is not to be confused with the notion of *numerical* stability that arises in the numerical analysis context. Numerical stability applies to algorithms, where an algorithm is numerically stable if it reliably produces a solution with small (backward) error for the range of inputs of interest. See table 1.1.

dynamical systems	**numerical analysis**
dynamical stability	conditioning
—	numerical stability

Table 1.1: There is an analogy between dynamical stability and conditioning, but there is no correlate of numerical stability in the context of dynamical systems.

Chaotic systems offer a clear example of the connection between dynamical stability and conditioning. The instability of chaotic dynamical systems is seen in the characteristic (on average) exponential divergence of solutions with slightly different initial conditions. This instability translates to an ill-conditioning of the problem since the result of a small variation in the initial conditions, due to roundoff error, and a small variation in the dynamics, a small defect, is that the solution produced by the computer usually diverges exponentially fast from the exact solution to the specified problem. Thus, even though the backward error is small the global error will be very large after running the integration for a nontrivial length of time.

This connection between dynamical stability and conditioning can be made more precise in the following way. Given equation (1.2) for the defect we may see that the interpolant $u(t)$ of the numerical solution y_n satisfies the equation

$$\dot{u} = f(u,t) + \delta(t), \qquad u(t_0) = y_0^* \tag{1.3}$$

exactly, where y_0^* is the closest machine number to y_0. We now wish to perturb the problem about the solution $u(t)$. Suppose that $y(t)$ solves the original system (1.1) exactly. Then we have (*e.g.*, Corless, 1992b, 332) that

$$y(t) = u(t) + \varepsilon y_1(x) + O(\varepsilon^2), \tag{1.4}$$

where $y_1(t)$ is the solution to the first variational equation

$$\dot{y}_1 = J_f(u(t),t)y_1(t) + v(t),$$

where $J_f(u,t)$ is the Jacobian of the vector field $f(u,t)$ and $\delta(t) = \varepsilon v(t)$, $\|v(t)\| \leq 1$, $\varepsilon \ll 1$ (these equations ensure that the defect is a small perturbation of the original problem, see section 2.1 for more). This equation has the solution

$$y_1(t) = \Lambda(t)\Lambda^{-1}(t)y_1(t_0) + \int_{t_0}^{t} \Lambda(t)\Lambda^{-1}(\tau)v(\tau)d\tau, \tag{1.5}$$

where $\Lambda(t)$ is a fundamental solution matrix of the homogeneous version of the first variational equation.[3] From (1.4) we see that (to first order) $\varepsilon y_1(t)$ is the global error, so that equation (1.5) shows that the global error is determined by an integral of the defect ($\varepsilon v(t)$) multiplied by the fundamental solution

[3] Corless (1992b, 332) points out that this matrix can be computed to $O(\varepsilon)$ or better.

matrix and its inverse. Thus, the quantity

$$\kappa(t) = \int_0^t \|\Lambda(t)\Lambda^{-1}(\tau)\| d\tau$$

is a condition number for ODE, since for $\|v(t)\| \leq 1$ and $\varepsilon \ll 1$, we have (Corless, 1992b, 329) that (over a finite time period $[0, T]$ and if $y_1(0) = 0$)[4]

$$|y(t) - u(t)| \approx \varepsilon |y_1(t)| \leq \kappa\varepsilon.$$

Now, the connection to dynamical stability is made since the fundamental solution matrix $\Lambda(t)$ actually characterizes the dynamical stability properties of the system. For perturbations of the initial condition or the vector field f of the equation, the norm of the fundamental solution matrix determines the rates at which exponential growth or decay of the distance between solutions occurs. More specifically, the fundamental solution matrix determines the Lyapunov exponents (Shimada & Nagashima, 1979, 1606-7). Thus, the dynamical stability of the differential equation is directly tied to the notion of the conditioning of the problem by the fundamental solution matrix.

A more general form of equation (1.5) is given by the Alekseev-Gröbner

[4] This is not the only condition number that one can define here. Generally, the condition number depends on the range of initial conditions one is interested in and the time period of interest. One could, therefore, use the integral over the entire domain of interest, as is done in Ascher *et al.* (1988). Treating the condition number as a function has advantages, however, since it enables one to determine for what times, or time ranges, the trajectory is particularly sensitive to perturbations of the problem.

nonlinear variation of constants formula, *i.e.*,

$$y(t) - u(t) = \int_{t_0}^{t} G(t, \tau)\delta(\tau)d\tau,$$

where $G(t, \tau)$ is a nonlinear analogue of the Green's function characterizing how the evolution of the system varies with variation of the initial conditions (a more precise formulation is given in chapter 4). In many cases, using the first variational equation the function $G(t, \tau)$ can be usefully approximated using J_f, the Jacobian of the vector field f, making the linear stability properties of the differential equation give useful information about how the defect connects to the global error.

These results are a major part of the motivation for the use of defect control methods, since they clarifiy the relationship between the defect and the global error, and in a way that does not depend on the details of the problem at hand. Nevertheless, most codes for IVP control the so-called *local error* in order to indirectly control the global error, and these codes have proved to be very successful in practice. There are two kinds of 'local error' that codes for IVP can control, which are defined in terms of the local exact solution.

Definition 1.6 (Local Exact Solution) Let y_n be the solution value to the IVP (1.1) produced by a numerical method before the n-th time-step is taken. Then the *local exact solution* is the solution $z_n(t)$ to the local IVP

$$\dot{z}_n(t) = f(z_n, t), \qquad z_n(t_n) = y_n, \tag{1.6}$$

the same ODE as (1.1) but with initial condition $z_n(t_n) = y_n$.

The term *local truncation error*, or simply *local error*, is sometimes used to refer to the *local error per step* and sometimes to the *local error per unit step.*

Definition 1.7 (LEPS) Let y_{n+1} be the solution value to the IVP (1.1) produced by a numerical method after the n-th time-step h_n. Then the *local error per step* (LEPS) is the error quantity

$$\epsilon_{n+1} = z_n(t_n + h_n) - y_{n+1},$$

where $z_n(t)$ is the local exact solution.

Definition 1.8 (LEPUS) Let y_{n+1} be the solution value to the IVP (1.1) produced by a numerical method after the n-th time-step h_n. Then the *local error per unit step* (LEPUS) is the error quantity

$$e_{n+1} = \frac{z_n(t_n + h_n) - y_{n+1}}{h_n},$$

where $z_n(t)$ is the local exact solution.

We will use the term 'local error' in cases where we are not referring specifically to one or the other of these two quantities.

From the backward error point of view, we would like to understand the success of codes that control the local error because they indirectly provide a control of the defect. In order to be able to understand the success of local error control codes in this way, it is required to establish the precise nature of the connection between the local error and the defect. Although it is difficult to understand this connection, it is easier than connecting the local and global error. And it is useful to examine this problem because if local error control can be understood to indirectly control the defect, which is a much more intuitive notion than local error, then it will be much easier for users to understand why the code works. This problem is considered in chapter 4 and some results are presented.

Various kinds of physical and modeling error[5] are always present in the mathematical modeling of real world systems, which means that in the context of modeling one must always examine how perturbations affect the specified model. In light of this issue, BEA becomes a powerful tool for analyzing the validity and applicability of numerical solutions to the specified model. The connection between numerical error introduced by the numerical solution and physical and modeling error is made clear using BEA since, under the right conditions, it is possible to treat the effect of all forms of error in terms of perturbations of the problem. The conditions under which this comparison is valid are discussed in chapter 5.

Special issues come into play in the application of BEA to ill-conditioned problems, specifically chaotic problems. In the case of chaotic problems BEA actually works quite well, even though the instability of such problems makes indirect control of the global error over nontrivial times impossible. The reasons why BEA is advantageous on chaotic problems are discussed in section 5.1. This is followed by a brief concluding section considering the reasons for the effectiveness of the various forms of BEA for ODE in the context of the mathematical modeling of real world systems.

[5] For a conceptual clarification and discussion of physical and modeling error see chapter 2.

Chapter 2

Backward Error Analysis for Ordinary Differential Equations

The roots of BEA as applied to ODE reach as far back as the original work by Wilkinson. Examples of early applications of BEA to ODE include Osborne (1964) who treated a difference equation satisfied by the exact solution to the specified problem as a perturbation of a difference equation obtained by finite-difference approximation, and Fox & Mayers (1968) who analyzed the stability of numerical methods for solving various problems, including ODE, using BEA. There were a number of papers in the 70s that used BEA to analyze the numerical solution of particular ODE problems. It was not until the 80s and 90s, however, that theoretical studies of different approaches to BEA for ODE were conducted and BEA became a common manner of treating the error introduced by numerical methods for ODE.

In the consideration of error analysis for ODE, one must be mindful of the various sources of error. Enright (2010) describes three potential sources of error in the numerical solution of IVP (1.1). First there is *modeling error*. This type of error is generated only in the formulation of the mathematical model, either because the exact equations of motion are either not completely known or because they are too complicated to solve directly, so that the formulated model only approximates the true dynamics. A simple way of representing modeling error is where the actual system we are interested in is

$$\dot{x}(t) = g(x,t), \qquad x(t_0) = y_0,$$

with g unknown or too complicated, but $\|g(x,t) - f(x,t)\|$ is small for the range of values of (x,t) of interest. In this case the exact solution $y(t)$ of (1.1) satisfies

$$\begin{aligned} \dot{y}(t) &= f(y,t) \\ &= g(y,t) + \mu(t), \qquad y(t_0) = y_0, \end{aligned} \tag{2.1}$$

where $\|\mu(t)\| = \|f(y,t) - g(y,t)\|$ is small for some norm relevant to the problem.

Second there is *floating point error.* This type of error is generated only by the floating point (FP) arithmetic system used, because the IVP is defined on the computer by a subroutine that evaluates $f(y,t)$. Each derivative evaluation is computed in FP arithmetic and thus satisfies

$$\begin{aligned} \texttt{yp} &= \texttt{fl}(f(y,t)) \\ &= f(y,t) + \phi(t), \end{aligned}$$

where $\|\phi(t)\|$ depends on f, the code for the subroutine that evaluates it, and the FP system used. $\|\phi(t)\|$ is a small multiple of $\|f\|\rho$, where $\rho \approx 10^{-15}$ for double precision IEEE FP systems (Enright, 2010, 5).

Third there is *discretization* or *truncation error.* This type of error is generated only by the numerical method, because most IVP do not admit closed form solutions and so an approximation amenable to solution using a computer must be used. Most software for the solution of IVP provides a continuous (at least piecewise C^1, but sometimes C^1 or smoother) interpolant $u(t)$ of the numerical approximation on the integration interval $[t_0, T]$ which satisfies

$$\dot{u}(t) = f(u,t) + \phi(t) + \tau(t), \qquad u(t_0) = y_0^*,$$

where y_0^* is the closest FP number to y_0, since the numerical method actually solves the problem $\dot{v} = f(v,t) + \phi(t),\ v(t_0) = y_0^*$. The numerical method often tries to ensure that $\|\tau(t)\| \lessapprox C\varepsilon$, where ε is some user-specified tolerance and C is a constant close to 1. Note that this implies that the defect $\delta(t)$ calculated from the interpolant $u(t)$ by a computer is equal to $\tau(t)$, which will be much larger in norm than $\phi(t)$ provided that the tolerance ε is not too close to the machine epsilon and the subroutine used to compute f is numerically stable.

A fourth kind of error that is not mentioned by Enright, but is nevertheless important to consider when using numerical solution of differential equations to model real world systems, is *physical error.* This type of error is generated only by physical interactions between the system being modeled and its environment, either because of actual physical perturbations of the system being modeled or error introduced in the measurement of parameters in the model.

Thus, the complete description of the system being modeled is of the form

$$\dot{x}(t) = g(x,t) + \pi(t),$$

where generally $\|\pi(t)\| \ll \|y(t)\|$.

The line that is drawn between physical error and modeling error will depend on the theory or theories being used and the modeling assumptions involved. To clarify the difference between modeling and physical error consider the usual application of classical mechanics to generate a macroscopic model of the simple pendulum system (consisting of a bob, a string, and the pivot):

$$\ddot{\theta} = -\tfrac{g}{\ell}\sin\theta,$$

where g is the gravitational acceleration at the earth's surfce, ℓ is the length of the string, and θ is the angle made between the string attached to the swinging bob and the vertical. The modeling assumptions include idealizing the bob as a point mass, assuming the string does not stretch and is massless, and assuming no friction at or movement of the pivot. These modeling assumptions introduce modeling error by taking one away from more accurate macroscopic classical models of the simple pendulum system, which relax some or all of the above idealizing assumptions.[1] Further modeling error is introduced when mathematical methods are used to generate a simplified model that is more

[1] Note that relaxing these assumptions can result in an increase of the dimensionality of the model. Thus, the modeling error would be determined by the difference between the simpler model and an appropriate projection of the more complex model into a space of the same dimension as the simpler model.

tractable. This would include the usual technique of linearizing the equation, as well as dimensional analysis and perturbation theory.

Now, the physical error in this case includes physical perturbations of the actual pendulum being modeled due to interactions between the pendulum system (bob, string, pivot) and neighbouring systems outside the scope of the pendulum system. This could include, for example, vibrations of the pivot due to a neighbouring freeway as well as the effect of air resistance on the bob and string. Now, by changing what we count as the system being modeled, we change what counts as modeling error and as physical error. For example, as soon as we include the effect of air resistance into the model, the air in the vicinity of the pendulum becomes part of the system being modeled, and the the actual effect of the air resistance as could be modeled using classical mechanics (compared to an idealized model of air resistance) becomes modeling error and not physical error. Although the distinction between modeling and physical error can be difficult to determine, the distinction nevertheless serves to separate the effect of modeling and mathematical idealizations from the effect of physical perturbations.

Let us now consider the consequences of this point of view for error analysis. The model of the system of interest is given by

$$\dot{y}(t) = g(y,t), \qquad y(t_0) = y_0. \tag{2.2}$$

Letting $\gamma(t) = \mu(t) + \nu(t)$, where $\nu(t) = \phi(t) + \tau(t)$ is the *numerical error*, we can interpret the three sources of error involved in the solution of the model of the system of interest in terms of perturbations of the ODE of interest:

$$\dot{u}(t) = g(u,t) + \gamma(t), \qquad u(t_0) = y_0^*. \tag{2.3}$$

Notice that $\gamma(t)$ is the defect between the model problem (2.2) and the problem (2.3) solved exactly by the numerical method. This equation emphasizes that by taking a backward error point of view, the analysis of error is reduced to the issue of how small perturbations affect the problem of interest. In cases where one is interested in the actual phase trajectory of the physical system being modeled, the main question one wishes to address is the relationship between the size $\|\gamma(t)\|$ of the perturbation and the size $\|y(t) - u(t)\|$ of the (global) error, the pointwise difference between the exact solutions of the model and perturbed equations. Now, analyzing the problem by considering the defect $\gamma(t)$ would be difficult to do directly in practice because $g(y,t)$ and $\mu(t)$ are usually not known or cannot be calculated. An advantage of BEA is that the problem can usefully be analyzed by considering the size of the defect $\tau(t)$, which can be calculated by a computer used to obtain the numerical solution of $\dot{v}(t) = f(v,t) + \phi(t),\ v(t_0) = y_0^*$, the (simplified) version of the model problem seen by the computer.[2]

To see why we can usefully consider $\|\tau(t)\|$ it is important to recognize that the actual physical situation is described by a perturbed version of the original model, because any physical system is subject to perturbations, which may be very small. Consequently, the actual physical situation will be modeled by the

[2] It is a simplified version of the model problem because the vector field $f(v,t) = g(v,t) + \mu(t)$ is the result of the idealizations and simplifications that introduce the modeling error $\mu(t)$.

problem

$$\dot{x}(t) = g(x,t) + \pi(t), \qquad v(t_0) = y_0 + \pi_0. \tag{2.4}$$

Thus, we may now see that equations (2.4) and (2.3) emphasize the manner in which the error introduced in the solution of a problem can be treated on a par with physical perturbations. Understanding the situation this way, we see physical perturbations shake the system off of the original problem (2.2), so that the original problem is a modified version of the problem that actually tracks the behaviour of the physical system. Thus, both physical error on the one hand, and modeling and numerical error on the other, shake the system off of the original model problem. Consequently, the consideration of how small perturbations affect the model problem is essential whenever one is modeling real world phenomena. The essential point here, however, is that the presence of physical perturbation means that, if the original problem is to be any use in applications, the exact solution of any problem sufficiently close to the original problem will also be a valid solution to the original problem.

Now, to make the connection to why we may focus on the defect $\tau(t)$, consider the following. Assuming that $\|\mu(t)\|$ is very small, *i.e.*, that the posed model captures the dominant behaviour of the system being modeled, then as long as $\|\nu(t)\|$ is small, one knows that one has a high quality solution, or that an exact solution to (1.1) is valid. Also, we wish to ensure that $\|\phi(t)\| \ll \|\tau(t)\|$ so that floating point error does not interfere with numerical solution and the calculation of the defect. So, $\tau(t)$ dominates perturbation of (1.1). With tight tolerances and high order methods we can usually ensure that $\|\tau(t)\| < \|\pi(t)\|$ for the largest sources of physical error, so the numerics are perturbing the problem to a lesser degree than physical perturbations are (for some suitable norm depending on the problem). So, we know that the exact solution we obtain to (2.3) must also be valid. If, in addition, $\tau(t)$ modifies the problem (2.2) in a manner that is similar to how $\pi(t)$ modifies the problem, *i.e.*, if the numerical error perturbs the problem in a similar way that physical perturbations do, then one knows that one has exactly solved a problem that is *just as valid* as the model of interest. The ability to treat numerical error on a par with physical perturbation, enabling the use of the powerful general methods of perturbation theory, and the ability to regard the problem exactly solved by a numerical method as just as valid as the one originally posed, enabling us to get just as much insight from the numerical solution as we would get from the exact solution to the original problem, are two major advantages of the backward error point of view. This is discussed further in chapter 5.

There are three main varieties of BEA in the contemporary literature, which work by modifying the differential equation, the conditions on an equation, or both. The first is defect analysis, which works by modifying the equation and holding the conditions fixed. This is the most natural kind of BEA for ODE and the point of view that has been emphasized so far. The numerical method is understood to exactly solve an equation that differs from the specified one by just the defect (1.2), which is regarded as a non-autonomous perturbation of the specified ODE (1.1).[3] The defect is most often encoun-

[3] The defect can be regarded as an autonomous perturbation of the ODE by increasing the dimensionality of the system by one using the standard method for converting a non-autonomous system into an equivalent autonomous one, *i.e.*, by setting $x_{n+1} = t$ and adjusting the equations of the system accordingly.

tered in the context of defect control, where the size of the norm of the defect, for a suitable norm, often the max norm, is controlled by a numerical method. Numerical methods using defect control use an interpolant of the emerging solution of a numerical method to estimate or compute the defect, which is then used to control the step-size of the numerical method. This kind of error analysis is particularly useful when the problem is subject to a non-negligible amount of physical or modeling error, so that the specified DE ought to be regarded as approximate anyway, and so the consideration of a modified problem that we can solve exactly is entirely justified.

The second variety of BEA for ODE is shadowing, which works for IVP by modifying the initial conditions of a problem while leaving the equation (1.1) fixed. The main task in shadowing is to show that the numerical method follows an exact solution of the specified problem with perturbed initial conditions for some period of time. This kind of analysis is much more involved than in defect control. Since the differential equation is being held fixed, the method is more suited to problems where the equations of motion are very well-known and/or the physical and modeling error are negligible, so we are interested in exact solutions to the original problem and so the consideration of a modification of the DE is less justified.

The third variety is the method of modified equations, which works by modifying both the equation (1.1) and the conditions on it. This approach uses both the specified equation and the equations defining the numerical method to determine the equations and conditions of a modified problem, the solutions of which are followed more closely by the numerical solution than the solutions of the specified one. Although the method of modified equations is not always considered a form of BEA, its application enables one to explain the behaviour of a numerical method in terms of a consideration of a nearby problem, which makes it very much in the spirit of BEA. Moreover, defect analysis can actually be considered as a special case of the method of modified equations, namely where the equation is subject to a non-autonomous perturbation and the conditions of the problem are held fixed. Thus, the method of modified equations is perhaps better regarded as a generalization of defect analysis and, hence, very much a form of BEA.

Although these three varieties of BEA tend to be considered on their own, combinations of these methods are also possible; indeed, the various methods can complement each other. For example, Corless (1994a) combines the method of modified equations and defect control in an analysis of the numerical solution of chaotic dynamical systems. We will consider this type of analysis in section 5.1.

We now turn to examine the main developments for the three varieties of BEA for ODE.

2.1 Defect Control

With a piecewise interpolant $u(t)$ of the solution y_n of a numerical method in hand, one can compute the defect

$$\delta(t) := \dot{u}(t) - f(u,t) = \varepsilon v(t),$$

where $\dot{y}(t) = f(y,t)$ is the original ODE, and ε is a small parameter, which is a user-specified tolerance for defect controlled methods. An example of this kind of calculation using piecewise cubic Hermite interpolation is provided in section 3.1. Rewriting this as

$$\dot{u}(t) = f(u,t) + \varepsilon v(t), \tag{2.5}$$

we see that provided that $\|v(t)\| < 1$ for some suitable norm suggested by the problem, then the numerical method exactly solves an ε-nearby problem. Depending on the problem it may be more appropriate to consider the relative defect

$$\dot{u} = f(u,t)(1 + \varepsilon v(t)),$$

or the defect[4] relative to u

$$\dot{u} = f(u,t) + \varepsilon v(t) u.$$

Part of the rationale for defect control is that the relationship of the defect to the global error much less sensitive to the method used as compared to the local error (for more on this see chapter 4). As mentioned above, controlling the defect enables the user to have a better understanding of how the specified tolerance relates to the global error of the numerical solution, through the Alekseev-Gröbner formula, which will be discussed in greater detail below (see chapter 4 for a statement of the Alekseev-Gröbner theorem). This makes it

[4] Note that in this case $\delta = \delta(u,t)$.

much easier for the user to interpret the results of a computation, and the result of keeping the size of the defect below the specified tolerance. A further advantage of defect control, particularly when using methods that bound the defect rather than estimate it, is that it enables one to regard the solution generated by a numerical method as just another step in the simplification process used to obtain an exact solution.

