Pergamon

Computers Math. Applic. Vol. 28, No. 10–12, pp. 107–121, 1994
Copyright©1994 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0898-1221/94 $7.00 + 0.00

0898-1221(94)00188-X

What Good Are Numerical Simulations of Chaotic Dynamical Systems?

R. M. CORLESS
Department of Applied Mathematics, University of Western Ontario
London, Canada N6A 5B7

Dedicated to the memory of M. A. H. Nerenberg (1936–1993)

Abstract—Numerical simulations of mathematical models can suggest that the models are chaotic. For example, one can compute an orbit and its associated finite-time Lyapunov exponents, and these computed exponents can be positive. It is not clear how far these suggestions can be trusted, because, as is well known, numerical methods can introduce spurious chaos or even suppress actual chaos. This focused review examines the fidelity of numerical methods. We look at the didactic example of the Gauss map from the theory of continued fractions, which allows a simple examination of *backward error analysis* for discrete dynamical systems and gives a clear picture of the effects of floating-point arithmetic. A similar use of backward error analysis, in the form of *defect control*, gives a useful understanding in the case of continuous dynamical systems. Finally, we discuss limitations of this 'backward' point of view.

1. INTRODUCTION AND MOTIVATION

This study of the interaction of numerical analysis and chaotic dynamical systems was originally motivated by a problem in flow-induced vibration. Flow-induced vibration has a huge literature and, in general, many extremely important practical consequences [1]. Our interest was mainly in a particular semi-empirical mathematical model of the flow-induced cross-stream vibration of a square prism under two-dimensional conditions [2–4].

It turns out that the *physical* problem being modelled can, under certain circumstances, be regarded as being chaotic [5,6]. Approximate analytical and numerical results suggested that the rather oversimplified mathematical model being examined might also be chaotic [7], again under certain circumstances. The agreement between the model and experiment was not good enough for anyone to worry about that particular calculation, but did suggest that when a sufficiently realistic mathematical model is finally made available, one had better be able to solve it reliably, even when the true solution of the model equations is chaotic.

The difficulty is that the trajectories of a chaotic differential equation are ill-conditioned; that is to say, tiny errors such as measurement errors in the initial conditions, neglected 'small' physical effects in the mathematical model, discretization errors, or even simple roundoff errors will be

This work was supported by the Natural Sciences and Engineering Research Council of Canada. Thanks go to G. Maxwell, who first introduced me to the theory of continued fractions, and also advised me to leave number theory alone and go into numerical analysis ('There are more jobs there'). I followed his advice, and found it good, but I am pleased to think that I managed to keep some contact with number theory. Thanks also to C. Moler for allowing his informal remarks on USENET to be moved to a more formal setting, and for modifying the code to use Matlab version 4. N. Madras pointed out the reference [8]. G. Labahn helped with the Maple code for the preparation of Figure 3. A. Stuart and W. Enright provided discussions at the 1993 Canadian Applied Mathematics Society meeting at York University in Toronto, which were very helpful in clarifying the rough anti-symmetry between chaotic and stiff problems.

Typeset by $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX

exponentially amplified as time progresses, and the details of the predicted motion quickly become uncorrelated with the actual motion.

We use the word 'ill-conditioned' for chaotic dynamical systems rather than 'unstable,' which we reserve for 'bad' numerical *methods* [9]. This usage is standard in numerical analysis, except in the initial-value problem community where ill-conditioned problems are also referred to as unstable, following the usage in the theory of differential equations. The reader should maintain awareness of the distinction between ill-conditioned problems and unstable methods in what follows, but should note that the words 'stable' and 'unstable' will also be used for equilibria or other *orbits* of dynamical systems. This should not cause confusion.

Anyone familiar with numerical instability would be initially skeptical of chaos known only through numerical solutions. The expert would suspect spurious, numerically introduced difficulties, and of course this does occur: see for example [10]. It is rather less expected to find that certain numerical methods can, on the contrary, *suppress* actual chaos [11]. The first part of [11] shows that the implicit Euler method can, for large enough step sizes, artificially stabilize truly unstable fixed-points and completely destroy any possible chaotic attractors.

See Figure 1, where we graph the computed Lyapunov exponent versus the stepsize: the larger h is, the less chaotic the problem appears to be. The Lyapunov exponent is computed with a standard method—see [11] for details.

