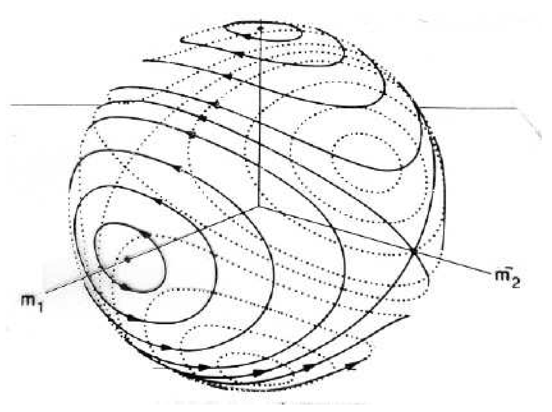
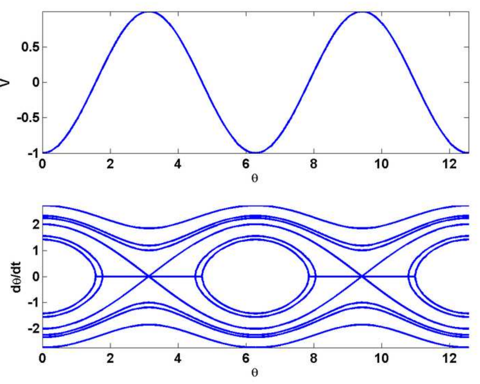
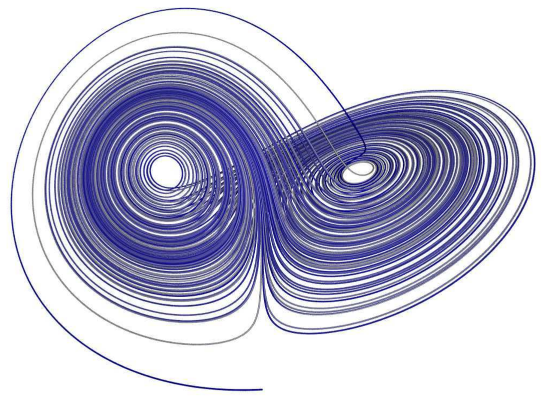






Dynamics Backward

Backward Error Analysis for Ordinary Differential Equations

Robert H C Moir

robert@moir.net

Western

Advantage of Backward Error Analysis

The main advantage of backward error analysis for models using ordinary differential equations (ODE) is that it enables one to find exact solutions to models of real world systems that are *just as valid* as posed models that cannot be solved exactly. This is done by treating numerical error as a perturbation just like physical perturbations, and by ensuring that the perturbations are small and *model* physical perturbations.

Modeling and Error

In many situations one can formulate useful ODE models of real world systems of the form

$$\frac{dy}{dt} \equiv \dot{y} = f(y,t), \tag{1}$$

with $f(y,t)$ a nonlinear vector field. There are two main kinds of error that enter in here. A background of theories chosen to formulate the model determines a vector field $g(y,t)$ that we usually do not know or cannot calculate. Thus, idealizing assumptions and mathematical simplifications introduce *modeling error* $\mu(y,t)$ when the model is generated:

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

Physical error $\pi(y,t)$ is always present because only part of the world is included in the model. Interactions between the model system and its environment perturb the model system. So, the actual system is described by some perturbed model:

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

Numerical Perturbation

Rather than being considered the result of generating approximate solutions, numerical error can be considered as a kind of perturbation of the ODE (1).

The use of floating point arithmetic to compute $f(y,t)$ introduces *floating point error* $\phi(y,t)$. And the discretization involved in the use of numerical methods introduces *truncation error* $\tau(y,t)$. A C^1 interpolant $u(t)$ of the numerical solution, which we can always obtain, then satisfies

$$\dot{u} = f(u,t) + \phi(u,t) + \tau(u,t) = f(u,t) + \delta(u,t). \tag{2}$$

Thus, numerical error can be seen as a perturbation $\delta(u,t)$ of the original equation (vector field) (1). $u(t)$ is the *exact solution* of the modified model (2).

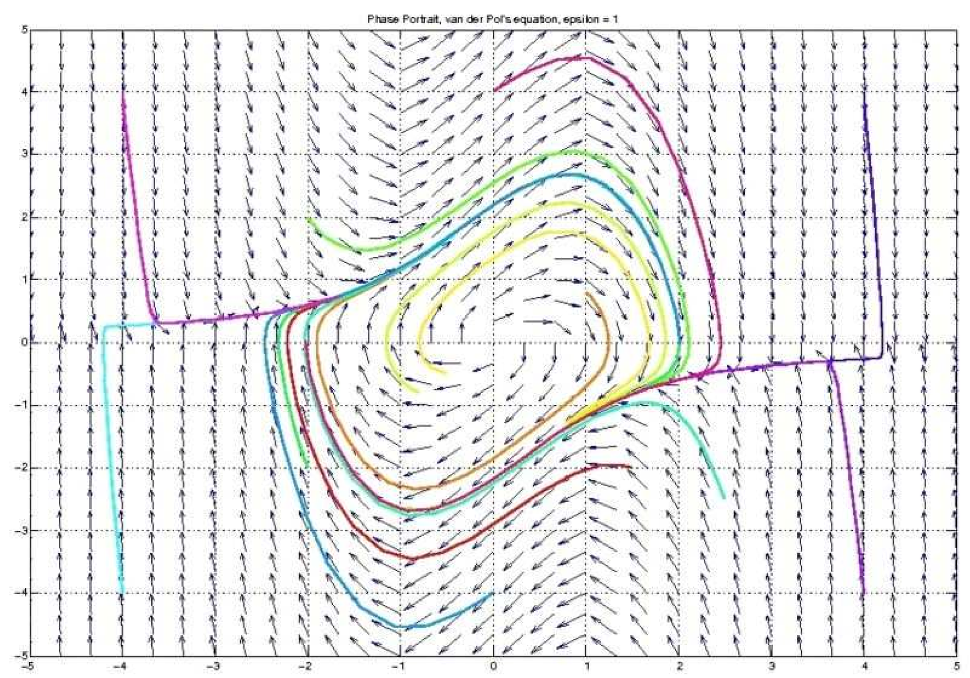
Backward Error for ODE

When obtaining a C^1 numerical solution $u(t)$ we usually want to ensure that the *global error* $y(t) - u(t)$ is small. This is the *forward error*, the difference between the actual solution and the computed one. Without knowing the exact solution $y(t)$, however, we cannot calculate the error.