The first advocate for the use of defect for practical error control was Zadunaisky (1966). This work was in the context of defect correction, *i.e.*, where the solution is improved by using an iterative method to decrease the defect, but the defect is used to provide a practical estimation of the error. Among the first works on the use of the defect as a means of controlling the global error are Hull (1968), Hull (1970) and Stetter (1976). Stetter (1976) begins to address the important issue of establishing so-called 'tolerance proportionality,' *i.e.*, a (preferably) linear relationship between the user-specified tolerance and the global error (for a more precise definition of tolerance proportionality see chapter 4). Using a certain kind of piecewise differentiable interpolant, in this paper he established tolerance proportionality for methods such that the local error is demonstrably equal to the defect, except for terms numerically small compared to the tolerance. Stetter's approach uses a special interpolant of the numerical solution and does not address the issue of how the requisite sort of interpolant may be obtained. The details of Stetter's main result are discussed in chapter 4.

Another important issue to consider is the *order* of the interpolants that one generates for a numerical method.

Definition 2.1 (Order of an Interpolant) Let $u_n(t)$ be an interpolant of a numerical solution y_n of (1.1) over the n-th time-step. Then, the *order* of the interpolant is the integer p such that

$$u_n(t_n + \theta h_n) - z_n(t_n + \theta h_n) = h_n^p \psi(t_n, y_n; \theta) + O(h_n^{p+1}),$$

where $\theta \in [0, 1]$, z_n is the local exact solution 1.6, and $\psi(t_n, y_n; \theta)$ is the principal error term for the interpolant, which depends on the numerical method. Thus, the order of an interpolant is the asymptotic rate of convergence of the local error for the interpolant over the time-step as the step-size goes to zero.

A suitable interpolant must have an order equal to or higher than the order of the numerical method. Enright *et al.* (1986) develop a general "bootstrapping" method for extending a RK formula pair to include high-order piecewise interpolants of computable accuracy. In their approach each of the pair of methods generates an interpolant, which together are used to estimate the error. They are then able to give an expression for the leading term in the local error. They discuss how the defect might be used to estimate the global error, using defect correction or the more standard technique of integrating the variational equation associated with the problem, but point out that the implementation of these techniques with their interpolants would not be straightforward.

One of the first defect control methods for ODE was provided by Enright (1989a). He introduces and justifies a defect control method for continuous Runge-Kutta methods (CRK), which are Runge-Kutta formula pairs with interpolants. The control mechanism tries to ensure that

$$\|\delta(t)\|_\infty \leq \varepsilon$$

over the entire interval of integration. The step is accepted if the above condition is satisfied, and then the next step size is selected according to the following heuristic:

$$h_{\text{new}} = h_{\text{old}} \left(\nu \frac{\|E_n(h)\|}{\varepsilon} \right)^{1/r},$$

where $\|E_n(h)\| = e_r h^r + O(h^{n+1})$ is an estimate of the size of the defect, estimated by sampling k values of $\delta(x)$, and $0 < \nu < 1$ is a "safety factor." The control of the error using the defect estimate is regarded as an attempt to control the maximum defect over the integration interval. Enright emphasizes two large parts of the motivation for the use of defect control: that it is able to produce methods for which the accuracy/ε relation is much less sensitive to the particular method used; and it is much easier for a user to understand and interpret than methods using local error control, because ensuring a small defect ensures that you are solving a nearby ODE and that the backward error is small. But there is a tradeoff between robustness and cost, so an important consideration in the use of defect control is the balance of the accuracy of the estimate of the defect (here depending on k) and the cost due to the derivative evaluations per step: Better robustness takes more computing time.

In order to demonstrate the reliability and efficiency of the use of defect control, as compared to the standard local error control, Enright (1989b) identifies and quantifies the effectiveness of quite natural and inexpensive defect control strategies. Specifically, he discusses four strategies that can be used to control the size of the defect for continuous Runge-Kutta methods, two of which involve local error control and two that involve defect control. The asymptotic properties of the methods are analyzed and compared to show that the defect control strategies, though more expensive, compare favourably with local error control strategies. An asymptotically correct estimate of the defect is provided for one of the defect-controlled methods, which enables more accurate estimation of the maximum defect over a time-step.

Definition 2.2 (Asymptotically Correct Estimate) An estimate of the defect over a time-step of a numerical method is *asymptotically correct* is if, asymptotically (as $h \to 0$), the particular shape of the defect is known.

Enright (1993) compares the relative efficiency and reliability of a wider range of continuous Runge-Kutta methods, of orders 4 to 8, including ones that are less expensive than those of (Enright, 1989b). He also introduces theoretical measures that can be used to evaluate the potential of such methods. Although most of the defect control codes examined are continuous Runge-Kutta codes, Higham (1989a) examines the problem of reliably estimating the defect for variable-step, variable-method Adams codes.

The methods developed in Enright (1989a) and Enright (1989b) are not particularly robust. The method in (Enright, 1989a) samples the defect at one or more fixed points within each step, but the quality of the sample point is problem-dependent and varies from step to step. Higham (1989b) develops Enright's method of defect control by presenting two interpolants for which the asymptotic behaviour is known *a priori*, in that each component of the defect behaves asymptotically like a multiple of a known polynomial. This allows optimal sample points to be chosen. Indeed, it becomes possible to construct an asymptotically correct approximation to the defect over the entire step using only a single sample value. Limitations of Higham's methods are that they are only applicable to low-order Runge-Kutta methods and they produce a defect with a lower asymptotic order than is optimal. Higham (1991b) addresses these limitations, obtaining methods applicable to Runge-Kutta methods of any order, by introducing Runge-Kutta defect control using Hermite-Birkhoff interpolation. More recently, Enright & Hayes (2007) achieve a balance of cost and robustness for defect-controlled CRK methods. An important feature of the codes provided is that they provide a user-friendly implementation of direct defect control and code is provided both for FORTRAN and Matlab addressing the important issue of the flexibility and usability of the code.

Although the methods developed by Higham (1989b, 1991b) and Enright & Hayes (2007) provide asymptotically correct estimations of the defect, they still only approximate the maximum value of the defect over the interval of integration. Corless & Corliss (1992), however, develop a defect control method that uses interval arithmetic to guarantee a bound on the defect over the interval of interest. This approach is complementary to that of Lohner (1987), which involves computing an interval that is guaranteed to enclose the exact solution of the specified problem.

As has been emphasized, part of the attraction of the use of defect control is that it ensures that the solution produced by a numerical method is an exact solution to a nearby ODE. Although this is true in general,[5] it is for this reason that defect control is particularly attractive in the case of chaotic problems. Corless (1992b) explains how for what he calls a 'well-enough conditioned problem,'[6] defect control gives useful solutions for chaotic problems. The sensitivity of chaotic problems to variation of the initial conditions makes attempts to control the global error futile, but control of the defect for well-enough conditioned chaotic problems ensures that one obtains the exact solution to just as valid a model as the one specified and that one can gain just as much insight from the numerical solution as one would get from the exact solution. The advantages of using backward error analysis for chaotic problems is discussed in greater depth in section 5.1. For further details one may also see (Corless, 1994a) and (Corless, 1994b).

Defect control methods have also been developed for other classes of ODE. It was suggested by Cash & Silva (1993) that monitoring the defect could be appropriate for the solution of boundary value problems (BVP) for ODE when there are difficulties in estimating the global error. Enright & Muir (1996) went on to develop continuous mono-implicit Runge-Kutta (CMIRK) methods used to estimate the defect for BVP, which compared favourably with previous software packages. More recently, Kierzenka & Shampine (2001) improved upon this work developing `bvp4c`, a defect-based BVP solver that is now a standard component of the MATLAB PSE. They are able to provide inexpensive, asymptotically correct, estimates of the L_2 and L_∞ norms of the

[5] Aside from so-called stiff problems, for which the defect can be small but the global error large, where defect control becomes prohibitively expensive or impractical. Part of the reason for this is pointed out in chapter 4.

[6] The notion of a well-enough conditioned problem will be discussed in section 5.1.

defect using a Lobatto quadrature formula. They also point out that that the acceptance test on the size of the defect automatically takes into account how well the collocation equations are satisfied. This is a distinct advantage of defect-based methods for BVP.

Enright & Hayashi (1998) develop a generic approach using CRK to solve retarded and neutral delay differential equations (DDE) using defect control. They are able to show that the global error of the numerical solution is controlled both efficiently and reliably by controlling the size of the defect and using discontinuity detection. In (Enright & Hayashi, 1997), they present a method DDVERK that implements their approach. A more recent package for DDE using defect control is `ddesd` developed by Shampine (2005), which is now part of the Matlab PSE. `ddesd` does not solve neutral DDE and does not track discontinuities, but it is robust and accurate and it has a much simpler interface than DDVERK. It also includes the capability to deal with event location and restarts of the integration.

Most of the backward error approaches to the numerical solution of differential-algebraic equations (DAE) have used the method of modified equations (see section 2.3), but two recent Ph.D. theses from the University of Toronto, Nguyen (1995) and MacDonald (2000), have considered the use of defect control for DAE. Nguyen (1995) considered, among other things, defect-based error control strategies suitable for classes for index 2 and index 3, where the index is the number of derivative evaluations of the equation defining the problem required, semi-explicit DAE. MacDonald (2000) implements a 'least squares' CRK method that provides a continuous approximation for the solution of the DAE without assuming that the problem has a special form or that the index of the problem is known. He investigates and justifies a defect control and stepsize choosing strategy based monitoring and bounding the defect.

2.2 Shadowing

Let $y(t)$ be the exact solution to an IVP

$$\dot{y}(t) = f(y), \quad y(t_0) = y_0.$$

A numerical method will produce a sequence y_n of discrete points representing approximations to $y(t_n)$, where $t_{n+1} = t_n + h_n$. Such a sequence is called a *pseudo-trajectory*. Let $u(t)$ be a piecewise differentiable interpolant of the pseudo-trajectory y_n. The idea of shadowing is to show that there exists an exact solution $s(t)$ (the shadow) that remains uniformly close to the pseudo-trajectory interpolant $u(t)$ but having a slightly different initial condition, *i.e.*,

$$\dot{s}(t) = f(s(t)), \qquad \|s(t) - u(t)\| < \varepsilon,$$

for a nontrivial time interval $[t_0, T]$.

Consider, for simplicity, a fixed time-step h. Since a numerical method produces an approximation to a discrete orbit of the evolution homeomorphism φ_h, or time h flow, of an ODE rather than a continuous solution,[7] the numerical

[7] Whether or not φ_h is a diffeomorphism depends on the smoothness of the vector field f.

solution y_n is considered a *δ-pseudo orbit.*

Definition 2.3 (δ-Pseudo Orbit) A numerical solution y_n to (1.1) is a *δ-pseudo orbit*, or a *noisy orbit*, if

$$\|y_{n+1} - \varphi_h(y_n)\| \leq \delta, \qquad i \leq n \leq j,$$

where δ is the *noise amplitude.* Thus, a δ-pseudo orbit is δ away from being an exact orbit of φ_h, or that the local error per step is less than δ.

Shadowing results involve showing an exact orbit ε-shadows a pseudo-trajectory.

Definition 2.4 (ε-Shadowing) An exact orbit x_n of φ_h, $i \leq n \leq j$, *ε-shadows* a pseudo-trajectory, if

$$\|y_n - x_n\| \leq \varepsilon, \qquad i \leq n \leq j.$$

Shadows of a pseudo-trajectory may only exist for a period of time, and a pseudo-trajectory is said to have a glitch if a shadow only exists for a finite amount of time.

Definition 2.5 (Glitch) A pseudo-trajectory is said to have a *glitch* at some point $n = G$ if there is some relevant ε such that an exact trajectory x_n ε-shadows y_n for $i \leq n \leq G$ but no such exact trajectory exists for $n > G$.

Rarely can it be rigorously proved that such a glitch has occurred,[8] since in practice it is possible that what appears to be a glitch is just a failure of

[8] This may be done, for example, by showing that the numerical method produces points that are outside of the domain on which φ_h acts.

the given method to find a shadow. For this reason, Hayes (2001) suggests that a failure of the method should be called a *soft glitch* and and the actual nonexistence of a shadow called a *hard glitch*.

Anosov (1967) is credited with originating the shadowing technique. In that paper he showed for hyperbolic systems on compact manifolds that for any $\varepsilon > 0$ there is a $\delta > 0$ such that every infinite-length δ-pseudo orbit remaining in an invariant set S of a map φ is ε-shadowed by a true trajectory in S.

Definition 2.6 (Hyperbolic System) A dynamical system is *hyperbolic* if all solutions to the variational equation can be divided into two classes, those that contract exponentially and those that expand exponentially.[9]

Bowen (1975) proved a general shadowing result for hyperbolic sets of diffeomorphisms, which was extended by Franke & Selgrade (1977) for hyperbolic sets of vector fields. Odani (1990) proved a shadowing result for generic homeomorphisms φ on compact differentiable manifolds of dimension 3 or smaller.

Definition 2.7 (Generic Property) A *generic* property on a topological space is one that holds on a dense open set, or more generally on a residual set, where a set is *residual* if it is the intersection of countably many sets with dense interiors.[10]

[9] A more technical definition is the following. A subset X of the phase space of a dynamical system has a *hyperbolic structure* with respect to a smooth vector field f if the tangent space at each point in X can be decomposed into the direct sum of two invariant subspaces, one stable and one unstable. A similar definition applies for φ a diffeomorphism on a compact smooth manifold.

[10] If a topological space is a Baire space then every residual set is dense. Thus, from the Baire category theorem, residual sets are dense for every complete metric space and every locally compact Hausdorff space.

Although the hyperbolicity conditions required are quite involved, Chow & Van Vleck (1992) prove a similar theorem where the map φ can change at each step, which therefore has consequences for variable step-size methods.

Most systems in practice are not uniformly hyperbolic, though many have enough hyperbolicity along trajectories in the vicinity of the pseudo-orbit so that finite-time shadowing results can be proved. Systems that have this property are called *pseudo-hyperbolic*.

Definition 2.8 (Pseudo-Hyperbolic System) A dynamical system is *pseudo-hyperbolic* if some portion of the phase space has a hyperbolic structure, *viz.*, if for some period of time the solutions to the variational equation can be divided into exponentially contracting solutions and exponentially expanding solutions.

Such finite-time shadowing was demonstrated for the logistic and Hénon map by Hammel *et al.* (1987). Coven *et al.* (1988) and Nusse & Yorke (1988) also consider 1-dimensional maps. Hammel *et al.* (1988) considers 2-dimensional maps. A general finite-time shadowing result for maps was proved by Chow & Palmer (1991, 1992).

Although the initial studies came from the dynamical systems literature and did not specifically consider systems of ODE, Coomes *et al.* (1994b) prove a finite-time shadowing result for systems of autonomous ODE. Eirola (1993) is an early study of shadowing methods in the context of one-step methods applied to ODE over infinite-time lengths. Results are proven for linear systems and for solutions in the vicinity of hyperbolic steady-state solutions. Also, Corless & Pilyugin (1995) showed that generic systems of ODE on compact smooth manifolds of arbitrary dimension have a slightly weaker tracing property than the pseudo-orbit tracing property, *viz.*, that the computed trajectory of the modified problem is contained in an ε-neighbourhood of some trajectory of the specified problem.

Since, in general, systems are not hyperbolic, in practice we only expect to establish finite-time shadowing results. The first studies of shadowing for nonhyperbolic systems of are due to Hammel *et al.* (1987, 1988) for chaotic maps, as mentioned above, and Grebogi *et al.* (1990) for chaotic systems of ODE. The motivation of these papers is to show that shadows exist for nontrivial periods of time in chaotic dynamical systems, even though roundoff error ensures that the computed trajectory differs significantly from the exact one after a small number of iterations of a numerical method. The method they use consists of two parts: *refinement*, where one uses an iterative method similar to Newton's method to refine a noisy trajectory to produce a nearby trajectory with less noise; and *containment*, where one can prove the existence of an uncountable number of nearby exact trajectories. Refinement is important since the less noisy the trajectory the longer a shadow can be shown to exist. The algorithm is discussed in section 3.2.

Now, for dynamical systems in general it is sufficient to demonstrate the existence of a shadow, *i.e.*, the existence of an ε-nearby true orbit with different initial conditions. In the context of ODE, however, this is insufficient, since it matters not only that the computed pseudo-trajectory follows the *path* of a true trajectory but also that the two trajectories are *synchronized.* For example, when modeling a periodic orbit we require not only that the true trajectory and pseudo-trajectory are ε-close but also that the time at which the true trajectory completes an orbit is ε-close to the time at which the pseudo-trajectory does. To obtain this stronger type of shadowing result, time rescaling is necessary. A number of approaches to time rescaling have been developed. Hayes & Jackson (2005) give synopses of several methods, including: Van Vleck (1995) who uses an explicit rescaling by adding time to the variational equation; Coomes *et al.* (1994b, 1995) who present an implicit rescaling method based on finding points on a nearby true orbit lying on hyperplanes perpendicular to the direction of motion, *i.e.*, the vector field; and Hayes (2001) and Hayes & Jackson (2003) who develop a similar idea, where it is shown that the true solution passes through a hyperplane containing each point of the pseudo-trajectory in a small time interval. Using modifications of these shadowing techniques for ODE, results can be proved for periodic trajectories, where time rescaling is crucial to prevent persistent growth in time error (see, *e.g.*, Coomes *et al.* , 1994a, 1997; Van Vleck, 1995).

An important and subtle issue that arises with the shadowing results is whether the exact solution that shadows a numerical one is typical of true orbits chosen at random. It could be the case that the exact solution is not generic, but the solutions that shadow the numerical one are. In this case the behaviour of the exact solution could be qualitatively different than that of the shadowing solutions. Quinlan & Tremaine (1992) showed that there are cases where this happens. There are a number of discussions of this cited in Hayes & Jackson (2005), including Corless (1994b).

Because shadowing computations are so expensive and do not scale well, applications of shadowing tend to work only for small, *i.e.*, low dimensional, problems. The largest problems for which shadowing results have been proved are studies of the gravitational n-body problem. They are good systems to study since they are theoretically relatively simple yet display a wide variety of dynamical behaviour. For references to several of these studies see (Hayes & Jackson, 2005, 316-317).

The main shadowing results apply to IVP, but there are a few studies of the method for BVP, DDE and DAE. Coomes (1997) proves a theorem specifying shadowing conditions for problems where the solution is restricted to some submanifold, *e.g.*, DAE. For work on the use of shadowing in the study of singular BVP see Liu (2005) and Lin (1989). Work has also been done by Al-Nayef *et al.* (1997) on shadowing for neutral type DDE.

2.3 The Method of Modified Equations

Since the difference equations that define a numerical method can be difficult to analyze directly, a better understanding of the method can be obtained by generating a modified problem, the exact solution of which is closer to the numerical solution than the numerical solution is to the exact solution of the original equation. With such a modified problem in hand, an explanation of the *qualitative* behaviour of the numerical method is possible by means of a qualitative analysis of the modified equation. For example, things like a qualitative change in the dynamics compared to the exact solution and the emergence of spurious limit sets can be explained. Although the method is called the "method of modified equations," the method generally does require modified *problems*, since in general any initial, boundary or algebraic condition must also be modified.[11]

The usual strategy to obtain a modified problem is to find an equation which has an exact solution with zero local error by perturbing about the local exact solution $z_n(t)$ (*cf.* equation (1.6)). This is accomplished by manipulating the expression for the local truncation error that is furnished by the numerical method. Restricting attention to fixed time-step one-step methods for simplicity, the numerical method with step-size h applied to an IVP will provide a formula of the form

$$y_{n+1} = y_n + hI(y, t; h),$$

where $I(y, t; h)$ is an increment function. If the method is of order s it has a LEPUS that is $O(h^s)$. In order to obtain a modified equation from the expression for the LEPUS we can expand $z_n(t_n + h)$ in a Taylor series about $z_n(t_n)$ and use the expression for y_{n+1} given by the method. The result of this is

$$\begin{aligned} e_{n+1} &= \frac{[y_n + h\dot{y}_n + \frac{h^2}{2}\ddot{y}_n + \cdots] - [y_n + hI(y_n, t_n; h)]}{h} \\ &= \dot{y}_n - I(y_n, t_n; h) + \tfrac{h}{2}\ddot{y}_n + \tfrac{h^2}{6}\dddot{y}_n + \cdots . \end{aligned}$$

Since we seek a perturbation about the local exact solution with zero local truncation error, the method of modified equations therefore starts with the

[11] I say "in general" here because defect analysis is properly considered a special case of the method of modified equations, but there the initial condition is left fixed.

equation

$$0 = \dot{x} - I(x,t;h) + \tfrac{h}{2}\ddot{x} + \tfrac{h^2}{6}\dddot{x} + \cdots . \tag{2.6}$$

Since equation (2.6) is a singular perturbation of the original ODE some manipulation is required. There are two parts to the manner in which this is usually done. Suppose that we seek a modified equation that is $O(h^p)$ close to the numerical solution, where $p > s$. First we truncate the equation to $O(h^p)$ and then we differentiate it to obtain additional equations which allow us to eliminate the higher derivatives. The result of this will be an equation

$$\dot{x} = f(x,t) + h^s F(x,t),$$

where $F(x,t)$ is an explicit function of x and t that we find, the solution of which is $O(h^p)$ close to the numerical solution. This equation can then be examined in order to better understand the behaviour of the numerical solution. As was indicated above, it really is a modified *problem* that is generated, since any initial, boundary or algebraic condition must also be ensured to be satisfied to the same order p.

As an example of the derivation of a modified equation consider the fixed time-step forward Euler method, which is order 1. We will compute the second order modified equation. Truncating equation (2.6) to second order we obtain

$$\dot{x} - f(x,t) + \tfrac{h}{2}\ddot{x} = 0. \tag{2.7}$$

Differentiating this we obtain

$$\ddot{x} = J_f(x,t)\dot{x} + f_t(x,t) - \tfrac{h}{2}\dddot{x},$$

where $J_f(x,t)$ is the Jacobian of $f(x,t)$. Substituting the resulting expression for $\ddot{x}$ into the equation (2.7) we obtain

$$\dot{x} - f(x,t) + \tfrac{h}{2}(J_f(x,t)\dot{x} + f_t(x,t) - \tfrac{h}{2}\dddot{x}) = 0.$$

Substituting the expression for $\dot{x}$ from (2.7) into this yields

$$\dot{x} - f(x,t) + \tfrac{h}{2}\left(J_f(x,t)(f(x,t) - \tfrac{h}{2}\ddot{x}) + f_t(x,t) - \tfrac{h}{2}\dddot{x}\right) = 0.$$

Neglecting $O(h^2)$ terms then yields

$$\dot{x} - f(x,t) + \tfrac{h}{2}J_f(x,t)f(x,t) + \tfrac{h}{2}f_t(x,t) = 0.$$

We therefore obtain the modified equation

$$\begin{aligned} \dot{x} &= f(x,t) - \tfrac{h}{2}J_f(x,t)f(x,t) - \tfrac{h}{2}f_t(x,t) \\ &= \left(I - \tfrac{h}{2}J_f(x,t)\right)f(x,t) - \tfrac{h}{2}f_t(x,t). \end{aligned}$$

In the case of autonomous systems this reduces to

$$\dot{x} = \left(I - \tfrac{h}{2}J_f(x)\right)f(x). \tag{2.8}$$

A limitation of having *finite-order* modified equations, *i.e.*, where p is finite, is that although the solution to the modified problem follows the numerical solution more closely than the solution to the original problem does, the modified equation could be of no use in understanding the long-time asymptotic behaviour of the numerical solution. An approach that can be useful for such an analysis is to try to find an *infinite-order* modified equation. This can be sometimes done (Corless, 1994a) using the same approach as before except that the series in equation (2.6) is not truncated and one seeks a pattern in the equations obtained by differentiating (2.6) to eliminate higher derivatives, thereby obtaining a modified equation with an infinite series, called a *B-series*,[12] a notion introduced by Hairer & Wanner (1974). B-series can also be constructed in a more formal way (Calvo *et al.* , 1994) based on the method of analyzing one-step methods using (rooted) trees (see, *e.g.*, Butcher, 1987). If the infinite series can be summed and/or efficiently computed then the infinite-order modified equation can be useful for investigating the long-time asymptotics of the numerical method on the problem at hand.

The origin of the method is in the study of numerical methods for the solution of partial differential equations (PDE). Its origins go back as far as Garabedian (1956), where the method was used to analyze successive overrelaxation methods for the solution of finite-difference approximations to elliptic PDE. An early study of the method itself appears in (Hirt, 1968), which examines the method as it is used to investigate the computational stability of finite-difference equations. Other papers examining the method are (Warming

[12] The 'B' is for Butcher.

& Hyett, 1974) and (Morton, 1977). Warming & Hyett (1974) show how a finite-order modified equation can be used to determine necessary and sufficient conditions for computational stability, and how it can be used to gain insight into the nature of both dissipative and dispersive errors. Morton (1977) appears to be the first discussion of the method as applied to initial-boundary-value problems. References to many of the early papers that use the method in the investigation of the properties of partial difference schemes for PDE, particularly their dispersive and dissipative properties, are available in Griffiths & Sanz-Serna (1986).

The first consideration of the range of applicability and the shortcomings of the method is due to Griffiths & Sanz-Serna (1986). Through a careful examination of a few particular numerical methods applied to simple ODE and PDE, they drew a number of conclusions about the method, which include: one must ensure that the derivatives in the $O(h^p)$ remainder in the finite-order modified equation are bounded as $h \to 0$; the numerical method being analyzed must be numerically stable as $h \to 0$ to ensure that estimates of the local error imply estimates of the global error; and that the finite-order modified equation cannot be used to infer fixed h long-time stability and stability as $h \to 0$ for arbitrary finite times. They also made the point about modified problems, *viz.*, that all the side conditions for the differential equation must be satisfied to the same order as the modified equation.

From the point of view of the method as applied to ODE, one of the earliest papers is a paper in the dynamical systems literature by Feng (1991). Treating vector fields and flows as formal power series, he proves structure preservation theorems relating near identity formal maps derived from a formal vector field. He points out that since numerical methods can be characterized in terms of a near identity map depending on the step size and approximating the flow of the original system, his theory has "implications for the construction, analysis, assessment and understanding of numerical methods, particularly those applied to Hamiltonian, Liouville and contact systems." And, indeed, many of the earliest studies and uses of the method of modified equations for ODE are in the context of its use for geometric integrators, some of which are mentioned below. Another use of the method of modified equations in the context of dynamical systems is (Reddien, 1995), which is a study of the stability of a variety of Runge-Kutta and Adams methods at weakly attracting equilibria, weakly attracting periodic solutions and Hopf points.

As indicated above, the method of modified equations is important in the context of geometric numerical methods since it is useful for showing that the flow of the modified equation exactly solved by a particular numerical method possesses the structural features of the flow of the original equation that are relevant to the problem being solved, *e.g.*, symplecticness (or symplecticity), symmetry, energy conservation, reversibility, integral invariants, *etc.* This allows one to explain interesting phenomena such as the almost conservation of energy, the linear error growth in Hamiltonian systems, and the existence of periodic solutions and invariant tori. An early example of the use of the method of modified equations for this purpose is a paper by Hairer (1994), who uses the method of modified equations to show that for a wide variety of symplectic integrators applied to Hamiltonian problems, the modified problem is also Hamiltonian.

Since the topic of geometric integrators has a large literature, and it is somewhat tangential to the current focus, we only provide a selection of works in this area. Sanz-Serna (1992) provides an early survey of the use of the method of modified equations for symplectic integrators, *i.e.*, integrators that preserve the symplectic structure of the flow of a system of ODE. Two major classes of symplectic integrators are considered: the subset of the standard Runge-Kutta or Runge-Kutta-Nyström methods that can be shown to be symplectic; and those based on generating functions, which includes Hamilton-Jacobi methods, applicable to Hamilton-Jacobi equations. In addition to carefully explaining the notions of 'symplecticness' and symplectic integrators, Sanz-Serna also discusses the general properties of symplectic integrators and provides a summary of their practical performance. For a more recent survey of the structure preserving integration of Hamiltonian systems, see Hairer (2005). Calvo *et al.* (1994) is another introductory article, which, as mentioned above, uses the analysis of numerical methods using rooted trees to present the method in terms of the use of formal B-series. The method is then illustrated in an application to ODE. Sanz-Serna & Calvo (1994) devote a book to the subject.

Reich (1997) provides the first application of the method of modified equations to constrained Hamiltonian systems, which are important in the physics literature. It is based on an extension of the integrator to an open neighbourhood of $\mathcal{M}$ (the constraint manifold) so that standard techniques can be applied. As well as being an excellent introduction to geometric numerical methods, Hairer *et al.* (2006) provide another method for BEA of constrained

Hamiltonian systems. These two methods, however, cannot guaranteee a globally defined modified Hamiltonian. Hairer (2003) shows how this can be done for partitioned Runge-Kutta methods.

Problems of ODE on manifolds need not be formulated in terms of an ODE together with a constraint. Such problems can also be formulated as dynamical evolution in a Lie group. Faltinsen (2000) extends the method to differential equations on manifolds using Lie group methods. If the Lie algebra is nilpotent a global stability analysis can be done in the Lie algebra. In the general case, however, this linear analysis fails. In order to show that there is a perturbed differential equation on the Lie group with a solution that is exponentially close to the numerical integrator after several steps, he proves a generalized version of the Alekseev-Gröbner theorem (Theorem 4.1, p. 73), which implies many stability properties of Lie-group methods.

Turning to discussions of the method of modified equations itself, Hairer & Lubich (1997) study the influence of the truncation of the formal modified equation to the difference between the numerical solution and the exact solution of the perturbed equation. They obtain results on the long-time behaviour of numerical solutions and consider applications to phase portraits near various equilibria and steady-state solutions. Reich (1999) aims at providing a unifying framework and a simplification of the existing results and corresponding proofs on the long time behaviour of numerical integration methods. Unlike previous methods, the BEA is based on a recursive definition of the modified vector field so that explicit Taylor expansions are not required.

Some developments for geometric integrators are the following. Gonzalez *et al.* (1999) introduce a technique that can be used to prove the modified equations inherits a qualitative property from the underlying system of ODE. The technique applies to arbitrary one-step methods and unifying and extending similar results proved in other ways. In a recent paper, Chartier *et al.* (2007) develop high-order, structure-preserving numerical integrators for ODE using modified differential equations, with an emphasis on methods represented explicitly as B-series. Bond & Leimkuhler (2007) extend the standard BEA for Hamiltonian systems to systems involving collisions, which introduce intermittent impulses into continuous evolution.

It was originally thought that symplectic integrators had to be fixed time-step methods, since varying the time-step would not be compatible with structure preservation. However, Hairer (1997) developed a way of combining variable time-step with symplectic integrators and justifies the method using BEA. More recently, Hairer & Soderlind (2005) develop completely explicit, reversible, symmetry-preserving, adaptive step-size selection algorithms for geometric numerical integrators. They use BEA and reversible perturbation theory to analyze a new step density controller. They are able to preserve structure and the excellent long-time behaviour of constant-step methods, but with the added accuracy and efficiency of multistep methods. Blanes & Budd (2005) use modified equations to analyze variable time-step geometric integration methods and determine certain limitations of the methods.

Chapter 3

Numerical Methods for Ordinary Differential Equations Using Backward Error

3.1 Defect Control

The defect can be computed for any numerical method by constructing a suitable interpolant $u(t)$ of the solution y_n, such that $u(t_n) = y_n$. To illustrate how this can be done, consider the non-autonomous problem

$$\dot{y} = f(y,t) = \cos(\pi t y), \qquad y(0) = y_0. \tag{3.1}$$

solved using the implicit trapezoidal rule. This gives us values y_n of the solution at times t_n, and it computes the values of $\dot{y}(t_n)$ at each solution time. Consider a single interval $[t_n, t_{n+1}]$. In case that the numerical method does not compute the values of $\dot{y}(t_n)$, we can use the expression for $\dot{y}$ in (3.1) to evaluate the derivatives of the solution at the end points; these are already calculated by the implicit trapezoidal rule, making its use computationally efficient. Then we can use cubic Hermite interpolation to construct an interpolant $u_n(t)$ over each interval $[t_n, t_{n+1}]$. This generates a piecewise cubic Hermite interpolant $u(t)$ over the entire interval of integration $[t_0, T]$. Moreover, this interpolant is C^1, since it is constructed to match derivatives at the end points of successive intervals. The implicit trapezoidal rule is the same as the two-stage theta method, which has the Butcher tableau

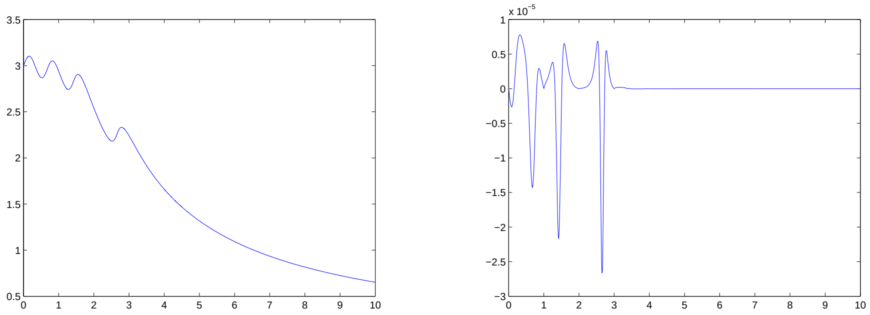
$$\begin{array}{c|cc} 0 & & \\ 1 & 1-\theta & \theta \\ \hline & 1-\theta & \theta \end{array}$$

for $\theta = \frac{1}{2}$. This method is implemented in the Matlab code `theta2`, provided in appendix A.2. Using this method with the step-size $h = 0.001$ and $y_0 = 3$, we obtain the solution shown in figure 3.1(a).[1] Since the equation for the piecewise cubic Hermite interpolant on each interval $[t_n, t_{n+1}]$ is

$$\begin{aligned} u_n(t) = (\theta-1)^2(2\theta+1)y_n + \theta(\theta-1)^2 h_n f(y_n, t_n) + \\ \theta^2(-2\theta+3)y_{n+1} + \theta^2(\theta-1)^2 h_n f(y_{n+1}, t_{n+1}), \end{aligned} \tag{3.2}$$

in local coordinates $\theta = (t - t_n)/h_n$, and we are assured that the interpolant $u(t)$ is C^1, we are able to compute the derivative of the interpolant. The Matlab function `pchi` (code provided in appendix A.3) computes the interpolant $u(t)$ and its derivative at whichever points along the integration interval one chooses. Using this function to compute the defect at 1000 points along the

[1] The function `theta2` uses Newton iteration to compute each time step, the tolerance for which was set to 10^{-8}.



(a) The IVP (3.1) solved using the implicit midpoint rule.

(b) The defect of the solution in 3.1(a) computed using piecewise cubic Hermite interpolation.

Figure 3.1: An example of computing the defect.

interval we find the defect plotted in figure 3.1(b). It follows from this plot that the numerical solution is the exact solution of the problem

$$\dot{u} = \cos(\pi t u) + 10^{-4} v(t), \qquad u(0) = u_0,$$

where $\|v(t)\| \leq 1$. Although this example does not use defect control, the same procedure can be used for any variable time-step method, since the procedure only requires a solution sequence y_n and the values of $\dot{y}(t_n)$ at each time-step t_n.

As an example of a defect control algorithm we consider a defect-controlled Euler method.[2] Consider, for simplicity, the scalar version of the ODE (1.1) and the local variable $\theta = (t - t_n)/h_n$. Let $k_0 = f(y_n, t_n)$ and $k_1 = f(y_{n+1}, t_{n+1})$.

[2] The code for `ode1d`, a MATLAB implementation of the following defect-controlled Euler method, is provided in appendix A.1.

Then one step of the Euler method is

$$y_{n+1} = y_n + h_n k_0. \tag{3.3}$$

Now, to analyze the defect for this method we need an interpolant, and we can obtain an asymptotically valid estimate of the defect using the interpolant

$$u_n(t) = y_n + h_n(1 + \theta - \theta^2)\theta k_0 + \theta^2 h_n(\theta - 1)k_1, \tag{3.4}$$

which can be obtained from equations (3.2) and (3.3). By evaluating the Taylor expansion of $f(y,t)$ at (y_n, t_n) and (y_{n+1}, t_{n+1}) it can be shown that

$$f(y,t) = k_0 + \theta(k_1 - k_0) + O(h_n^2),$$

where $k_1 - k_0$ is $O(h_n)$. Using this expression for $f(y,t)$ and the interpolant, it can then be shown that

$$\delta(t) = 3\theta(\theta - 1)(k_1 - k_0) + O(h_n^2).$$

Since for a single step we are interested in the interval $\theta \in [0,1]$, we thus have that, asymptotically (as $h_n \to 0$),

$$\delta(t) = 3\theta(1 - \theta)(k_1 - k_0). \tag{3.5}$$

It is easily seen that the maximum of this function occurs at $\theta = \frac{1}{2}$. Therefore substituting $\theta = \frac{1}{2}$ into (3.5) we obtain an asymptotically valid bound on the defect,

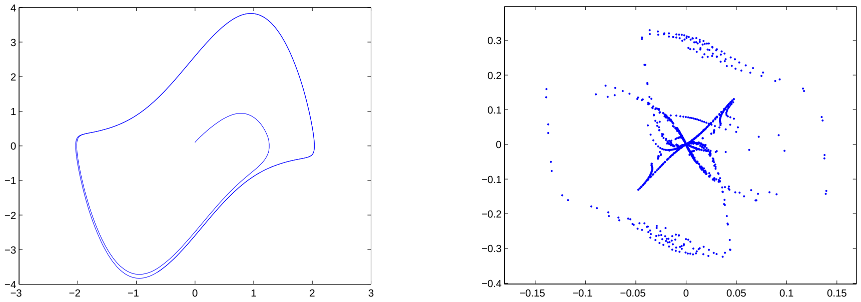
$$\|\delta(t)\|_\infty \approx \tfrac{3}{4}\|k_1 - k_0\|_\infty \leq \varepsilon, \tag{3.6}$$

which we try to ensure is less than the user-specified tolerance ε. Thus, to determine whether the step will be accepted or not we test to see whether or not the asymptotic bound is less than the tolerance. It provides an efficient test since, although k_1 is not needed for the current step of the Euler method, it will be required for the next step and so it would have to be computed anyway. Higher-order methods require more work. Intermediate values of the solution would then be computed using the interpolant (3.4). If the condition (3.6) is not satisfied, then a heuristic can be used to adjust the step-size, which would then be checked again, or step-size control algorithms using linear digital control theory can be used (see Söderlind, 2003) in much the same way as for local error controlled codes. The control condition here is trivially extended to allow vector values. Interpolation of the solution is then done componentwise. It should be mentioned here that the defect-controlled Euler method is not intended for practical use. It is, however, useful as a simple example of how a defect control algorithm can be implemented and for conceptual exploration.

As an illustration of the use of this defect-controlled Euler method, consider the unforced van der Pol equation

$$\ddot{x} + \epsilon(x^2 - 1)\dot{x} + x = 0, \qquad y(0) = y_0. \tag{3.7}$$

This is written in the form $\dot{y} = f(y,t)$ in the usual way by letting $y = (y_1, y_2)^T = (x, \dot{x})^T$. We will solve this equation for the parameter value



(a) The IVP (3.7) solved using `ode1d`.

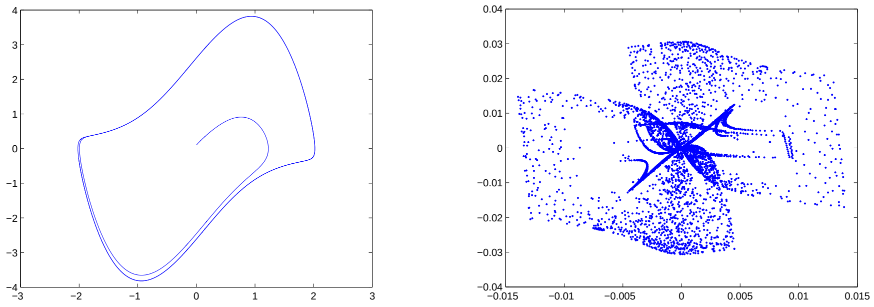
(b) Estimated maximum defect on each integration step of the solution in 3.2(a) computed using piecewise cubic Hermite interpolation.

Figure 3.2: Van der Pol equation solved using the defect-controlled Euler method `ode1d` with tolerance $\varepsilon = 10^{-1}$.

$\epsilon = 2$ using `ode1d`, a MATLAB implementation of the above described defect-controlled Euler method (code provided in appendix A.1). Setting the tolerance $\varepsilon = 10^{-1}$, we obtain the numerical solution plotted in figure 3.2(a). Using `pchi` to compute piecewise cubic Hermite interpolants for each component of the solution and estimating the maximum defect on each integration step, we find that the maximum defect is of the form seen in figure 3.2(b).[3] We observe from figure 3.2(b) that the method `ode1d` is controlling the defect properly, since the maximum value of the defect is seen to be of the same order of magnitude as the tolerance $\varepsilon = 10^{-1}$. That this is not coincidence is seen by reducing the tolerance to $\varepsilon = 10^{-2}$ and repeating the same kind of computation. The results of this are seen in figure 3.3.

Although there are good reasons to use defect control for IVP, such as the relative problem independence of error estimates as compared to local

[3] This computation is described in more detail in section 5.1.



(a) The IVP (3.7) solved using `ode1d`.

(b) Estimated maximum defect on each integration step of the solution in 3.3(a) computed using piecewise cubic Hermite interpolation.

Figure 3.3: Van der Pol equation solved using the defect-controlled Euler method `ode1d` with tolerance $\varepsilon = 10^{-2}$.

error control and its effectiveness on chaotic problems (*cf.* section 5.1), the main packages that use it are for solving BVP and DDE. Many of the earlier defect control algorithms, however, are for IVP. An early paper considering numerical methods with defect control is by Hanson & Enright (1983), who consider algorithms for controlling the defect for variable order Adams methods for IVP. Since multistep codes produce a (piecewise) polynomial interpolant of the numerical solution, they are easily modified to compute the defect. One of the advantages of the defect approach, namely that it can provide an inexpensive estimate of the error that is not particularly sensitive to the problem at hand, is made clear by showing that the defect could be directly related to the local error, and hence the user prescribed tolerance. There are also the papers by Enright (1989a,b) mentioned in the previous section. One of the defect control algorithms in Enright (1989a) uses an asymptotically valid estimate of the defect, which is important for the selection of optimal points for the evaluation of the defect. The problem of finding interpolants for which one has information about the asymptotic behaviour of the defect is addressed more fully in Higham (1989b). Here two classes of interpolant are presented for which the asymptotic behaviour of the interpolant is known, which allows the selection of optimal points of evaluation, yielding asymptotically correct estimation of the maximum value of the defect over each time-step.