The approach of backward error analysis is to find a *modified problem* to which $u(t)$ is the *exact* solution. This done by calculating the *backward error* or *defect* $\delta(u,t)$:

$$\delta(u,t) := \dot{u} - f(u,t), \tag{3}$$

the same quantity appearing in equation (2).



Phase portrait for the van del Pol equation, with vector field (arrows) and solutions (curves) displayed.

A small defect translates into a small global error provided that small perturbations of the ODE (1) lead to only small variations of the solution. Equations that have this property are called *well-conditioned*. In such a case, controlling the defect indirectly controls the global error.

Many problems, however, are *ill-conditioned*, *i.e.* small perturbations result in large variations of the solution. Chaotic problems, due to their sensitivity to initial conditions, are classic examples of ill-conditioned problems. For such problems we cannot control the global error, but BEA can still be useful (see 'well-enough conditioning').

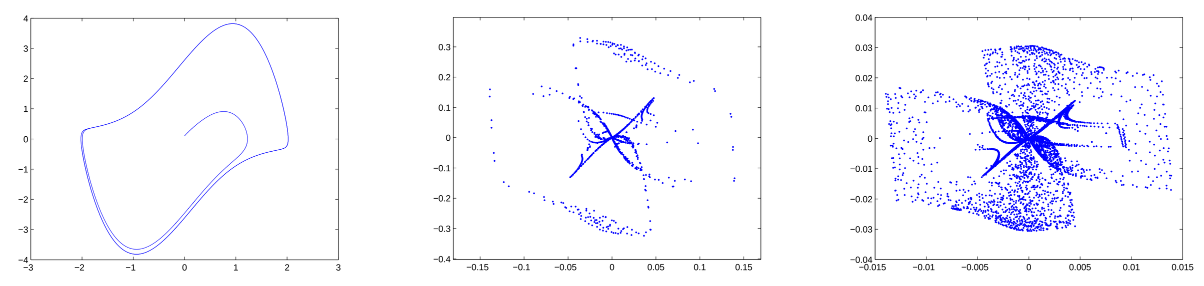
Forms of BEA for ODE

Defect Control

Defect analysis works by modifying the (vector field of the) ODE (1) and holding the (initial, boundary, or algebraic) conditions on it fixed. This is a natural approach since C^1 interpolants can be computed for any numerical method, enabling the defect to be used to control the error of the method.

The images below are the result of applying a Matlab code `ode1d`, a defect-controlled Euler method using piecewise cubic Hermite interpolation, to the unforced van der Pol equation with $\epsilon = 2$.

Van der Pol equation solved using defect-controlled `ode1d`



phase trajectory | $\delta(t)$ for TOL $= 0.1$ | $\delta(t)$ for TOL $= 0.01$

Shadowing

Shadowing works for initial value problems by modifying the initial conditions of a problem while leaving the equation (vector field) 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 method is well 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.

Method of Modified Equations

This approach works by modifying both the equation (vector field) and the conditions on it. This approach uses both the specified equation and the equations defining the numerical method to determine the equation and conditions of a *modified problem*, which models the behaviour of the numerical solution better than the original problem does.

This approach is useful in order to better understand the behaviour of numerical methods and in the generation of numerical methods that preserve the geometric properties of certain problems, *e.g.* Hamiltonian problems.

Well-Enough Conditioning

For models of real world systems we require that whichever quantities are of interest, which need not be the forward error, are stable under perturbations of the problem. This kind of stability is necessary for a model to be useful at all, because modeling and physical error are always present to some degree.

Problems where quantities of interest do not vary significantly for small perturbations of the problem are called *well-enough conditioned*.

The reason that BEA is useful on chaotic problems is that provided the problem is well-enough conditioned, *e.g.* the first Lyapunov exponent is stable, then by keeping the defect smaller than sources of physical perturbation, we can ensure that the numerical solution is the exact solution to just as valid a problem as the problem initially specified.

Obtaining Valid Solutions

Consider a *space* of problems of the form (1), where the points are the possible vector fields $f(y,t)$. The presence of physical perturbations ensures that there is an entire *neighbourhood of valid problems* around the specified one, the size of which is determined by the size of $\pi(y,t)$ (see 'modeling and error').

Thus, provided we can ensure that $\|\delta(u,t)\| \ll \|\pi(y,t)\|$, which we can with the precision available on most computers, we can be assured that our numerical solution solves a valid problem exactly.

To be sure that we have solved *just as valid* a problem, we must in general ensure that the numerical perturbation *models* a reasonable physical perturbation. This is usually possible; for example, many numerical methods can be made to dissipate energy very slowly (using geometric numerical methods).

References

[1] R.M. Corless (1994) Error Backward *Chaotic Numerics.* American Mathematical Society, p. 31.

[2] R.H.C. Moir (2010) Reconsidering Backward Error Analysis for Ordinary Differential Equations UWO, M.Sc. Thesis.