One other important algorithm using defect control for IVP is the one described by Corless & Corliss (1992). Since a major part of the motivation for the defect control approach is that the backward error point of view allows one to regard one's numerical solution as the exact solution of a nearby problem, it is important to be able to have a guaranteed bound on the defect when one needs one in order to ensure that one has an ε-nearby problem. The algorithms outlined by Corless & Corliss (1992) take as input initial and final times t_0, T, and initial conditon y_0 and a tolerance $\varepsilon = \|\delta(t)\|$. The output are the nodes t_n, the boundaries of the steps, a continuous $u(t)$ that solves (1.3) exactly and guarantees that $\|\delta(t)\| \leq \varepsilon$ for all $t \in [t_0, T]$.

The algorithm can be adapted for a variety of numerical methods for solving ODE. The main loop of the algorithm involves the following. A continuous approximate solution $u(t)$ is computed on $t_i \leq t \leq t_n + h_n$ using a continuous numerical method or as an interpolant obtained from a discrete method. The defect is then computed, which is easily done using a polynomial interpolant, and an enclosure Δ of $\|\delta(t)\|$ is computed. This is the only part of the algorithm that uses interval arithmetic. Then if $\Delta > \varepsilon$ the step is rejected and h_n is reduced, if $\Delta \ll \varepsilon$ the step is rejected and h_n is increased, and otherwise the step is accepted. The method described to compute the bound Δ uses interval Taylor operators, which achieve tight bounds on the range of the defect and its derivatives. For more details see (Corless & Corliss, 1992, 7).

As mentioned above, defect control has become a common method for the solution of BVP and DDE for ODE. Since these problems are not the focus of this work, the algorithms will not be discussed in detail. An advantage of the use of defect control for BVPODE is that it is better able to deal with poor guesses for the mesh and for the solution. Any method for the solution of BVP that produces a continuous approximation of the solution can benefit from the use of defect control. A common approach to the solution of BVPODE is the use of collocation methods, which have been used for the codes COLSYS (Ascher *et al.* , 1979) and its descendent COLNEW (Bader & Ascher, 1987). In the standard implementation of collocation methods, a continuous approximation of the solution over entire interval of the problem is generated. This makes it easy to assess the quality of the solution by computing the defect. Enright & Muir (1996) use an alternative approach using CMIRK methods, which yields an inexpensive interpolant of the solution obtained on the mesh. Using the continuous interpolant obtained from the method, defect control is used to determine when to terminate and how to redistribute the mesh, rather than using global error control for these purposes. The algorithm for implementing this approach is implemented in the FORTRAN 77 code MIRKDC.

More recently, Muir *et al.* (2003) have implemented a parallel version of MIRKDC, called PMIRKDC. The main computational cost of the MIRKDC algorithm involves the treatment of large almost block diagonal (ABD) linear systems, treated using sequential ABD software COLROW. PMIRKDC replaces this with the parallel ABD software RSCALE, and parallelizes the setup of the ABD systems and the solution interpolants. The parallelization of the MIRKDC code is able to produce almost linear speed ups in execution time.

A major issue with software packages for the solution of ODE has been that the user interfaces have been very complicated and difficult for users to learn, particularly code developed using FORTRAN. Concern for this issue in defect control software is not new, going back at least as far as the software package DEPAC developed by (Shampine & Watts, 1980), but there has been a renewed interest in this more recently.[4]

Simplification of the user interface was a major concern in the development of `bvp4c` by Kierzenka & Shampine (2001) for the Matlab PSE. This package allows for the solution of BVP with non-separated boundary conditions that can depend on a vector of unknown parameters, and analytical partial derivatives are not required. It also produces inexpensive, asymptotically correct L_∞ and L_2 norm estimates of the defect. Kierzenka & Shampine (2001) point out that `bvp4c` can be regarded as a collocation method, and that the acceptance test based on the size of the defect automatically takes into account how the collocation equations are satisfied. `bvp4c` is also a vectorized code and includes an option to speed up the solution of a problem by vectorizing the evaluation of the vector field.

[4] For a discussion of design issues for ODE solvers in general see (Enright, 2002) and (Shampine, 2007).

Shampine *et al.* (2006) developed the software package BVP_SOLVER to be a more user friendly version of MIRKDC. Not only does this software extend MIRKDC to problems with unknown parameters and ODE problems with a singular coefficient, it also uses more efficient CRK methods and includes auxiliary routines for evaluating the solution and its derivative and for extension of the problem interval. In order to emulate the convenience of `bvp4c` in FORTRAN, it was necessary to utilize the capabilities of FORTRAN 90/95. This enables a radical reduction in the number of user-supplied arguments and the subroutines that must be supplied by the user. The code also approximates partial derivatives using finite differences by default, but allows the user to supply a routine to do the evaluation when it is required. It also enables the code to handle memory allocation dynamically, whereas in MIRKDC this was handled by the user before running the computation. Although it is a defect control algorithm, the code also enables the estimation of the global error at mesh points.

As for IVP, the generation of interpolants with asymptotically correct estimates of the defect is important. Some codes, including `bvp4c`, as mentioned above, include interpolants that have asymptotically correct estimates, but the FORTRAN codes MIRKDC and BVP_SOLVER do not. Recent work by Enright & Muir (2010) addresses this problem by developing a method to generate interpolants such that one knows *a priori* where the maximum defect on each mesh subinterval occurs. The availability of more reliable defect estimates can also improve the mesh redistribution process, which thereby provides an improvement in the overall efficiency of the computation.

An important issue in the use of defect control methods for the solution of BVPODE is the estimation of condition numbers for the problem at hand, *i.e.*, numbers relating the size of the defect $\delta(t)$ to the size of the global error $y(t) - u(t)$. Shampine & Muir (2004) investigate a conditioning constant appropriate to BVP solvers that control residuals and an inexpensive way to estimate this constant. A key idea is the implementation of an approximate matrix norm algorithm due to Higham and Tisseur. It is demonstrated that the estimated conditioning constants are quite helpful in assessing the quality of numerical solutions obtained from a control of the defect. In particular, it enables the detection of numerical pseudosolutions to BVP that have no solution. Although this paper considers a conditioning constant, Corless has pointed out that codes should ideally provide a conditioning *function*, rather than a conditioning constant, since the ability to determine where in the problem interval a problem is poorly conditioned should be important from the point of view of selecting an optimal mesh.[5]

The fact that DDE codes produce continuous approximations to the solution makes it apparent that defect control a natural approach to take, since it is inexpensive to sample. Enright & Hayashi (1998) developed and studied a method for solving general neutral-type DDE. This approach is implemented in (Enright & Hayashi, 1997) using a sixth-order CRK method to produce DDVERK, a defect-controlled DDE solver. The details of the algorithm and numerical results are provided. More recently, defect control has been used by Shampine (2005) to develop an effective program `ddesd` for the solution of

[5] Private communication.

DDE with time- and state-dependent delays. The code is based on an explicit continuous RK method. This code has a well-designed interface, which makes it easy to formulate and solve DDE, even those with complications such as event location and restarts. `ddesd` has been incorporated into the MATLAB PSE.

3.2 Shadowing

Shadowing algorithms are based on the containment and refinement procedure developed by Hammel *et al.* (1988) for 2-dimensional maps, which therefore applies to the finite-time flow φ_{h_n} approximated by a numerical method. This procedure was generalized to n-dimensional Hamiltonian systems by Quinlan & Tremaine (1992). For simplicity we will just consider the 2-dimensional case. For a description of the 3-dimensional case, see (Hayes & Jackson, 2005, 306-307).

The algorithm outlined by Hammel *et al.* (1988) is the following: Let $x_{n+1} = \varphi(x_n)$ be a 2-dimensional homeomorphism, and let y_n be a δ-pseudo-orbit of the map, *i.e.*, $|y_{n+1} - \varphi(y_n)| < \delta, 0 \leq n \leq G$. Strictly speaking, we assume that $y_{n+1} = T(\varphi(y_n))$, where T is a truncation operator corresponding to machine roundoff error. To ensure the integrity of this calculation we compute $\varphi(y_n)$ to a higher precision than that of the truncation operator, and check that the error in $\varphi(y_n)$ does not corrupt the computation of y_{n+1}. Now, we want have a procedure to show that there exists a true orbit x_n nearby. To do this we will construct a sequence of parallelograms A_n in the neighbourhood of each y_n within which the true orbit must lie. Now, we require for each

$n \geq 1$ that $\varphi(A_n)$ maps across A_{n+1} in a particular way. Two conditions must be satisfied:

(i) $A_{n+1} \cap \varphi(A_n) \neq \varnothing$;

(ii) each A_n has distinguished (parallel) sides C_n^1 and C_n^2 such that $\varphi(A_n) \cap C_{n+1}^i \neq \varnothing$, while $\varphi(C_n^i) \cap A_{n+1} = \varnothing$, $i = 1, 2$.

The ability to satisfy these conditions depends on the pseudohyperbolicity of the map φ. Assuming that the map is hyperbolic in the vicinity of y_n for $0 \leq y_n \leq G$, then let $\hat{e}_n$ and $\hat{c}_n$ be unit vectors pointing along the expanding and contracting directions at the n-th step. From a computational point of view these vectors are the average directions of expansion and contraction. These vectors, respectively, can be found iteratively using

$$\hat{e}_{n+1} = \frac{Y_n \hat{e}_n}{\|Y_n \hat{e}_n\|}, \qquad \hat{c}_{n+1} = \frac{Y_n^{-1} \hat{c}_n}{\|Y_n^{-1} \hat{c}_n\|},$$

where Y_n is the Jacobian of φ evaluated at y_n computed at the higher precision. Because of the hyperbolicity, φ tends to suppress errors in the contracting direction $\hat{c}_n$ and φ^{-1} tends to suppress errors in the expanding direction $\hat{e}_n$. This is the reason that the expression for $\hat{c}_n$ is iterated forwards using Y_n starting with a randomly selected unit vector and the expression for $\hat{e}_n$ is iterated backwards using Y_n^{-1} starting with another randomly selected unit vector. The iteration gives $\hat{e}_n$ and $\hat{c}_n$ aligned with the expanding and contracting directions after a few iterates. When the expanding and contracting directions can no longer be resolved to machine precision, a (soft) glitch occurs, and this is the length $n = G$ of the shadow.

Now, the two sides C_n^1 and C_n^2 will be parallel to $\hat{c}_n$ and the other two sides of each parallelogram A_n will be E_n^1 and E_n^2, each parallel to $\hat{e}_n$. We will mention how the A_n can be selected shortly, but for simplicity of exposition we assume for the moment that each A_n is the same size. Given that A_n contains y_n, we ensure that condition (i) holds. Then the hyperbolicity ensures that the conditions in (ii) are satisfied since, relative to A_{n+1}, the action of φ causes the sides C_n^i to contract and be pushed further away (by the expansion) and the side E_n^i to expand and be pressed closer together (by the contraction). That the C_n^i are pushed further away under the action of φ ensures that $\varphi(C_n^i) \cap A_{n+1} = \varnothing$ holds. And that $\varphi(A_n)$ can be thought of expanding A_n in one direction and contracting it in the other ensures that $\varphi(A_n) \cap C_{n+1}^i \neq \varnothing$.

We now define

$$J_0 = \bigcap_{j=0}^{G} \varphi^{-j}(A_j) \neq \varnothing,$$

which is the region around the initial point y_0 of the pseudo-orbit such that its image under φ^n is contained in A_n for all $0 \leq n \leq G$. This gives us the idea of containment, *viz.*, since $J_0 \neq \varnothing$, there exists a family of true orbits x_n that ε-shadow the pseudo-orbit y_n, where

$$\varepsilon = \max_{0 \leq j \leq G} \left\{ \max_{x_j \in A_j} d(y_j, x_j) \right\}.$$

Once the A_n are calculated this quantity is easily calculated.

Now, to maximize the length of the shadow, *i.e.*, the size of G, we use a procedure to produce a less noisy orbit y_n^* that is uniformly close to y_n, and it will be the y_n^* that will be at the centre of the A_n. This is the refinement procedure. One way to do this, outlined by Hammel *et al.* (1988, 468) and Grebogi *et al.* (1990, 1529), is the following. Let Ξ_n be the one-step error

$$\Xi_{n+1} = y_{n+1} - \varphi(y_n), \tag{3.8}$$

which we know is bounded by δ. Then we will construct the refined orbit y_n^* by letting

$$y_n^* = y_n + \Upsilon_n, \tag{3.9}$$

where Υ_n is kept small. From equations (3.8) and (3.9) we have that

$$\Upsilon_{n+1} = \varphi(y_n^*) - \Xi_{n+1} - \varphi(y_n), \tag{3.10}$$

where $y_{n+1}^* = \varphi(y_n^*)$. Since we are keeping Υ_n small, we can take the Taylor expansion of $\varphi(y_n^*)$ about y_n, so that $\varphi(y_n^*) \simeq \varphi(y_n) + Y_n\Upsilon_n$. Thus, equation (3.10) becomes

$$\Upsilon_{n+1} \simeq Y_n\Upsilon_n - \Xi_{n+1}.$$

Using the vectors $\hat{e}_n$ and $\hat{c}_n$, we let $\Upsilon_n = \alpha_n\hat{e}_n + \beta_n\hat{c}_n$ and $\Xi_n = \xi_n\hat{e}_n + \zeta_n\hat{c}_n$. Since we can calculate $\varphi(y_n)$ using higher precision arithmetic, we can compute ξ_n and ζ_n directly. We find α_n and β_n recursively and by iterating

$$\alpha_{n+1} = |Y_n\hat{e}_n|\alpha_n - \xi_{n+1}, \qquad \beta_n = |Y_n\hat{c}_n|\beta_n - \zeta_{n+1}.$$

The computation here can be made stable by computing the β_n in the forward direction starting at $n = 0$ and the α_n in the backward direction starting at

$n = G$. In this way, a refined pseudo-orbit y_n^* is obtained that is less noisy than the original y_n.

Finally, we need a procedure to choose the A_n such that they are as small as possible, to minimize ε. Hammel *et al.* (1988) point out that this can be accomplished in the following way: Recall we are computing $\varphi(x)$ to a higher precision than the other quantities. Let $\varphi(x)^*$ be the computed value using the higher precision. This way we will have that $|\varphi^*(x) - \varphi(x)| < \delta^*$. Then the A_n can be made to satisfy conditions (i) and (ii) by ensuring that $\text{dist}(\varphi^*(E_n^i), E_{n+1}^i) > \delta^*$ and that $\text{dist}(C_n^i, \varphi^{*-1}(C_{n+1}^i)) > \delta^*$, where again we exploit the hyperbolicity to suppress errors. These conditions ensure that the *true* image of A_n overlaps A_{n+1} in the way required to satisfy the conditions.

There are questions concerning the reliability of this type of procedure. One question is: How do we know that the refinement procedure converges on a true orbit of the system, using finite precision FP arithmetic? This issue is addressed by the procedure above for two-dimensional maps (see also Hammel *et al.* , 1987). For higher-dimensional systems this issue was addressed by Sauer & Yorke (1991), who showed that under the conditions of a theorem they prove, *viz.*, if certain quantities evaluated at the points of the orbit are sufficiently small, then the iterated application of the refinement procedure results in a sequence of pseudo-orbits whose limit is an exact orbit that is not very far from the original pseudo-orbit. This result does not prove that the refinement procedure always works, however, since it requires establishing the conditions of the theorem for each refinement calculation to prove that an exact shadow exists. Quinlan & Tremaine (1992) showed that for simple systems glitches can occur that do not depend on any refinement algorithm. And Hayes & Jackson (2005) point out that even if refinement is successful, in the sense that it converges to machine precision, this only establishes the existence of a nearby pseudo-orbit with less noise, not that there is a nearby exact orbit. Hayes (1995) was also able to show examples where refinement failed to find a pseudo-orbit with less noise when one existed and where the iteration continues indefinitely without converging or blowing up. Thus, it is unknown whether convergence of the nonrigorous refinement procedure to machine precision implies the existence of an exact orbit of similar length to that of the pseudo-orbit, but this is usually taken as good evidence that such an exact orbit exists. For other limitations of the method, see (Quinlan & Tremaine, 1992).

As was mentioned in section 2.2, for the shadowing of ODE problems an extension of the above described procedure is required, *viz.*, is it necessary to show that shadowing of a pseudo-orbit by an exact orbit occurs in both space and time. The procedures for doing this mentioned in section 2.2 all work by relying on a shadowing theorem for ODE and then computing quantities along the pseudo-orbit that ensure that the conditions of the theorem are met.

As an example, we consider the method developed by Coomes *et al.* (1994b) for time rescaling. The statement of the theorem requires the definition of a variety of quantities. The norms in this context are taken to be the Euclidean norm for vectors and the induced operator norms for matrices and linear operators. Let y_n, $0 \leq n \leq N$, be a δ-pseudo-orbit of a system (1.1) with flow φ_t, with an associated sequence of times h_n, $0 \leq n \leq N$. The relevant definition of ε-shadow for ODE systems, then, is a true orbit x_n, $0 \leq n \leq N$, of φ_t for times $t = t_n$, *viz.*, $\varphi_{t_n}(x_n) = x_{n+1}$, such that

$$\|x_n - y_n\| \leq \varepsilon \quad \text{and} \quad \|t_n - h_n\| \leq \varepsilon.$$

In a manner similar to above, Y_n is now considered to be a sequence of matrices that differ from the Jacobian of φ_{t_n} evaluated at y_n by less than δ, *i.e.*,

$$\|Y_n - D\varphi_{h_n}(y_n)\| \leq \delta, \qquad 0 \leq n \leq N-1. \tag{3.11}$$

We also define a sequence A_n of $(n-1) \times (n-1)$ matrices in the following way. Let S_n, $0 \leq n \leq N$, be an $n \times (n-1)$ matrix which has as its columns an orthonormal basis for the $n-1$-dimensional subspace orthogonal to y_n. Then we let

$$A_n = S_{n+1}^* Y_n S_n, \qquad 0 \leq n \leq N-1,$$

which makes A_n the restriction of Y_n to the subspace orthogonal to $f(y_n)$ followed by a projection onto the subspace orthogonal to $f(y_{n+1})$. We then define a linear operator L: $(\mathbb{R}^{n-1})^{N+1} \to (\mathbb{R}^{n-1})^N$, defined such that if ξ_n is a sequence in $(\mathbb{R}^{n-1})^{N+1}$, then $L\xi \in (\mathbb{R}^{n-1})^N$ is defined by

$$(L\xi)_n = \xi_{n+1} - A_n \xi_n, \qquad 0 \leq n \leq N-1.$$

Since this operator is onto it has a right inverse. Let L^{-1} be a right inverse of L. Then, finally, we define seven constants. Let ε_0 be a positive number and let U be a convex open set containing the sequence $y_n, 0 \leq n \leq N$ such that if

x is in the ε_0-ball around y_n, then the solution $\varphi_t(x)$ is defined for $0 \le t \le 2h_n$ and is in U. Then, for such a U we define

$$M_0 = \sup_{x\in U} \|f(x)\|, \qquad M_1 = \sup_{x\in U} \|Df(x)\|, \qquad M_2 = \sup_{x\in U} \|D^2 f(x)\|,$$

$$\Delta = \inf_{0\le n\le N} \|f(y_n)\|, \qquad \Theta = \sup_{0\le n\le N-1} \|Y_n\|, \qquad h = \sup_{0\le n\le N-1} h_n.$$

With these definitions, we may now state the finite-time shadowing theorem of (Coomes *et al.* , 1994b): Let $y_n, 0 \le n \le N$ be a δ-pseudo-orbit of an autonomous system $\dot{x} = f(x)$, and let

$$C = \max\{\Delta^{-1}(\Theta\|L^{-1}\| + 1), \|L^{-1}\|\}. \tag{3.12}$$

If δ satisfies the inequalites

(i) $C(M_1+1)\delta \le \frac{1}{2}$,

(ii) $4C\delta < \min_{0\le n\le N-1} h_k, 4C\delta < \varepsilon_0$,

(iii) $8\left(M_0M_1 + 2M_1e^{2M_1h} + 2M_2he^{4M_1h}\right)C^2\delta \le 1$,

then the pseudo-orbit $y_n, 0 \le n \le N$, is ε-shadowed by a true orbit $x_n, 0 \le n \le N$, with

$$\varepsilon \le 4C\delta. \tag{3.13}$$

For a proof of this theorem, see (Coomes *et al.* , 1994b, 38-43).

With the statement of the theorem, we can now sketch how the numerical algorithm outlined by Coomes *et al.* (1994b, 38) works. Consider a standard one-step method applied to the autonomous version $\dot{x} = f(x),\ x(t_0) = y_0$, of

(1.1), which generates a solution sequence $y_n, 0 \leq n \leq N$, and corresponding time-steps h_n. The matrices Y_n are then computed at each step by applying the same numerical method for a time-step of h_n to the larger IVP

$$\dot{x} = f(x), \quad \dot{X} = Df(x)X, \qquad x(t_0) = y_n, \quad X(t_0) = I.$$

With an appropriate choice of a one-step method and control of the size of the time-steps h_n, we can control the local errors so as to produce a δ-pseudo-orbit y_n and matrices Y_n that satisfy condition (3.11) (Coomes *et al.* , 1994b, 38).

Then an appropriate U given the problem being considered is selected, which could be some absorbing set for the problem. Then the constants M_0, M_1 and M_2 can be determined. The details of how $\|L^{-1}\|$ is calculated, which involves computing the matrices S_n, are technical and are described in detail in (Coomes *et al.* , 1995), as is the entirety of the algorithm. With all of this in place, we can then calculate the quantities $h, \delta, \Delta, \Theta$, and $\|L^{-1}\|$, updating them at each step of the integration. Then at each step the conditions (i)-(iii) can be checked. If they are satisfied, then we may conclude that there is an exact orbit of the system that ε-shadows the pseudo-orbit up to that point in time. The integration is then continued until the conditions of the theorem cannot be satisfied.

For descriptions of the methods mentioned in section 2.2, *i.e.*, those developed by Chow & Palmer (1991), Chow & Palmer (1992), Chow & Van Vleck (1994), Sanz-Serna & Larsson (1993), Sauer & Yorke (1991), see the discussion in (Hayes & Jackson, 2005, 312-16), which also considers Coomes *et al.*

(1994b).

Notice that the quantity $\|L^{-1}\|$ in equation (3.13), where C is defined in equation (3.12), acts as a magnifying factor between the distance ε to the shadow and δ, which is a bound on the local error of the pseudo-orbit. Thus, we see that $\|L^{-1}\|$ acts like a condition number, where ε is analogous to the forward error and δ to the backward error. The operator L^{-1} is closely related to the operator $\mathcal{L}^{-1}$, which is a right inverse of the operator $\mathcal{L}$: $(\mathbb{R}^{n-1})^{N+1} \to (\mathbb{R}^{n-1})^N$ defined such that for a sequence ξ_n in $(\mathbb{R}^{n-1})^{N+1}$, $\mathcal{L}\xi \in (\mathbb{R}^{n-1})^N$ is defined by $(\mathcal{L}\xi)_n = \xi_{n+1} - D\varphi(y_n)\xi_n$, where φ: $\mathbb{R}^n \to \mathbb{R}^n$ is a C^2 function. Chow & Palmer (1992) prove a theorem that shows that $\|\mathcal{L}^{-1}\|\delta$ is approximately the shadow distance for n-dimensional pseudohyperbolic systems. For this reason $\|\mathcal{L}^{-1}\|$ is sometimes called the *condition number*. It has also been referred to as the *modulus of continuity* and *brittleness*. If the condition number is similar to the size of the inverse of the machine epsilon or larger then the shadowing distance becomes of the order of the size of the variables themselves, and accurate shadowing becomes impossible.

3.3 Method of Modified Equations

Although not much work has been done on the automation of the method of modified equations, Ahmed & Corless (1997) describe such an algorithm for explicit one-step methods and provide a Maple implementation of that algorithm in an appendix to that paper. The suggestion of automating the method goes back a bit further, as Corless (1994a) points out that Char *et al.* (1991) said that "there may be some scope for" automating the method of modified equations with a symbolic manipulation package such as Maple. It is seen that the procedure for deriving modified equations described in section 2.3 is algorithmic. So an algorithm that can be implemented using a computer algebra system can follow a similar approach.

The algorithm described by Ahmed & Corless (1997) is the following. Consider an explicit one-step numerical method $y_{n+1} = y_n + hI(y_n, h)$. Letting $y(t) = y_n$ and $y(t+h) = y_{n+1}$ then we first form the expression for the local truncation error[6]

$$e = \sum_{n=1}^{p} \frac{h^{n-1}}{n!} y^{(n)}(t) - I(y(t), h),$$

which is then rearranged to isolate y' and differentiated p times. After each differentiation one more term in the series is truncated so that expressions for $y^{(n)}$ are obtained for $1 \leq n \leq p$. This process ensures that $y^{(p)}$ only contains derivatives of lower orders.

For the purposes of induction, assume that we have expressions for $y^{(n)}, i+1 \leq n \leq p$, in terms of $y^{(i)}$ and below. We then substitute these expressions into the expression for $y^{(i)}$. This expression will contain $y^{(i)}$ on the right hand size but only multiplied by some multiple of h. We recursively substitute this expression into itself until an expression for $y^{(i)}$ containing terms of with lower derivatives. This equation is then used to eliminate $y^{(i)}$ from the higher derivatives, completing the induction step. After this process is repeated p times we obtain as our p-th order modified equation $\dot{y} = y^{(1)}$.

Although it is less general, Hairer & Vilmart (2006) describe automation

[6] Ahmed & Corless (1997, 4) point out that I could also be expanded in a series in h, but since I is independent of h for explicit methods this is often trivial.

of a procedure that uses the method of modified equations to generate higher-order versions of the symplectic and time-reversible discrete Moser-Veselov (DMV) algorithm for application to the free rigid body problem. The problem is preprocessed by generating a modified problem using the method of modified equations, to which the DMV method is applied. The result is a higher-order version of the DMV algorithm. An implementation of the DMV algorithm using quaternions is described and a Maple script for the computation of the quantities required for the preprocessing is provided.

Chapter 4

Connections Between Local Error and the Defect

The error quantity that is usually of central interest in the numerical solution of ODE is the global error $E(t)$, where

$$E(t_{n+1}) = y_{n+1} - y(t_{n+1}).$$

It is generally not possible to control the global error directly (see, *e.g.*, Shampine & Watts, 1976, 173). Instead the usual strategy is to try to control the LEPS $\epsilon(t)$, where

$$\epsilon(t_{n+1}) = y_{n+1} - z_n(t_{n+1}),$$

where we recall that $z_n(t)$ is the local exact solution to (1.1) with initial condition $y(t_n) = y_n$. The basic rationale for the use of local error control as an indirect control on the global error is the following bit of reasoning, due to Shampine & Watts (1976). The global error can be written as

$$E(t_{n+1}) = [y_{n+1} - z_n(t_{n+1})] + [z_n(t_{n+1}) - y(t_{n+1})].$$

The first term in brackets is just the local error and the second term is a quantity that depends on the stability of the differential equation, since its size depends on how much the integral curves of (1.1) starting at y_n and $y(t_n)$ spread apart by $t = t_{n+1}$. In particular, for small step-sizes the second term is approximately

$$[I + h_n J_n] \cdot E(t_n),$$

where $h_n = t_{n+1} - t_n$ and $J_n = J_f(t_n, y_n)$. Thus, the expression for the global error breaks up into a term depending on the numerical method used and a term independent of the numerical method, depending only on the stability of the equation itself, since the eigenvalues of the Jacobian determine the (linear) stability properties of the equation. Attempting to control the local error approximately controls the global error if the equation is not dynamically unstable. For such cases, if the local error begins to grow for a given step-size, a method is able to detect the developing numerical instability and compensate by reducing the step-size in order to keep the method stable. For equations with positive Lyapunov exponents, however, local error control loses its efficacy since it is no longer possible to approximately control the global error by reducing the step size. Defect control also fails to control the global error in such cases, yet it can still be useful for reasons that are discussed in section 5.1.

An alternative to control of the local error is to attempt to control the defect, as has been discussed in sections 2.1 and 3.1. In the case of the defect, there is a rigorous result connecting the defect and the global error, *viz.*, the following theorem:

Theorem 4.1 (Alekseev-Gröbner Theorem) Let $y(t)$ be the exact solution to the ODE (1.1) with initial condition $y(t_0) = y_0$ and let $u(t)$ be the solution to the modified equation

$$\dot{u}(t) = f(u,t) + \delta(u,t), \qquad u(t_0) = y_0,$$

with defect $\delta(u,t)$.[1] Then, if $\varphi_{t,t_0,f}$ is the flow of the vector field f of the ODE and $\frac{\partial \varphi_{t,t_0,f}}{\partial y_0}$ is continuous, then

$$E(t) = y(t) - u(t) = \int_{t_0}^{t} \frac{\partial \varphi_{t,\tau,f}(u(\tau))}{\partial y_0} \delta(u(\tau),\tau) d\tau. \tag{4.1}$$

If the flow is not too sensitive to changes in the initial condition at the various points along the solution $u(t)$, then the partial derivative of the flow can be approximated by the Jacobian $J_f(u(t))$, which can be calculated given the solution $u(t)$. The resulting expression for $y(t)$ is the solution of the first variational equation. In this case, since we know $\delta(t)$, we can compute an estimate of the global error. This also provides a rationale for defect control, since there is a rigorous result connecting the defect to the global error in a way that is not particularly sensitive to the problem. As was the case for the local error, the expression for the global error breaks up into two factors: $\delta(t)$, which depends on the numerical method (and interpolant); and $G(t,\tau) = \frac{\partial \varphi_{t,\tau,f}(u(\tau))}{\partial y_0}$, which depends only on the dynamical stability (conditioning) of the problem itself.

[1] Note that unlike in section 2.1, where the defect was not a function of u, the Alekseev-Gröbner theorem applies to the general case where the defect can depend on u.

Defect control has an important feature that makes it more appealing than local error control: since the defect can be viewed as a perturbation of the initial equation, one is easily able to interpret what the control code is doing, *viz.*, it is ensuring that the modified problem actually being solved by the code is ε-*close* to the specified problem, where ε is the specified tolerance.

Definition 4.2 (ε-Nearby Problems) Given the problem (1.1), $\varepsilon > 0$, and a vector norm $\|\cdot\|$, a modified problem is ε-*close*, or ε-*nearby*, to (1.1) if there is a function $v(u,t)$ such that

$$\dot{u}(t) = f(u,t) + \varepsilon v(u,t),$$

where $\|v(u,t)\| \leq 1$, *i.e.*, if the defect $\delta(t)$ of the modified problem can be expressed as $\delta(u,t) = \varepsilon v(u,t), \|v(u,t)\| \leq 1$.

Then, when the problem is relatively stable, *i.e.*, well-conditioned, one is able to infer a small global error from a small defect. It remains the case, however, that most variable step-size codes used for solving IVP in ODE control the local error and not the defect. From a backward error point of view, we would like to understand the success of local error control codes in terms of local error control providing an indirect control of the defect. This motivates an attempt to better understand the connection between the local error and the defect.

There are a number of results connecting the local error and the defect in the literature. Stewart (1970) shows that the defect and the LEPUS are in a certain sense equivalent. This is accomplished by showing that, under reasonable assumptions on the structure of the error bound and assuming maximum and minimum step-sizes $h_{\max}$ and $h_{\min}$, the set of solution sequences produced by LEPUS control codes with a given ∞-norm error bound on the LEPUS contains the set of solution sequences produced by defect control codes within a slightly smaller ∞-norm error bound on the defect, and *vice versa.* Let L be a Lipschitz constant for the vector field f over the entire range of integration $[t_0, T]$. Then the results establish that any solution sequence with the LEPUS controlled to within ε can be interpreted as defect controlled to within $\varepsilon(1 + Lh_{\max})$. And any solution sequence with the defect controlled to within ε can be interpreted as LEPUS controlled to within $\varepsilon e^{Lh^+_{\min}}$, where $h^+_{\min} = T/N$ with N the number of steps. Thus, Stewart establishes that there is a close connection between the LEPUS and the defect, and that for problems for which L is not too large, which includes relatively stable non-stiff problems, a LEPUS controlled code can be interpreted to be a defect controlled code with only a slightly larger tolerance.

Note also that the connection Stewart makes is between the LEPUS and the defect, not the LEPS and the defect. In a preliminary consideration of how to develop a theory of variable step-size, variable method ODE solvers, Stetter (1976) considers conditions under which ODE solvers achieve tolerance proportionality, *i.e.*, a relationship between the tolerance ε and global error $E(t)$ of the form

$$E(t) = w(t)\varepsilon + o(\varepsilon), \tag{4.2}$$

where $w(t)$ and $w'(t)$ are bounded on $[t_0, T]$ and $o(\varepsilon)$ is understood here to mean a term numerically negligible compared to ε. He proves a theorem that shows that in order for ODE solvers to achieve tolerance proportionality, they must control the LEPUS and not the LEPS. This theorem shows that equation (4.2) being satisfied for all $t \in [t_0, T]$ is equivalent to the condition that the local errors $\epsilon(t_{n+1}, \varepsilon)$ generated for a tolerance parameter ε are of the form[2]

$$\epsilon(t_n, \varepsilon) = \overline{v}(t_{n+1}, t_n) h_n \varepsilon + o(h_n \varepsilon), \tag{4.3}$$

where $\overline{v}(t, \tau)$ behaves like an integral mean over $[\tau, t]$ of a function independent of ε and bounded on $[t_0, T]$. He proves this by first showing that the tolerance proportionality condition (4.2) is equivalent to having an interpolant $u(t)$ that satisfies (1.3) with a defect

$$\delta(t) = v(t)\varepsilon + o(\varepsilon),$$

where $v(t)$ is independent of ε and bounded for $[t_0, T]$. He then shows that this condition is equivalent to the conditions of the theorem. One direction of the required equivalence then follows from the fact that the Alekseev-Gröbner theorem implies that the local errors are of the form (4.3) by taking

$$\overline{v}(t_{n+1}, t_n) = \frac{1}{h_n} \int_{t_n}^{t_{n+1}} G(t_{n+1}, \tau) v(t) d\tau.$$

The converse follows by interpolating the numerical solution using the interpolant

$$u(t) = z_n(t) + \left(\frac{t - t_n}{h_n} \right) \epsilon(t_{n+1}, \varepsilon), \qquad t \in [t_n, t_{n+1}] \tag{4.4}$$

[2] The $o(h_n \varepsilon)$ term in this equation appears as $o(\varepsilon)$ in (Stetter, 1976, 192), but Higham (1991a, 461) point out the correction.

and using the assumption that the maximum step-size is $o(1)$ as ε becomes small. Note that this interpolant is C^0 but not C^1. Thus, Stetter showed that, asymptotically (as $\varepsilon \to 0$), achieving tolerance proportionality is equivalent to solving a nearby system of ODE, provided that the defect depends linearly on ε in the asymptotic limit. He also shows that, asymptotically, control of the LEPUS does provide indirect control of the defect, and hence can achieve tolerance proportionality of the global error. Stetter also shows that by interpolating between grid points with the interpolant $u(t) + o(\varepsilon)$, equation (4.2) is satisfied for all $t \in [t_0, T]$, *i.e.*, this interpolant is sufficient for tolerance proportionality, and hence the relationship between the LEPUS and the defect.

Higham (1991a) reformulates and clarifies the details of this result. In the reformulation, he is able to point more clearly to limitations of Stetter's theorem. An important limitation is that the interpolant of the first derivative $w'(t)$ of the function $w(t)$ in equation (4.2) must be continuous. He also emphasizes that because the theorem is an asymptotic result, it only applies when ε is small, where what counts as small will depend on the problem at hand. Thus, the result requires experimentation or elaboration in order to determine if tolerance proportionality can be achieved by the ODE solvers used in practice. Consideration of the details is beyond the scope of the present work, but Higham (1991a) includes numerical experiments in this paper, and also proves two corollaries, hinted at in Stetter (1981), that relate to practical error control methods. The second corollary provides conditions under which the solution sequences y_n produced by RK methods using certain methods of error control (specifically LEPUS control, LEPS control with local extrapolation, and defect control with an interpolant of higher order than the method) when interpolated using the interpolant in (4.4) achieve tolerance proportionality and the above mentioned relationship between the LEPUS and the defect. Higham & Stuart (1998) apply these results of (Higham, 1991a) to the long-time behaviour of systems where the vector field f has a special structure, *viz.*, special kinds of dissipative, contractive and gradient systems.

Following an approach to using the method of modified equations suggested by Babuska, Griffiths (1988) shows a manner in which controlling the LEPS or the LEPUS are understood to be equivalent to minimizing a defect in the L_1 and L_∞ norms, respectively. He consideres a predictor-corrector pair of methods formed from forward and backward Euler. He supposes that the time grid generated by the method is characterized by a continuously differentiable, monotonically increasing function F: $[t_0, T] \rightarrow [0, 1]$ defined by $t_n \mapsto n/N$. It follows that $h_n \approx 1/(N\dot{F}(t_n))$. He then selects a particular interpolant $u(t)$ generated from the local exact solution $z_n(t)$, so that $u(t)$ has the property that the defect $\delta(t)$ is equal to the LEPUS to leading order in h_n. This then enables him to consider the result of minimizing $\|\delta(t)\|$ by varying the function F for different function norms $\|\cdot\|$ on $[t_0, T]$. He shows that minimizing $\|\delta(t)\|_1$ with respect to F is equivalent to keeping the LEPS constant. If we let this constant be ε, then if ε is held constant, it follows that, asymptotically, $u(t)$ satisfies the modified equation (1.3) such that $\|\delta(t)\|_1 = O(\sqrt{\varepsilon})$. He also shows that minimizing $\|\delta(t)\|_\infty$ is equivalent to keeping the LEPUS constant, and that, by keeping the LEPUS at constant value ε, $u(t)$ satisfies (1.3) with

$\|\delta(t)\|_\infty = O(\varepsilon)$. Griffiths points out that, although this result was proved for a particular numerical method, the approach can be extended to higher order methods and also shows how other criteria for the control of the local error would behave.

These results serve to establish that controlling the LEPUS does control a defect provided that the bound on the LEPUS is sufficiently small, where what counts as small will depend on the particular problem. One limitation of these results in practice, however, is that they all rely in one way or another on the numerical solution being interpolated with the 'ideal' interpolant (4.4), or something similar depending on the local exact solution $z_n(t)$ and the LEPS. Because we can only estimate the LEPS and we do not have access to the local exact solution, such interpolants can hardly be considered computable in practice. Thus, we see that control of the LEPUS does control the defect in the asymptotic regime where $\varepsilon \to 0$, but this assumes that the defect is computed using the ideal interpolant. Thus, provided that the tolerance on the LEPUS is sufficiently tight, where what counts as tight will depend on the problem at hand, the control of the LEPUS can be understood to be controlling a defect, and so the numerics are tracking the exact solution to a close to ε-nearby problem. In order to make the stronger claim that, under the appropriate asymptotic conditions, the numerics actually find the exact solution to a nearly ε-nearby problem, the defect that the control of the local error controls must be computable. The current results on this issue all rely on some sort of ideal interpolant.

Determining the conditions under which control of the local error controls the defect for arbirary interpolants, or even the interpolants used in practice, is beyond the scope of the present work. We may, however, examine certain aspects of the relationship between the LEPUS and the defect for arbitrary interpolants by considering the general relationship between the local error and the defect.

The general relationship between the local error and the defect is given by an application of the Alekseev-Gröbner theorem. Applying the result to the local problem (1.6), we have that

$$\epsilon(t) = z_n(t) - u(t) = \int_{t_n}^{t} G(t,\tau)\delta(u(\tau),\tau)d\tau, \qquad t \in [t_n, t_{n+1}].$$

Dividing both sides by $h_n = t_{n+1} - t_n$ and setting $t = t_{n+1}$, we see that

$$e_{n+1} = e(t_{n+1}) = \frac{z_n(t_{n+1}) - u(t_{n+1})}{h_n} = \frac{1}{t_{n+1} - t_n}\int_{t_n}^{t_{n+1}} G(t,\tau)\delta(u(\tau),\tau)d\tau. \tag{4.5}$$

Thus, in general, we see that the LEPUS $e(t_{n+1})$ is equal to the time average of the product of the defect and the function $G(t,\tau)$, which depends on the conditioning of the problem, over the length of the step.

Consider an IVP (1.1). For sufficiently small time-steps, depending on the severity of the nonlinearity, or simply asymptotically as $h \to 0$, the local behaviour of this system near the point (y_n, t_n) is well described by the linearized system

$$\dot{y} = J_n y, \qquad y(t_0) = y_0.$$

where $J_n = J_f(y_n, t_n)$. In this case, $G(t,\tau) = J_n$ and is constant, so it comes out of the integral in (4.5) and, treating the defect as an non-autonomous perturbation of the system, we have that

$$e(t_{n+1}) = J_n \langle \delta(t) \rangle, \tag{4.6}$$

where $\langle \delta(t) \rangle$ is the time-average of $\delta(t)$ over $[t_n, t_{n+1}]$. This result is interesting from the point of view of stiff equations. Although there is no agreed upon definition of stiffness, since it depends on the problem and the method, a useful heuristic, that covers certain aspects of stiffness, is to think in terms of large eigenvalues.[3] If J_n has large eigenvalues, then it amplifies vectors in certain directions significantly. We notice from equation (4.6) that even if the average of the defect over the step has a small component in one any of these directions, then, even if the defect is small, the local error could be very large. The same is true of the global error in light of equation (4.1) and the fact that a linear approximation to the derivative of the flow is the Jacobian of the vector field. Thus, defect control would not work to control the global error.

This issue is actually better analyzed in terms of singular values. Let $J_n = U \Sigma V^*$ be the singular value decomposition of J_n, so that

$$J_n v_i = \sigma_i u_i,$$

where v_i and u_i are the column vectors of U and V and σ_i are the corre-

[3] Considering the idea of stiffness as being associated with the problem having two vastly different time scales, a better heuristic is a large *range* of eigenvalues. In order to explain why BEA fails on this kind of problem, however, it is the additional condition that the largest eigenvalue is large that matters.

sponding singular values. If $\delta(t)$ has a small component in the direction v_1, corresponding to the largest singular value σ_1, then even though the defect may be small, the local and global error could be quite large, depending on the value of σ_1. Thus, when the square roots of the eigenvalues of $J_n^* J_n$ are very large, regarding the defect as a perturbation of the original ODE, then the problem is stiff in a technical sense and we can expect backward error analysis to fail on such a problem.

Considering the asymptotic regime ($h \to 0$) where we can equate $G(t,\tau) = J_n$, then equation (4.5) gives us other nice relations. For the remainder of this chapter, all the norms are vector p-norms unless explicitly stated otherwise. In this regime we have from (4.5) that

$$e(t_{n+1}) = \frac{J_n}{h_n} \int_{t_n}^{t_{n+1}} \delta(u(\tau),\tau)d\tau. \tag{4.7}$$

Taking the p-norm of both sides we find that

$$\begin{aligned} \|e(t_{n+1})\| &\leq \frac{\|J_n\|}{h_n} \left\| \int_{t_n}^{t_{n+1}} \delta(u(\tau),\tau)d\tau \right\|, \\ &\leq \frac{\|J_n\|}{h_n} \int_{t_n}^{t_{n+1}} \|\delta(u(\tau),\tau)\| d\tau, \\ &\leq \|J_n\| \max_{t_n \leq \tau \leq t_{n+1}} \|\delta(u(\tau),\tau)\|. \end{aligned}$$

Therefore,

$$\|e(t_{n+1})\|_p \leq \|J_n\|_p \max_{t_n \leq \tau \leq t_{n+1}} \|\delta(u(\tau),\tau)\|_p. \tag{4.8}$$

Alternatively, using Hölder's inequality with p and q conjugates, following the same line of reasoning as before, we have from (4.7) that

$$\|e(t_{n+1})\|_1 \leq \|J_n\|_p \max_{t_n \leq \tau \leq t_{n+1}} \|\delta(u(\tau), \tau)\|_q. \tag{4.9}$$

As a special case we have

$$\|e(t_{n+1})\|_1 \leq \|J_n\|_2 \|\delta(u(t), t)\|_\infty, \tag{4.10}$$

where the ∞-norm here is now the function norm on the the interval $[t_n, t_{n+1}]$.

The relations (4.8), (4.9), and (4.10) establish a relationship similar to the previous results between the sizes of LEPUS and the defect in the asymptotic limit $h_n \rightarrow 0$, and they hold for any interpolant $u(t)$ of the numerical solution used to compute the defect $\delta(t)$. They show that, asymptotically, control of the defect indirectly controls the local error, provided that the norm of the Jacobian is not too large. So for relatively stable non-stiff problems, for small step-sizes or tight tolerances, the sizes of which will depend on the problem, the local error can only be slightly larger than the defect. Although this does not further the main desideratum of this chapter, since this is to clarify the conditions under which control of the local error indirectly controls the defect, these results do clarify the asymptotic relationship between the local error and the defect for arbitrary interpolants.

Chapter 5

Advantages of Backward Error Analysis

5.1 Backward Error Analysis on Chaotic Problems

It is usually thought that the main advantage of backward error analysis is that when it is applied to a well-conditioned problem a small backward error implies a small forward error (global error in the present context). Since chaotic problems are ill-conditioned by definition, it would then be expected that backward error analysis would be of little use on such problems. But this all depends on what one is looking for in a numerical solution.

Because of the sensitivity to initial conditions exhibited by chaotic problems, small roundoff errors in the computation of values of the solution become amplified and the global error can become large very quickly. Since this is the case for any numerical method, the presence of truncation and roundoff error makes the accurate computation of trajectories of chaotic systems over long time periods impossible. An important point about models of chaotic physical systems that is not often emphasized, however, is that small physical perturbations not taken into account by the model have the same effect. Thus, when physical perturbations are present, the phase trajectory of the actual system could quickly diverge from the phase trajectory of the specified model. This basic point shows that the result of the presence of physical perturbations in the system of interest is that the global error is often not a useful quantity to consider when it comes to chaotic problems, and in such cases it would not be even if we were able to solve models in exact arithmetic. Thus, for chaotic problems, we generally must look for different kinds of information from a numerical solution.

Since the main interest in the control of local error is as a means of controlling the global error, it would appear that for chaotic problems a major reason for the use of local error control is undermined. This is not so for defect control, however, as Corless (1992b, 1994a) has argued. Control of the defect to keep it small ensures that the numerical solution that one obtains is an exact solution to a nearby problem, even on chaotic problems. The question raised here, then, is: How useful are exact solutions to nearby problems when the problem is so ill-conditioned that even tiny numerical errors grow exponentially fast? Addressing this question requires that one rethink what one should expect from a numerical solution to an ill-conditioned ODE problem. This becomes quite a natural thing to do if one adopts a backward error point of view.

Corless (1992b) makes the point that if the answer to a question depends sensitively on the input then one is asking the wrong question. Taking this point, the result of the fact that physical and modeling error are always present is that by trying to (indirectly) control the global error for a numerical solution of a chaotic problem one is trying to answer the wrong kind of question. To draw useful conclusions from the exact solution a nearby problem one requires that some quantity of physical interest is stable under some relevant class of perturbations. Corless refers to such a quantity as a 'statistic,' where the term applies both to deterministic and stochastic systems (see Corless, 1992b, 324, for details on this).

Definition 5.1 (Statistic) A *statistic*, in the present context, is some function s that can depend on various parameters of a problem. For example, we could have that $s(u(t), y_0, \varepsilon, v(t))$, so that the statistic depends on variables such as the exact solution $u(t)$ to the perturbed problem, the initial condition y_0, the tolerance ε and the defect $v(t)$.

The well-behavedness condition that is required for a statistic is something akin to stability of solutions under perturbation of the initial condition, or well-posedness, *i.e.*, that there exists a unique solution to the problem that depends continuously on the data. If there were no such well-behaved quantity then no useful conclusions could be drawn from the numerical solution since the presence of physical and modeling error will cause the model to give completely different values than those possessed by the system being modeled. Corless calls any system that has a well-behaved statistic a *well-enough conditioned problem.*[1]

Definition 5.2 (Well-Enough Conditioning) A *well-enough conditioned* problem

[1] As an example of a problem that is not well-enough conditioned, Corless (1992b, 324)

is one that has a statistic that is stable under a (continuous) class of perturbations that is relevant to the problem (1.1) being considered.

It is worth pointing out that if this statistic is chosen to be the global error, then the standard definitions of well-conditioning for ODE problems are recovered.

A good example of what a useful statistic would be in the context of chaotic problems is the largest Lyapunov exponent. Establishing that this is a statistic is a difficult problem, however, since it requires proving a stability/continuity result for the Lyapunov exponents of the system as the problem is varied. Based on a computed solution one can estimate the Lyapunov exponents of the modified problem exactly solved by the numerical method (as in, *e.g.*, Dieci & Vleck, 2005), but without a continuity result for the Lyapunov exponent this does not imply that the computed Lyapunov exponents are close to those of the original problem. Nevertheless, Corless & Pilyugin (1995) showed that if the original and modified problems are both generic, then provided that the shadow distance ε is sufficiently small, solutions of the modified problem (1.3) are traced by solutions of the original problem (1.1). This is suggestive that for generic systems, the largest Lyapunov exponent is a stable statistic, but it requires that we know that the original system is generic. If the original

considers the problem

$$\dot{y}(t) = |1 - y^2|, \qquad y(0) = 0,$$

which has exact solution $y(t) = \tanh t$. Now, the modified problem $\dot{y} = |1 - y^2| + \varepsilon$ is qualitatively different, since for $\varepsilon > 0$ the solution blows up in finite time, where the time of the singularity is proportional to $\ln \varepsilon$. The fact that the location of the singularity is not fixed, and that ε must be exponentially small in order to have the singularity occur at a finite range of times of interest, makes it unlikely that any meaningful statistic would be well-behaved (Corless, 1992b, 324).

problem was not generic, even if chaotic behaviour was generic in the neighbourhood surrounding the original problem, the original problem could fail to be chaotic. This situation, however, is also addressed by a backward error point of view for reasons we now consider.

Many models, particularly ones of complicated phenomena, are subject to a significant amount of modeling error, so that the specified model is often subject to significant idealization. Recall from chapter 1 that even though we usually focus on the specified problem (1.1), the presence of modeling error $\mu(t)$ means that this problem is actually

$$\dot{y}(t) = f(y,t) = g(y,t) + \mu(t), \qquad y(t_0) = y_0.$$

Thus the system we are modeling is actually

$$\dot{x}(t) = g(x,t), \qquad x(t_0) = y_0. \tag{5.1}$$

For example, models of galactic dynamics may not take into account to influence of exotic forms of dark matter that we do not know about, and it will not take into account the motion of every star.[2] Thus, the specified problem is actually a perturbation of the correct model of the system of interest. And even if it were the case that $\mu(t) = 0$, there will always be some degree of physical error $\pi(t)$ present. This non-autonomous perturbation could be

[2] It is to be recognized that, properly speaking, in general the system 5.1 will be a different dimension than the original system 1.1. The manner of describing the modeling situation here is not intended to be a rigorous representation of how modeling with ODE works, but rather as sufficiently precise tool for the purpose of clarification.

something like a violent storm on Jupiter that causes small perturbations to its orbit and hence perturbs the motion of the bodies in the solar system ever so slightly. Or it could be something more prosaic like the vibrations from a nearby freeway affecting a laboratory experiment. In any case, the presence of perturbations means that the actual physical situation is described by

$$\dot{x}(t) = g(x, t) + \pi(t), \qquad x(t_0) = y_0.$$

Also, since perturbations can be quite different in different contexts, it is really an entire range of problems of this form that are to be understood as describing the system of interest.

From this point of view, then, every ODE model is a modified version of a correct model of the system of interest, even though the modification may be extremely small. This shows that the specified model is usually not so specially situated that conclusions about the actual physical system being modeled have to be drawn from it. We also see from this that in every consideration of problems (1.1) we must be mindful of how small perturbations affect the problem. The basic insight here for chaotic problems, then, which Corless (1992b) attributes to Enright, is that the result of the omnipresence of physical and modeling error is that a problem is chaotic in a practical sense if nearby problems have positive Lyapunov exponents. Thus, the use of BEA enables us to conclude that the presence of chaotic behaviour in the numerical solution of models of real world systems provides evidence that the system being modeled exhibits chaotic behaviour. In cases where modeling error is significant, so that it is not clear that the model describes the behaviour of a real system, even if the system as posed does not have positive Lyapunov exponents, this practical definition still applies. For example, there is a significant degree of idealization involved in the derivation of the Lorenz equations from the equations of fluid dynamics, so it is not clear that the Lorenz equations describe the behaviour of any real system. And though we do not have a proof that the Lorentz equations are chaotic, the presence of chaos in simulations of the Lorenz equations nevertheless shows that the Lorenz system is chaotic in a practical sense (Corless, 1992b, 332-33).

Turning to think of the effect of numerical error, recall that the numerical error $\nu(t) = \phi(t) + \tau(t)$ introduced in the numerical solution of ODE can be viewed as another form of perturbation of the model (5.1). Moreover, it is often the case that the numerical error can be made smaller in norm than the largest sources of physical error. We usually have that $\|\phi(t)\| \approx 10^{-15}\|f\|$, provided that the subroutines we use to evaluate functions are numerically stable, so that we will usually have $\|\phi(t)\| \ll \|\pi(t)\|$ for the largest sources of physical error. And with high-order numerical methods and tight tolerances we will usually have that $\|\tau(t)\| \ll \|\pi(t)\|$ for the largest sources of physical error. Thus, in cases of careful modeling and computation, the physical error will dominate the other sources of error so that the numerics are modifying the problem much less than the physical world is. The insight to be gained from this, then, is that if the numerical perturbations are small compared to reasonable physical perturbations and if the system is well-enough conditioned, *i.e.*, the quantities of interest are sufficiently stable under perturbation, then one can get just as much insight from the problem exactly solved by the numerics as one would obtain from the exact solution to the originally specified problem.

As a result of this, the presence of chaos in numerical simulations shows that there are chaotic problems in both the neighbourhood of the specified problem and the range of problems corresponding to the system of interest. This shows how Enright's practical notion of chaos can be inferred from the presence of chaos in numerical simulations. From a technical point of view, this can be see to imply that if chaotic behaviour is generic in a neighbourhood of the specified problem, then the system is chaotic in this practical sense, since, in this case, there are chaotic problems arbitrarily close to the specified problem. At least this is so unless the original problem is non-generic and somehow all physical perturbations push this problem onto other non-generic problems. In this case it really is the behaviour of non-generic problems that matters. However, that physical perturbations can be both continuous and stochastic makes this possibility seem unlikely. Turning things around, it is also possible that the numerical perturbations push the original problem onto non-generic problems. In this case the behaviour observed from the results of numerical simulations could be non-generic, so that chaos in the simulation may not imply chaos in the system. Although it is difficult to determine how likely this possibility is, it does not appear to be an unreasonable possibility since all numerical perturbations are the result of discretization of continuous problems, which make them a quite special kind of perturbation (*cf.* Corless, 1992a).

It is worth noting that though we have been focusing on the issue of genericity in the consideration of chaotic problems, similar considerations will apply when the statistic of interest is something other than the largest Lyapunov exponent. In such a case, Enright's notion of 'chaotic in a practical sense' would translate into 'well-enough conditioned in a practical sense,' *viz.*, a problem would be well-enough conditioned in a practical sense with respect to some statistic s if s is sufficiently stable in a neighbourhood of the specified problem. In this way the behaviour of s in numerical solutions can be understood to give us evidence for how s behaves in the system being modeled.

Even though a small defect means that one has a high quality solution even in the case of well-enough conditioned chaotic problems, there are other issues that we must consider if we really are to gain as much insight from the exact solution to the modified problem as we would get from the exact solution of the specified one. Aside from the smallness of the defect and well-enough conditioning, we must also consider whether the perturbation introduced by the defect is physically reasonable and what effect typical physical perturbations will have on the problem. These are issues that must be addressed not only for all kinds of BEA, but also any kind of error analysis. It is important for a full error analysis to examine how physically reasonable perturbations affect the model in question. Corless (1992b, 325) mentions tools for dealing with this question, such as the first variational equation and perturbation theory. This raises issues for qualitative analysis methods of systems of ODE. If one is using such methods to gain insight into the dynamics, then one must be careful that the invariant manifolds of the specified system are stable both under physical perturbations and the perturbations introduced by the numerics. Although this is a delicate issue (Corless, 1992b, 325), it does not raise any problems specifically for BEA since this is a serious concern for any kind of error analysis. Adequately addressing the issue of whether the perturbations due to numerical error can reasonably regarded as physical is required to ensure that the perturbation from the defect really can be treated on an equal footing with physical perturbations. Since this issue applies to all kinds of BEA and is not specific to chaotic problems, we will pospone further consideration of this until the following section.

An issue that is worth discussing here is what conclusions can be drawn from the structure of the defect. The structure of the defect can actually be a source of insight. Corless (1994a, 18-20) showed that the method of modified equations can be used to explain the structure of the defect of the solution to the Lorenz equations produced by the fixed time-step forward Euler method. For chaotic values of the parameters in the Lorenz equations and following the solution on the attractor, it was found that the defect has a similar structure to that of the Lorenz mask. This shows that there are correlations between the defect and the solution. Corless explained the correlation using the method of modified equations to show that the first term in the modified equation accounts for over 99.5% of the size of the defect.

We perform a similar kind of analysis here on the Rössler system

$$\begin{aligned}\dot{x} &= -y - z,\\ \dot{y} &= x + ay,\\ \dot{z} &= b + z(x - c),\end{aligned}$$

solved using the defect-controlled Euler method `ode1d`. We let the parameters take the chaotic values $a = 0.4$, $b = 2$ and $c = 4$. Solving this using `ode1d` with the tolerance set to $\varepsilon = 10^{-1}$ we obtain the solution plotted in figure 5.1. We can interpolate the solution using the MATLAB method `pchi` on each component of the solution, thereby obtaining a vector of piecewise cubic Hermite interpolants that interpolates the entire solution. Since `pchi` also returns the derivative of the interpolants, this enables us to compute the defect. The result is plotted in figure 5.2(a). We see that the defect has a lot of structure and, as was found by Corless for the fixed time-step Euler method, it mimics the behaviour of the solution. To make the structure more clear, using the same defect, an estimate of the maximum value of the defect over each integration interval is calculated. Using a line plot, as in figure 5.2(b), it becomes clear that the structure in the defect is indeed mimicing the structure of the solution in 5.1. Slightly extending the method of Corless (1994a), this can be partially accounted for using the method of modified equations.

Recall from section 2.3 that using the method of modified equations, the numerical solution to a system $\dot{y} = f(y,t)$ using a fixed time-step Euler method

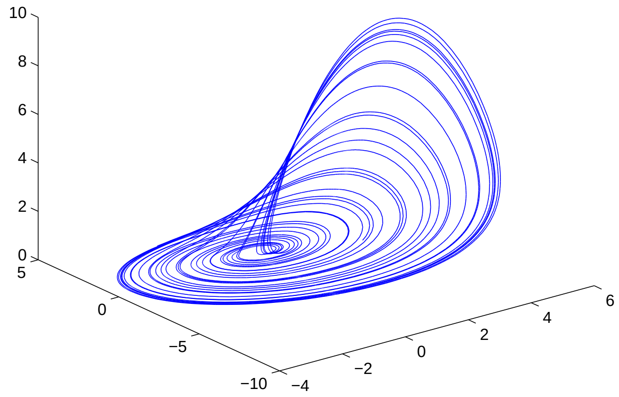


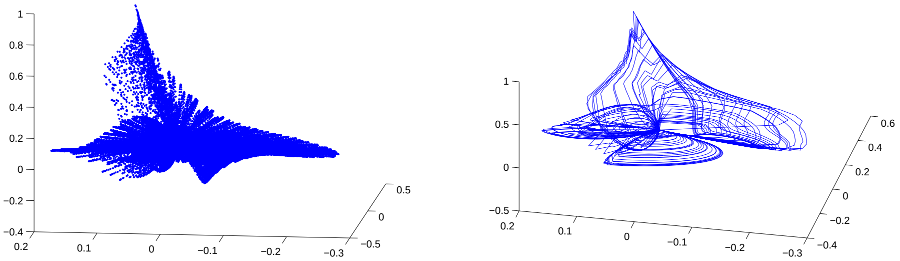
Figure 5.1: The Rössler system solved using `ode1d`.

is a second-order solution to

$$\dot{y} = \left(I - \tfrac{h}{2} J_f(y)\right) f(y), \tag{5.2}$$

This analysis relies on the method being used having a fixed time-step. To apply this method for a variable step method we must rescale the time so that the times are equally spaced. Let this time rescaling be given as $t = F(\theta)$, and for convenience choose the fixed step-size to be 1. Using this rescaling, we transform the original system to the system

$$\begin{aligned} \frac{dx}{d\theta} &= (-y - z)F'(\theta), \\ \frac{dy}{d\theta} &= (x + ay)F'(\theta), \\ \frac{dz}{d\theta} &= [b + z(x - c)]F'(\theta). \end{aligned}$$



(a) The defect of the `ode1d` solution evaluated at 80 000 points along the integration interval.

(b) Estimate of the maximum defect over each integration step of the solution.

Figure 5.2: The defect of the `ode1d` solution to the Rössler system computed using `pchi`.

Letting the original Rössler system be given by $\dot{y} = f(y)$, we may then conclude that our `ode1d` solution is a second-order solution to

$$\dot{y} = \left(I - \tfrac{1}{2}F'(\theta)J_f(y)\right) f(y), \tag{5.3}$$

since $h = 1$. Thus, letting $\delta(t)$ be the original defect, shown in figure 5.2, the term $\delta_1(t) = -\frac{1}{2}F'(\theta(t))J_f(y)f(y)$ provides a first order approximation to $\delta(t)$. To compute the Jacobian of this system we must be able to compute the derivative of the function $F(\theta)$. Since we only know the value of F at the grid points, we use forward differences to estimate the derivative at the beginning of each time-step. Based on this, the values of $F'(\theta)$ between grid points can be evaluated using the built-in MATLAB function `pchip`.

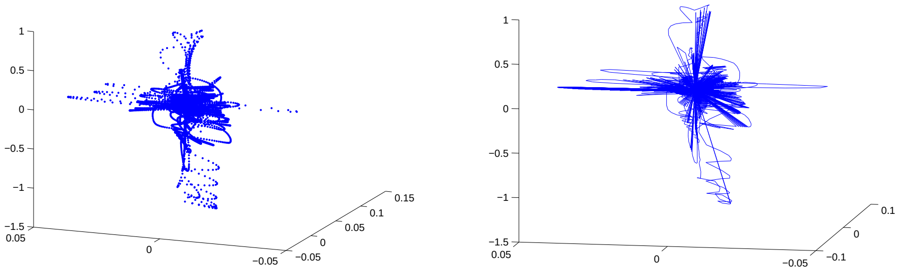
Now, following the method of (Corless, 1994a) we can interpolate the `ode1d` solution using `pchi` by matching the derivatives from (5.3), and not those of the equation $\dot{y} = f(y)$. We can then compute the defect $\delta_2(t)$ by substituting this interpolant back into (5.2). The result of this is that we have the exact solution to

$$\dot{y} = \big(I - \tfrac{1}{2}F'(\theta(t))J_f(y)\big)\, f(y) + \delta_2(t), \tag{5.4}$$

where $\delta_2(t) = \varepsilon^2 b(t)$ for some function $b(t)$. We wish to determine whether $\|b(t)\|_\infty \lessapprox 1$. The result of the computation of $\delta_2(t)$ is given in figure 5.3(a). Estimating the maximum defect on each integration interval as before, we obtain the plot in figure 5.3(b). We notice the interesting result that the maximum value of the z-component of the defect of the modified equation is about the same as for the defect of the original equation, *i.e.*, $|b(t)| \not\lessapprox 1$ for this component, but that the x and y components are of the order of the tolerance $\varepsilon = 10^{-1}$, so that $|b(t)| \lessapprox 1$ for these components. Thus, $\delta_1(t)$ is accounting for a large part of the defect $\delta(t)$ of the original equation. This partially explains the correlations found in figure 5.2. Looking carefully at figure 5.3 it appears that there is structure surrounding the origin similar to that seen in figure 5.2, but surrounded by noise. A more careful estimate of the maximum value of the defect over each time-step may reveal this structure more clearly. This also raises the question of whether the third-order modified equation can account for the large z-component of $\delta_2(t)$, which we will not address here.

5.2 Backward Error Analysis and Modeling

We have seen that the omnipresence of physical and modeling error means that the specified model problem is always subject to perturbations, so that a proper error analysis of any model must include an examination of the effects of perturbations on the model and its solutions. Because BEA allows numer-



(a) The defect $\delta_2(t)$ of the `ode1d` solution for the modified equation (5.2) evaluated at 80 000 points along the integration interval.

(b) Estimate of the maximum defect over each integration step of the solution.

Figure 5.3: The defect of the `ode1d` solution to the Rössler system computed using `pchi`.

ical error to be treated as a perturbation in the same way as physical and modeling error, there are strong reasons for regarding the solution provided by a numerical method as the exact solution to a nearby problem nearby that is *just as valid* as the solution to the original one, provided that the numerical defect is small. As was pointed out, this is so provided that the problem is well-enough conditioned *and* provided that the perturbation can reasonably be regarded as physical. We take up this last issue in this brief concluding section.

An important consideration in all forms of BEA for ODE is the justification of the comparison of numerical with physical and modeling error. First of all, the comparison depends on the norm chosen to measure the various errors and care must be taken to select the appropriate norm for the problem at hand. But more importantly, as has been pointed out, *e.g.*, by Hayes & Jackson (2005), it also matters how the numerical error affects the problem *qualita-*

tively. Numerical error, *i.e.*, discretization and roundoff error, may produce a perturbation in the problem that is not physically reasonable. Hamiltonian systems are a classic example, since roundoff, or numerical methods that do not preserve the symplectic structure of the flow, introduce spurious dissipation (or accumulation) of energy when computing phase trajectories for a conservative system. Other kinds of distortion are also possible, for example Hayes & Jackson (2005, 300-301) point out that the numerical error may also introduce nonphysical energy transport, even if the energy is conserved. Thus, it is an important matter, and one underemphasized in the literature, to consider whether the qualitative features of numerical error warrant a comparison to physical and modeling error.

The point to be made here is that many of the effects of roundoff and truncation error can be reasonably regarded as physical as long as the effects are small. Small amounts of dissipation do constitute a reasonable physical perturbation because of the system interacting with its environment, and small amounts of energy transport are reasonable due to either physical or modeling error. For example, if the vector field of the model is slightly different than the vector field of the system being modeled then energy could distribute itself differently in the actual system than what occurs in the specified model. Often when these effects are large it is because certain key structural features of the equations defining the model are not preserved by the numerical method being used. This issue is addressed by the development of specialized geometric methods for certain kinds of problems, which enable numerical solutions that better reflect the physical structure of the system being modeled. This can significantly reduce the distortions of numerical error, with the consequence that effects like spurious dissipation of energy are significantly reduced. Thus, the use of the method of modified equations, central in the development of geometric numerical methods, addresses the issue of whether numerical error can be regarded as physical. Therefore, the point of view of BEA, that valid numerical solutions are exact solutions to nearby problems that are just as valid as the solution to the specified problem, extends to the method of modified equations. It also extends to shadowing methods, since a perturbation of the initial conditions of a problem, provided that it is small, is amounts to the same effect as measurement error in the initial value of a dynamical system, which is always present to some degree. Thus, provided that one is using the kind of backward error analysis appropriate for the problem at hand, or some appropriate combination of them, BEA methods for ODE provide a strong set of tools for obtaining exact solutions to problems that are just as valid as the exact solution to the problem originally posed.

We can summarize the main point of this chapter in the following way. The omnipresence of physical and modeling error when mathematical models are used to describe, explain and predict real world phenomena, has the effect that the neighbourhood of the specified problem in the space of problems one is considering contains a variety of valid problems for the modeling task at hand. In the case where the problem one is considering is well-enough conditioned, and the quantities of interest vary continuously in that neighbourhood, then the entire neighbourhood contains valid problems. A central point, however, is that which nearby problems are truly valid is determined by the problems that physical and modeling perturbations push the specified problem onto. Thus, it is also these perturbations that determine which problems are just as valid as the specified one. So, it is only when the problem is well-enough conditioned, *and* the numerical error can reasonably be regarded as modeling the effects of physical error, that the numerical solution can be regarded as the exact solution to a problem that is just as valid as the one originally posed. It is worth pointing out that the latter condition may be relaxed in the case of a well-enough conditioned problem where the numerical error is known to be many orders of magnitude smaller than the dominant sources of physical error, but not in cases where the numerical error is not negligible compared to such sources of physical error. This section made the case that with the selection of numerical methods appropriate for the problem at hand, the numerical error can reasonably be understood as modeling the effects of physical error, and so if the problem is well-enough conditioned then the numerical solution can be understood to be just as valid as the exact solution to the problem originally posed.

There is a subtle and difficult issue that arises, however, when we do not know that the problem is well-enough conditioned, since it is possible that the problems solved by the numerics have different qualitative behaviour than not only the specified problem but also the problems in the neighbourhood that physical and modeling perturbations push this problem onto. Two responses to this issue when it cannot be analyzed rigorously, are the following. One is that Enright's notion of a problem that is 'chaotic in a practical sense' should generalize to a problem that is 'well-enough conditioned in a practical sense,' so that when the numerics survey a sample of problems in the neighbourhood of the specified problem and the quantities of interest appear to be well-enough conditioned, this gives us evidence that the problem is indeed well-enough conditioned. The second is that because physical perturbations can be both continuous and stochastic, it seems quite likely that physical perturbations cover a large range of the problems in the neighbourhood surrounding the specified problem. So, if the numerical error can reasonably thought to be modeling the effects of physical error, then we have good reason to think that the problems being solved by the numerics are included in the set of problems accessed by physical perturbations. Thus, in this case we have good reason to think that well-enough conditioning observed from the numerical solutions implies well-enough conditioning in the system being modeled. Thus, we may conclude that when well-enough conditioning is observed when a variety of the problems in the neighbourhood of the specified problem are solved numerically, and when the numerical error can reasonably be understood to be modeling the effects of physical error, there are strong reasons to believe two things: that the numerical solution is solving a problem that is just as valid as the one originally posed; and that insights drawn from the numerical solutions provide insights into the behaviour of the actual system being modeled.

Chapter 6

Conclusion

The three main approaches to BEA for ODE provide a powerful set of tools for the analysis of the error introduced in numerical solution of ODE problems. The analysis and control of the defect, which is computable in practice from a suitable interpolant, enables one to understand the numerical solution as the exact solution of a nearby ODE problem; provided that the defect is small, then we are assured that the numerical solution is a valid one. We have seen how the omnipresence of physical and modeling error enables us to understand the numerical solution as being just as valid as the exact solution to the problem originally posed, provided that the numerical perturbations can reasonably be regarded as physical, and that this is possible in most cases.

Although the analysis of the defect is legitimate in general, there are some special cases where we are interested in solutions to the specified problem and not a modified one. The shadowing results enable us to work with such a case by providing tools to establish when a numerical solution is shadowed by an exact solution with a modified initial condition. Since in practice, the initial condition is only known within some error bounds, these deep and powerful results enable proof that the numerical solution tracks an exact one that is just as valid as the solution with the specified initial condition provided that the initial condition of the shadowing solution is sufficiently close to the specified initial condition.

The method of modified equations has a number of important applications. It is a useful tool for understanding the behaviour of numerical methods by enabling the derivation of a modified problem that has solutions that the numerical solution follows more closely than the solutions to the original problem. It is also an important tool in the analysis and construction of geometric numerical methods for use on special problems where the flow of the specified equation has geometric properties that we want the numerical solution to preserve. And we have seen how it works well in conjunction with defect analysis in order to explain the structure of the defect.

The methods of BEA for ODE go beyond simply the analysis of numerical error. We may see that not only do the methods of BEA for ODE enable one to understand a numerical solution as the exact solution to a nearby problem, together they provide one with the flexibility to decide what *kind* of nearby problem the numerical solution solves. Analysis of the defect is the most flexible and multipurpose tool of the three, since it can be applied to any problem using any numerical method, and the control of the defect ensures that one has a valid solution. Shadowing results enable one to hold the equation fixed and understand the numerical solution as tracking a solution of the original equation. And the method of modified equations enables one to have a distinct amount of control over what kind of problem the numerics solve as a result of its role in the construction of geometric numerical integration methods.

We have also seen that, in keeping with the backward error point of view, they provide an important source of *insight* into the value of a numerical solution. They reduce the analysis of numerical error to an analysis of the effect of perturbations of the problem and or the data, enabling one to understand a numerical solution as an exact solution to a perturbed problem. Not only does this provide a clear and general way of understanding the effects of numerical error, the omnipresence of modeling and physical error has the effect that there is an entire neighbourhood of valid problems around the originally specified one. So, the use of BEA methods for ODE enables one to provide distinct criteria for validity, *e.g.*, that the defect is smaller than the largest sources of physical error, and consequently enables one to understand why a numerical solution is valid. In addition, as was argued in chapter 5, if the perturbations introduced by numerical error can be understood as *modeling* the effect of physical perturbations, then one is able to conclude that a numerical solution is *just as valid* as the exact solution to the problem originally posed. This shows not only that BEA for ODE is a powerful tool for the analysis of numerical error in the context of mathematical modeling, it also shows how computation can be understood, as it ought to be, as just another stage of the modeling process, not simply a device for getting approximate solutions from mathematical models.

Appendix A

Matlab Code

A.1 ode1d

```
function [tout,yout] = ode1d(F,tspan,y0,arg4,varargin)
%ODE1D  Solve non-stiff differential equations using a defect-controlled
%Euler method
%
%This code is a modified version of ODE23TX, from Numerical Computing
%with MATLAB by Cleve Moler, a textbook version of ODE23, written by
%Mark W. Richest and Lawrence F. Shampine
%
%   ODE1D(F,TSPAN,Y0) with TSPAN = [T0 TFINAL] integrates the system
%   of differential equations dy/dt = f(t,y) from t = T0 to t = TFINAL.
%   The initial condition is y(T0) = Y0.
%
%   The first argument, F, is a function handle or an anonymous function
%   that defines f(t,y).  This function must have two input arguments,
%   t and y, and must return a column vector of the derivatives, dy/dt.
%
%   With two output arguments, [T,Y] = ODE1D(...) returns a column
%   vector T and an array Y where Y(:,k) is the solution at T(k).
%
%   With no output arguments, ODE1D plots the emerging solution.
%
```

```
%   ODE1D(F,TSPAN,Y0,RTOL) uses the relative error tolerance RTOL
%   instead of the default 1.e-3.
%
%   ODE1D(F,TSPAN,Y0,OPTS) where OPTS = ODESET('reltol',RTOL, ...
%   'abstol',ATOL,'outputfcn',@PLOTFUN) uses relative error RTOL instead
%   of 1.e-3, absolute error ATOL instead of 1.e-6, and calls PLOTFUN
%   instead of ODEPLOT after each successful step.
%
%   More than four input arguments, ODE1D(F,TSPAN,Y0,RTOL,P1,P2,...),
%   are passed on to F, F(T,Y,P1,P2,...).

% Initialize variables.

rtol = 1.e-3;
atol = 1.e-6;
plotfun = @odeplot;
if nargin >= 4 & isnumeric(arg4)
   rtol = arg4;
elseif nargin >= 4 & isstruct(arg4)
   if ~isempty(arg4.RelTol), rtol = arg4.RelTol; end
   if ~isempty(arg4.AbsTol), atol = arg4.AbsTol; end
   if ~isempty(arg4.OutputFcn), plotfun = arg4.OutputFcn; end
end
t0 = tspan(1);
tfinal = tspan(2);
tdir = sign(tfinal - t0);
plotit = (nargout == 0);
threshold = atol / rtol;
hmax = abs(0.1*(tfinal-t0));
t = t0;
y = y0(:);

% Initialize output.

if plotit
   plotfun(tspan,y,'init');
else
   tout = t;
   yout = y.';
end
```

```
% Compute initial step size.

k0 = F(t, y, varargin{:});
r = norm(k0./max(abs(y),threshold),inf) + realmin;
h = tdir*0.8*rtol/r;

% The main loop.

while t ~= tfinal

   %hmin = 5e7*eps*abs(t);
   hmin = 16*eps*abs(t);
   if abs(h) > hmax, h = tdir*hmax; end
   if abs(h) < hmin, h = tdir*hmin; end

   % Stretch the step if t is close to tfinal.

   if 1.1*abs(h) >= abs(tfinal - t)
      h = tfinal - t;
   end

   % Attempt a step.

   tnew = t + h;
   ynew = y + h*k0;
   k1 = F(tnew, ynew, varargin{:});

   % Estimate the error.

   e = 3*abs(k1-k0)./4;
   err = norm(e./max(max(abs(y),abs(ynew)),threshold),inf) + realmin;
   %err = norm(e./max(abs(y),abs(ynew)),inf) + realmin;

   % Accept the solution if the estimated error is less than the tolerance.

   if err <= rtol
      t = tnew;
      y = ynew;
      if plotit
```

```
         if plotfun(t,y,'');
            break
         end
      else
         tout(end+1,1) = t;
         yout(end+1,:) = y.';
      end
      k0 = k1;     % Reuse final function value to start new step.
   end

   % Compute a new step size.

   h = h*min(5,0.8*rtol/err);

   % Exit early if step size is too small.

   if abs(h) <= hmin
      warning('Step size %e too small at t = %e.\n',h,t);
      t = tfinal;
   end
end

if plotit
   plotfun([],[],'done');
end
```

A.2 theta2

```
function [tout,yout] = theta2(F, DF, tspan, y0, h, theta, tol, Nmax)

%THETA2 - Fixed time two stage theta method for solving single or
%   systems of first-order ODE;  Newton's method is used to solve the
%   implicit equation which arises at each time step - For theta = 0 the
%   method is equivalent to forward Euler, for theta = 1 the method is
%   equivalent to backward Euler, and for theta = 0.5 the method is
%   equivalent to the implicit trapezoidal rule.
%
%This code is a modified version of ODE23TX, from Numerical Computing
%with MATLAB by Cleve Moler, a textbook version of ODE23, written by
```

```
%Mark W. Richest and Lawrence F. Shampine
%
%   THETA2(F,DF,TSPAN,Y0,H,TOL,NMAX) with TSPAN = [T0 TFINAL] integrates
%   the system of differential equations dy/dt = f(t,y) from t = T0 to
%   t = TFINAL. The initial condition is y(T0) = Y0.
%
%   The first argument, F, is a function handle or an anonymous function
%   that defines f(t,y).  This function must have two input arguments,
%   t and y, and must return a column vector of the derivatives, dy/dt.
%
%   The second argument, DF, is a function handle or an anonymous function
%   that defines the Jacobian Df(t,y) of f(t,y).  This function must have
%   two input arguments, t and y, and must return a matrix with the
%   derivatives, (df_i/dy_j)(t), as elements.
%
%   The fifth argument, H, is the fixed time step of the method
%
%   The sixth argument, TOL, is convergence tolerance applied to Newton's
%   method at each time step
%
%   The last argument, NMAX, is the maximum number of iterations of
%   Newton's method to be performed at each time step
%
%   With two output arguments, [T,Y] = THETA2(...) returns a column
%   vector T and a column vector Y where Y(k) is the solution at T(k).
%
%   With no output arguments, THETA2 plots the emerging solution.
%
%   Dependencies:
%
%   When applied to a system of equations, this routine makes use of
%   MATLAB's backslash operator to solve a linear system

plotfun = @odeplot;
t0 = tspan(1);
tfinal = tspan(2);
tdir = sign(tfinal - t0);
h=tdir*abs(h);
plotit = (nargout == 0);
t = t0;
```

```
y = y0(:);

% Initialize output.

if plotit
   plotfun(tspan,y,'init');
else
   tout = t;
   yout = y.';
end

% number of equations determines the method used

neqn = length(y0);

if (neqn == 1)
    while t ˜= tfinal

       % Stretch the step if t is close to tfinal.

       if abs(h) >= abs(tfinal - t)
          h = tfinal - t;
       end

       % Compute a step

       x = y;
   for j = 1:Nmax
           top = (x - y) - h * ((1 - theta)*F(t, y) + theta*F(t+h, x));
   bot = 1 - h * theta * DF(t+h, x);
   dx = top / bot;
   x = x - dx;
   if (abs(dx) < tol)
               break
           end
       end

       % Take a step.

   y = x;
   t = t + h;

       % Update output

        if plotit
          if plotfun(t,y,’’);
             break
          end
        else
          tout(end+1,1) = t;
          yout(end+1,:) = y.’;
        end
    end
else
    while t ~= tfinal

       % Stretch the step if t is close to tfinal.

       if abs(h) >= abs(tfinal - t)
          h = tfinal - t;
       end

       % Compute a step

       x = y;
       %w0 = y0;
   for j = 1:Nmax
           Fx = (x - y) - h * ((1 - theta)*F(t, y) + theta*F(t+h, x));
   DFx = eye(neqn) - h * theta * DF(t+h, x);
           dx = -DFx\Fx;
   x = x + dx;
   if (max(abs(dx)) < tol)
               break
           end
       end

       % Take a step

   y = x;
   t = t + h;
```

```
       % Update output

        if plotit
          if plotfun(t,y,'');
             break
          end
        else
          tout(end+1,1) = t;
          yout(end+1,:) = y.';
        end
    end
end

if plotit
   plotfun([],[],'done');
end
```

A.3 pchi

```
function [u,du] = pchi(tau,y,dy,t)
%PCHI  Piecewise cubic Hermite interpolation.
%  u = pchi(tau,y,dy,t) finds the continuously differentiable
%  piecewise cubic Hermite interpolant P(x), with P(tau(j)) = y(j) and
%  P'(tau(j)=dy(j), and returns u(k) = P(t(k)) and u'(k) = P'(t(k)).

%  sort the data
   n=length(tau);
%    [tau,index]=sort(tau);
%    y=y(index);
%    dy=dy(index);

%  Function and derivative values at interval end points

   yn  = y(1:n-1);
   fn  = dy(1:n-1);
   yn1 = y(2:n);
   fn1 = dy(2:n);
```

```
%  Find subinterval indices k so that x(k) <= u < x(k+1)

   k = ones(size(t));
   for j = 2:n-1
      k(tau(j) <= t) = j;
   end

%  Evaluate interpolant

   h = diff(tau);
   s = (t - tau(k))./h(k);
   u = (s-1).^2.*(2*s+1).*yn(k) + s.*(s-1).^2.*h(k).*fn(k) + ...
       s.^2.*(-2*s+3).*yn1(k) + s.^2.*(s-1).*h(k).*fn1(k);

%  Evaluate derivative of interpolant

   du = 6*s.*(s-1)./h(k).*yn(k) + (s-1).*(3*s-1).*fn(k) + ...
       2*s.*(-3*s+3)./h(k).*yn1(k) + s.*(3*s-2).*fn1(k);

end
```

Appendix B

Curriculum Vitae

Robert Hugh Caldwell Moir

robert@moir.net · robert.moir.net

Education

PhD, Philosophy 2004-2011
The University of Western Ontario, London, Canada
Thesis: *The Reasonable Effectiveness of Mathematics: An Examination of the Interrelation of Mathematical Structures and Physical Reality*
Supervisors: John Bell and Robert Batterman

MSc, Applied Mathematics 2009-2010
The University of Western Ontario, London, Canada
Thesis: *Reconsidering Backward Error Analysis for Ordinary Differential Equations*
Supervisor: Robert Corless

MA, Philosophy 2003-2004
The University of Western Ontario, London, Canada

BA, Mathematics and Philosophy (*First Class Joint Honours*) 2001-2003
McGill University, Montréal, Canada
Honours Thesis: *Infinity and Physical Theory*
Supervisor: Michael Hallett

BSc, Physics (minor Chemistry) (*First Class Honours*) 1995-2001

McGill University, Montréal, Canada

Areas of Specialization

Philosophy of Physics, Philosophy of Applied Mathematics

Areas of Competence

Logic, Philosophy of Mathematics, Philosophy of Science, Applied Mathematics

Awards and Distinctions

Research Awards
The University of Western Ontario

- Western Graduate Research Scholarship ($8,000), 2009-2010

Social Sciences and Humanities Research Council of Canada

- Doctoral Fellowship ($40,000), 2007-2009

The University of Western Ontario

- Western Graduate Research Scholarship ($8,000), 2005-2006

The University of Western Ontario

- Special University Scholarship ($13,000), 2003-2005

Academic Awards
Chemical Institute of Canada National High School Chemistry Examination

- Toronto District Winner, 1995

Publications

Proceedings

Moir, R. (2009). The Conversion of Phenomena to Theory: Lessons on Applicability from the Early Development of Electromagnetism. In: A. Cupillari (Ed.), *Proceedings of the Canadian Society for History and Philosophy of Mathematics, St John's NL, June 2009*, pp. 68-91.

Talks **Peer-reviewed *Abstract Submission

1. ** with Nicolas Fillion, "Explanation and Abstraction: The Case of Backward Error Analysis" Philosophy of Science Association Biennial Meeting, Montréal, Québec, 4-6 November 2010.
2. * with Nicolas Fillion, "Modeling and Explanation: Lessons from Modern Error Theory." Canadian Society for the History and Philosophy of Science (CSHPS) Conference, Concordia University, Montréal, Québec, 28-30 May 2010.
3. with Nicolas Fillion, "A Step Forward with Backward Error," PGSA Colloquium Series, Department of Philosophy, The University of Western Ontario, 12 March 2010.
4. * "The Conversion of Phenomena to Theory: Lessons on Applicability from the Development of Electromagnetism." Canadian Mathematical Society/Canadian Society for the History and Philosophy of Mathematics (CMS/CSHPM) Conference, Memorial University, St. John's, Newfoundland, 6-8 June 2009.
5. "From the World to Mathematics and Back Again: What We Can Understand About Applicability from the Development of Electromagnetism." PGSA Colloquium Series, Department of Philosophy, The University of Western Ontario, 25 March 2009.
6. * "Theories, Models and Representation: Lessons from Solid State Physics." Canadian Society for the History and Philosophy of Science (CSHPS) Conference, University of British Columbia, Vancouver, British Columbia, 3-5 June 2008.
7. "Theories, Models and Representation: Lessons from Solid State Physics." PGSA Colloquium Series, Department of Philosophy, University of Western Ontario, 12 March 2008.
8. "Interpretations of Probability in Quantum Mechanics." PGSA Conference, Department of Philosophy, University of Waterloo, June 2005.

Academic Experience

Instructor

The University of Western Ontario 2010-2011

- Critical Thinking (Full-Year Course), 2010–2011

Teaching Assistant

The University of Western Ontario 2003-2010

- Linear Algebra for Engineers (Half-Year Course), 2010
- Calculus (Half-Year Course), 2009
- Introduction to Philosophy (Full-Year Course), 2005–2006, 2007–2008
- Critical Thinking and Reasoning (Full-Year Course), 2003–2005

Research Assistant

The University of Western Ontario 2007-2010

- Robert Corless, Department of Applied Mathematics, 2009–2010
- Rotman Canada Research Chair in Philosophy of Science, 2009–2010
- The Joseph L. Rotman Institute for Science and Values, 2008-2009

Service

Conference Organization

- Logic, Mathematics, and Physics Graduate Philosophy Conference
  Department of Philosophy, University of Western Ontario
  2006: Co-organizer with J. Noland and D. MacDonald. Keynote Speaker: Michael Hallett (McGill University)

Main References

John L. Bell
Fellow of the Royal Society of Canada
Department of Philosophy
The University of Western Ontario
London, ON
N6A 5B8 Canada
Tel: (519) 661-2111 ext. 85750
E-mail: jbell@uwo.ca

Robert W. Batterman
Fellow of the Royal Society of Canada
Department of Philosophy
University of Pittsburgh
Pittsburgh, PA
15260 USA
Tel: (412) 624-5782
E-mail: rbatterm@pitt.edu

Robert Corless
Department of Applied Mathematics
The University of Western Ontario
London, ON
N6A 5B7 Canada
Tel: (519) 661-2111 ext. 88785
E-mail: rcorless@uwo.ca

Bibliography

Ahmed, M.O., & Corless, R.M. 1997. The method of modified equations in Maple. *In: Electronic Proc. 3rd Int. IMACS Conf. on Applications of Computer Algebra*.

Al-Nayef, A.A., Kloeden, P.E., & Pokrovskii, A.V. 1997. Semi-Hyperbolic Mappings, Condensing Operators, and Neutral Delay Equations. *Journal of Differential Equations*, **137**(2), 320–339.

Anosov, D.V. 1967. Geodesic flows on closed Riemannian manifolds of negative curvature. *Trudy Matematicheskogo Instituta im. VA Steklova*, **90**, 3–210.

Ascher, U., Christiansen, J., & Russell, R. 1979. ColsysA collocation code for boundary-value problems. *Codes for Boundary-Value Problems in Ordinary Differential Equations*, 164–185.

Ascher, U.M., Mattheij, R.M.M., & Russell, R.D. 1988. *Numerical solution of boundary value problems for ordinary differential equations*. Prentice-Hall.

Bader, G., & Ascher, U. 1987. A new basis implementation for a mixed order boundary value ODE solver. *SIAM Journal on Scientific and Statistical Computing*, **8**, 483.

Birkhoff, G., & Rota, G.C. 1989. *Ordinary differential equations*. Wiley & Sons.

Blanes, S., & Budd, C.J. 2005. Adaptive geometric integrators for Hamiltonian problems with approximate scale invariance. *SIAM Journal on Scientific Computing*, **26**(4), 1089–1113.

Bond, S.D., & Leimkuhler, B.J. 2007. Stabilized Integration of Hamiltonian Systems with Hard-Sphere Inequality Constraints. *SIAM Journal on Scientific Computing*, **30**(1), 134–147.

Bowen, R. 1975. Limit sets for Axiom A diffeomorphisms. *Journal of Differential Equations*, **18**(2), 333–339.

Butcher, J.C. 1987. *The numerical analysis of ordinary differential equations: Runge-Kutta and general linear methods*. Wiley-Interscience New York, NY, USA.

Calvo, M.P., Murua, A., & Sanz-Serna, J.M. 1994. Modified equations for ODEs. *Page 63 of: Chaotic numerics: an International Workshop on the Approximation and Computation of Complicated Dynamical Behavior, Deakin University, Geelong, Australia, July 12-16, 1993*, vol. 173. American Mathematical Society.

Cash, J.R., & Silva, H.H.M. 1993. On the numerical solution of a class of singular two-point boundary value problems. *Journal of Computational and Applied Mathematics*, **45**(1-2), 91–102.

Char, B.W., Geddes, K.O., Gonnet, G.H., Leong, B.L., Monagan, M.B., & Watt, S.M. 1991. *Maple V library reference manual.* Springer-verlag New York.

Chartier, P., Hairer, E., & Vilmart, G. 2007. Numerical integrators based on modified differential equations. *Mathematics of computation*, **76**(260), 1941.

Chow, S.N., & Palmer, K.J. 1991. On the numerical computation of orbits of dynamical systems: the one-dimensional case. *Journal of Dynamics and Differential equations*, **3**(3), 361–379.

Chow, S.N., & Palmer, K.J. 1992. On the numerical computation of orbits of dynamical systems: the higher dimensional case. *Journal of Complexity*, **8**(4), 398–423.

Chow, S.N., & Van Vleck, E.S. 1992. A shadowing lemma for random diffeomorphisms. *Random & Computational Dynamics*, **1**(2), 197–218.

Chow, S.N., & Van Vleck, E.S. 1994. Shadowing of lattice maps. *Page 97 of: Chaotic numerics: an International Workshop on the Approximation and Computation of Complicated Dynamical Behavior, Deakin University, Geelong, Australia, July 12-16, 1993*, vol. 172. Amer Mathematical Society.

Coomes, B.A. 1997. Shadowing orbits of ordinary differential equations on invariant submanifolds. *Transactions of the American Mathematical Society*, **349**(1), 203–216.

Coomes, B.A., Kogak, H., & Palmer, K.J. 1994a. Periodic shadowing. *Page 115 of: Chaotic numerics: an International Workshop on the Approximation and Computation of Complicated Dynamical Behavior, Deakin University, Geelong, Australia, July 12-16, 1993*, vol. 172. American Mathematical Society.

Coomes, B.A., Koçak, H., & Palmer, K.J. 1994b. Shadowing orbits of ordinary differential equations. *Journal of Computational and Applied Mathematics*, **52**(1-3), 35–43.

Coomes, B.A., Koçak, H., & Palmer, K.J. 1995. Rigorous computational shadowing of orbits of ordinary differential equations. *Numerische Mathematik*, **69**(4), 401–421.

Coomes, B.A., Koçak, H., & Palmer, K.J. 1997. Long periodic shadowing. *Numerical Algorithms*, **14**(1), 55–78.

Corless, R.M. 1992a. Continued fractions and chaos. *American Mathematical Monthly*, **99**(3), 203–215.

Corless, R.M. 1992b. Defect-controlled numerical methods and shadowing for chaotic differential equations. *Physica D: Nonlinear Phenomena*, **60**(1-4), 323–334.

Corless, R.M. 1994a. Error backward. *Page 31 of: Chaotic numerics: an International Workshop on the Approximation and Computation of Complicated Dynamical Behavior, Deakin University, Geelong, Australia, July 12-16, 1993*, vol. 172. American Mathematical Society.

Corless, R.M. 1994b. What good are numerical simulations of chaotic dynamical systems? *Computers & Mathematics with Applications*, **28**(10-12), 107–121.

Corless, R.M., & Corliss, G.F. 1992. Rationale for guaranteed ODE defect control. *Computer Arithmetic and Enclosure Methods, L. Atanassova and J. Herzberger, eds., North-Holland, Amsterdam*, 3–12.

Corless, R.M., & Pilyugin, S.Y. 1995. Approximate and real trajectories for generic dynamical systems. *Journal of Mathematical Analysis and Applications*, **189**(2), 409–423.

Coven, E.M., Kan, I., & Yorke, J.A. 1988. Pseudo-orbit shadowing in the family of tent maps. *Transactions of the American Mathematical Society*, **308**(1), 227–241.

Dieci, L., & Vleck, E.S.V. 2005. On the error in computing Lyapunov exponents by QR methods. *Numerische Mathematik*, **101**(4), 619–642.

Eirola, T. 1993. Aspects of backward error analysis of numerical ODEs. *Journal of Computational and Applied Mathematics*, **45**(1-2), 65–73.

Enright, W.H. 1989a. A new error-control for initial value solvers. *Appl. Math. Comput.*, **31**, 288–301.

Enright, W.H. 1989b. Analysis of error control strategies for continuous Runge-Kutta methods. *SIAM Journal on Numerical Analysis*, **26**(3), 588–599.

Enright, W.H. 1993. The relative efficiency of alternative defect control schemes for high-order continuous Runge-Kutta formulas. *SIAM Journal on Numerical Analysis*, **30**(5), 1419–1445.

Enright, W.H. 2002. The design and implementation of usable ODE software. *Numerical Algorithms*, **31**(1), 125–137.

Enright, W.H. 2010. *The Numerical Solution of Ordinary Differential Equations.* `http://www.cs.utoronto.ca/~enright/teaching/CSC2302/IVP.ps`. Course notes for CSC2302H at the University of Toronto, Fall 2010.

Enright, W.H., & Hayashi, H. 1997. A delay differential equation solver based on a continuous Runge–Kutta method with defect control. *Numerical Algorithms*, **16**(3), 349–364.

Enright, W.H., & Hayashi, H. 1998. Convergence analysis of the solution of retarded and neutral delay differential equations by continuous numerical methods. *SIAM Journal of Numerical Analysis*, **35**(2), 572–585.

Enright, W.H., & Hayes, W.B. 2007. Robust and reliable defect control for Runge-Kutta methods. *ACM Transactions on Mathematical Software (TOMS)*, **33**(1), 1.

ENRIGHT, W.H., & MUIR, P.H. 1996. Runge-Kutta software with defect control for boundary value ODEs. *SIAM Journal on Scientific Computing*, **17**(2), 479–497.

ENRIGHT, W.H., & MUIR, P.H. 2010. New interpolants for asymptotically correct defect control of BVODEs. *Numerical Algorithms*, **53**(2), 219–238.

ENRIGHT, W.H., JACKSON, K.R., NØRSETT, S.P., & THOMSEN, P.G. 1986. Interpolants for runge-kutta formulas. *ACM Transactions on Mathematical Software (TOMS)*, **12**(3), 218.

FALTINSEN, S. 2000. Backward error analysis for Lie-group methods. *BIT Numerical Mathematics*, **40**(4), 652–670.

FENG, K. 1991. Formal power series and numerical algorithms for dynamical systems. *Pages 28–35 of: Proc. Conf. Scientific Computation Hangzhou.*

FOX, L. 1987. James Hardy Wilkinson. *Biographical Memoirs of Fellows of the Royal Society*, **33**(11), 671–708.

FOX, L., & MAYERS, D.F. 1968. *Computing methods for scientists and engineers.* Oxford.

FRANKE, J.E., & SELGRADE, J.F. 1977. Hyperbolicity and chain recurrence. *Journal of Differential Equations*, **26**(1), 27–36.

GARABEDIAN, P.R. 1956. Estimation of the relaxation factor for small mesh size. *Mathematical Tables and Other Aids to Computation*, **10**(56), 183–185.

GIVENS, W. 1954. Numerical computation of the characteristic values of a real symmetric matrix. *ORNL-1574, Oak Ridge National Laboratory.*

GONZALEZ, O., HIGHAM, D.J., & STUART, A.M. 1999. Qualitative properties of modified equations. *IMA Journal of Numerical Analysis*, **19**(2), 169.

GREBOGI, C., HAMMEL, S.M., YORKE, J.A., & SAUER, T. 1990. Shadowing of physical trajectories in chaotic dynamics: Containment and refinement. *Physical Review Letters*, **65**(13), 1527–1530.

GRIFFITHS, D.F. 1988. *The dynamics of some linear multistep methods with step-size control. Appears in Numerical Analysis 1987 Eds: Griffiths, D.F. and Watson, G.A.*

GRIFFITHS, D.F., & SANZ-SERNA, J.M. 1986. On the scope of the method of modified equations. *SIAM Journal on Scientific and Statistical Computing*, **7**, 994.

HAIRER, E. 1994. Backward analysis of numerical integrators and symplectic methods. *Ann. Numer. Math.*, **1**(1-4), 107–132.

HAIRER, E. 1997. Variable time step integration with symplectic methods. *Applied Numerical Mathematics*, **25**(2-3), 219–227.

HAIRER, E. 2003. Global modified Hamiltonian for constrained symplectic integrators. *Numerische Mathematik*, **95**(2), 325–336.

Hairer, E. 2005. Important aspects of geometric numerical integration. *Journal of Scientific Computing*, **25**(1), 67–81.

Hairer, E., & Lubich, C. 1997. The life-span of backward error analysis for numerical integrators. *Numerische Mathematik*, **76**(4), 441–462.

Hairer, E., & Soderlind, G. 2005. Explicit, time reversible, adaptive step size control. *SIAM Journal on Scientific Computing*, **26**(6), 1838–1851.

Hairer, E., & Vilmart, G. 2006. Preprocessed discrete Moser–Veselov algorithm for the full dynamics of a rigid body. *Journal of Physics A: Mathematical and General*, **39**, 13225.

Hairer, E., & Wanner, G. 1974. On the Butcher group and general multi-value methods. *Computing*, **13**(1), 1–15.

Hairer, E., Lubich, C., & Wanner, G. 2006. *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*. Springer Verlag.

Hammel, S.M., Yorke, J.A., & Grebogi, C. 1987. Do numerical orbits of chaotic dynamical processes represent true orbits? *Journal of Complexity*, **3**(2), 136–145.

Hammel, S.M., Yorke, J.A., & Grebogi, C. 1988. Numerical orbits of chaotic dynamical processes represent true orbits. *Bulletin of the American Mathematical Society*, **19**, 465–470.

Hanson, P.M., & Enright, W.H. 1983. Controlling the defect in existing variable-order Adams codes for initial-value problems. *ACM Transactions on Mathematical Software (TOMS)*, **9**(1), 97.

Hayes, W. 1995. *Efficient shadowing of high dimensional chaotic systems with the large astrophysical n-body problem as an example*. M.Sc. thesis, University of Toronto.

Hayes, W. 2001. *Rigorous shadowing of numerical solutions of ordinary differential equations by containment*. Ph.D. thesis, University of Toronto.

Hayes, W., & Jackson, K.R. 2005. A survey of shadowing methods for numerical solutions of ordinary differential equations. *Applied Numerical Mathematics*, **53**(2-4), 299–321.

Hayes, W.B., & Jackson, K.R. 2003. Rigorous shadowing of numerical solutions of ordinary differential equations by containment. *SIAM Journal on Numerical Analysis*, **41**(5), 1948–1973.

Higham, D.J. 1989a. Defect estimation in Adams PECE codes. *SIAM Journal on Scientific and Statistical Computing*, **10**, 964.

Higham, D.J. 1989b. Robust defect control with Runge-Kutta schemes. *SIAM Journal on Numerical Analysis*, **26**(5), 1175–1183.

Higham, D.J. 1991a. Global error versus tolerance for explicit Runge-Kutta methods. *IMA Journal of Numerical Analysis*, **11**(4), 457.

Higham, D.J. 1991b. Runge–Kutta defect control using Hermite–Birkhoff interpolation. *SIAM Journal on Scientific and Statistical Computing*, **12**, 991.

Higham, D.J., & Stuart, A.M. 1998. Analysis of the dynamics of local error control via a piecewise continuous residual. *BIT Numerical Mathematics*, **38**(1), 44–57.

Hirt, C.W. 1968. Heuristic stability theory for finite-difference equations. *Journal of Computational Physics*, **2**(4), 339–355.

Hull, T.E. 1968. The Numerical Integration of ordinary differential equations. *Pages 134–144 of: Proc. Information Processing 68.*

Hull, T.E. 1970. The effectiveness of numerical methods for ordinary differential equations. *Studies in Numerical Analysis*, **2**, 114–121.

Kierzenka, J., & Shampine, L.F. 2001. A BVP solver based on residual control and the Maltab PSE. *ACM Transactions on Mathematical Software (TOMS)*, **27**(3), 316.

Lin, X.B. 1989. Shadowing lemma and singularly perturbed boundary value problems. *SIAM Journal on Applied Mathematics*, 26–54.

Liu, W. 2005. Geometric approach to a singular boundary value problem with turning points. *Dynamical Systems*, 624–633.

Lohner, R.J. 1987. Enclosing the solutions of ordinary initial and boundary value problems. *Computer Arithmetic: Scientific Computation and Programming Languages*, 255–286.

MacDonald, C. 2000. *A new approach for DAEs*. Ph.D. thesis, University of Toronto.

Morton, K.W. 1977. Initial-value problems by finite-difference and other methods. *The State of the Art in Numerical Analysis*, 699–756.

Muir, P.H., Pancer, R.N., & Jackson, K.R. 2003. PMIRKDC: a parallel mono-implicit Runge-Kutta code with defect control for boundary value ODEs. *Parallel Computing*, **29**(6), 711–741.

Nguyen, H. 1995. *Interpolation and error control schemes for algebraic differential equations using continuous implicit Runge-Kutta methods*. Ph.D. thesis, Citeseer.

Nusse, H.E., & Yorke, J.A. 1988. Is every approximate trajectory of some process near an exact trajectory of a nearby process? *Communications in Mathematical Physics*, **114**(3), 363–379.

Odani, K. 1990. Generic homeomorphisms have the pseudo-orbit tracing property. *Proceedings of the American Mathematical Society*, **110**(1), 281–284.

OSBORNE, M.R. 1964. An error analysis of finite-difference methods for the numerical solution of ordinary differential equations. *The Computer Journal*, **7**(3), 232–237.

PILYUGIN, S.Y. 1999. *Shadowing in dynamical systems*. Springer Verlag.

QUINLAN, G.D., & TREMAINE, S. 1992. On the reliability of gravitational N-body integrations. *Monthly Notices of the Royal Astronomical Society*, **259**, 505–518.

REDDIEN, G.W. 1995. On the stability of numerical methods of Hopf points using backward error analysis. *Computing*, **55**(2), 163–180.

REICH, S. 1997. On higher-order semi-explicit symplectic partitioned Runge-Kutta methods for constrained Hamiltonian systems. *Numerische Mathematik*, **76**(2), 231–247.

REICH, S. 1999. Backward error analysis for numerical integrators. *SIAM Journal on Numerical Analysis*, 1549–1570.

SANZ-SERNA, J.M. 1992. Symplectic integrators for Hamiltonian problems: an overview. *Acta Numerica*, **1**, 243–286.

SANZ-SERNA, J.M., & CALVO, M.P. 1994. *Numerical hamiltonian problems*. Chapman & Hall/CRC.

SANZ-SERNA, J.M., & LARSSON, S. 1993. Shadows, chaos, and saddles. *Applied Numerical Mathematics*, **13**(1-3), 181–190.

SAUER, T., & YORKE, J.A. 1991. Rigorous verification of trajectories for the computer simulation of dynamical systems. *Nonlinearity*, **4**, 961.

SHAMPINE, L., MUIR, P., & XU, H. 2006. A user-friendly Fortran BVP solver. *JNAIAM*, **1**(2), 201–217.

SHAMPINE, L.F. 2005. Solving ODEs and DDEs with residual control. *Applied Numerical Mathematics*, **52**(1), 113–127.

SHAMPINE, L.F. 2007. Design of software for ODEs. *Journal of Computational and Applied Mathematics*, **205**(2), 901–911.

SHAMPINE, L.F., & MUIR, P.H. 2004. Estimating conditioning of BVPs for ODEs. *Mathematical and Computer Modelling*, **40**(11-12), 1309–1321.

SHAMPINE, L.F., & WATTS, H.A. 1976. Global Error Estimates for Ordinary Differential Equations. *ACM Transactions on Mathematical Software (TOMS)*, **2**(2), 172–186.

SHAMPINE, L.F., & WATTS, H.A. 1980. *DEPAC-Design of a user oriented package of ODE solvers*. Tech. rept. SAND-79-2374, Sandia National Labs., Albuquerque, NM (USA).

SHIMADA, I., & NAGASHIMA, T. 1979. A numerical approach to ergodic problem of dissipative dynamical systems. *Prog. Theor. Phys*, **61**(6), 1605–1616.

SÖDERLIND, G. 2003. Digital filters in adaptive time-stepping. *ACM Transactions on Mathematical Software (TOMS)*, **29**(1), 26.

STETTER, H. 1976. Considerations concerning a theory for ODE-solvers. *Numerical Treatment of Differential Equations*, 188–200.

STETTER, H.J. 1981. Tolerance proportionality in ODE-codes. *Seminarber., Humboldt-Univ. Berlin, Sekt. Math.*, **32**, 109–123.

STETTER, H.J. 2004. *Numerical polynomial algebra.* Society for Industrial Mathematics.

STEWART, N.F. 1970. Certain Equivalent Requirements of Approximate Solutions of x'=f(t,x). *SIAM Journal on Numerical Analysis*, 256–270.

VAN VLECK, E.S. 1995. Numerical shadowing near hyperbolic trajectories. *SIAM Journal on Scientific Computing*, **16**, 1177.

VON NEUMANN, J., & GOLDSTINE, H.H. 1947. Numerical inverting of matrices of high order. *Bull. Amer. Math. Soc*, **53**(11), 1021–1099.

WARMING, R.F., & HYETT, B.J. 1974. The modified equation approach to the stability and accuracy analysis of finite-difference methods. *Journal of Computational Physics*, **14**(2), 159–179.

WILKINSON, J.H. 1963. Rounding errors in algebraic processes, volume 32. *Her Majesty's stationery office, London.*

WILKINSON, J.H. 1971. Modern error analysis. *SIAM review*, **13**(4), 548–568.

ZADUNAISKY, P.E. 1966. A method for the estimation of errors propagated in the numerical solution of a system of ordinary differential equations. *Page 281 of: The Theory of Orbits in the Solar System and in Stellar Systems*, vol. 25.