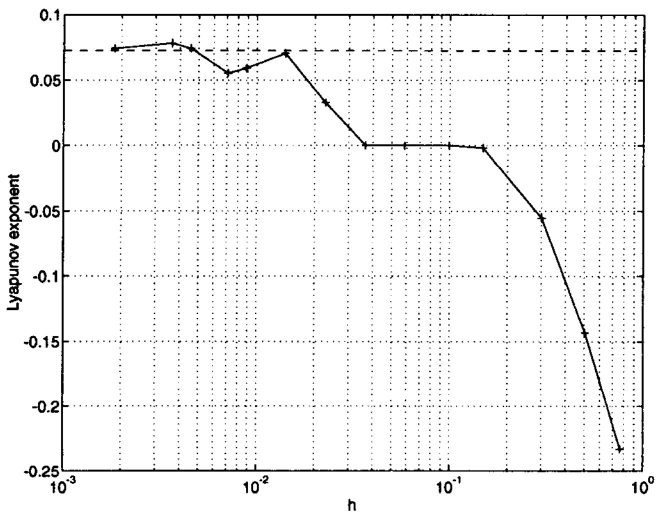

Figure 1. Implicit Euler can suppress chaos. The dashed line shows the accepted value of the largest Lyapunov exponent, from the literature.

On yet another level, that of floating-point arithmetic, we also see possibilities for deception. It was already observed in [12] that since the set of floating-point numbers is finite, then necessarily all computed orbits are ultimately periodic, and hence, not chaotic. Although one might expect this to be irrelevant in practice, we will see that this can be nontrivial.

Conversely, in the second part of [11], it was shown that floating-point simulation of a discrete map with a globally attracting fixed point at the origin can *appear* chaotic, for extremely long times, in either single, double, or extended precision, purely due to rounding error effects. See Figure 2, where we graph the Lyapunov exponent (computed approximately with an initial perturbation of various sizes δ) for the map

$$x_{n+1} = e^{-\eta} x_n + \varepsilon z(x_n),$$

where $z(x)$, mathematically identically zero, is computed by

$$z(x) = \begin{cases} 0, & \text{if } x = 0, \\ |x|^{-\beta}\left(x^{-1}\left((1+x)^2 - 1\right) - 2 - x\right), & \text{otherwise.} \end{cases}$$

The severe cancellation in this formula reveals the rounding errors made in the computation of $(1+x)^2$. The true Lyapunov exponent of this map is $-\eta < 0$, but is computed as being positive, for any δ. This roundoff-induced 'chaos' is totally spurious. Of course, this example is deliberately concocted to have bad numerical behaviour, but it is difficult to guarantee that such bad behaviour never occurs in any given program.

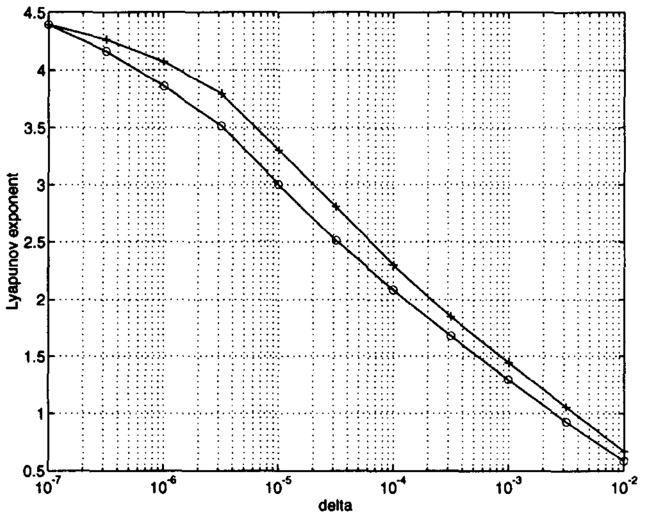

Figure 2. Roundoff error can introduce spurious chaos.

To summarize, there are four 'levels' of abstraction used here: the physical reality of the problem under study, the continuous mathematical model of that physical reality, the numerical discretization of that mathematical model, and the floating-point simulation of that discretization. Results from one level may or may not transfer easily to another level, and in particular, even qualitative features may not be preserved in that transfer. Moreover, there is no inherent bias either way: in any change of level we can introduce or destroy chaos.

Many problems do not require the second level, that of the continuous model, and one can go straight from the physical reality to the discrete model [13], but as we see, there is still the possibility of being fooled.

Paradoxically, however, physical experiments and numerical simulations of discretized mathematical models often seem to produce results in very good statistical or qualitative agreement—in the chaotic case—even though the details of the trajectories are completely different from the true ones after a very short time. This demands an explanation, and we would like to have a complete description of just how and when we can rely on numerical simulations of chaotic dynamical systems.

2. DEEPER EXAMINATION OF A SIMPLER PROBLEM: THE GAUSS MAP

To begin that explanation (to be sure, the full story is not known!), consider the following much simpler, discrete, dynamical system. It is almost unrelated to our motivating problem of flow-induced vibrations, but it falls in the general class of 'circle maps,' which *are* relevant to flow-induced vibrations [6]. If we define the Gauss map $G : [0,1) \to [0,1)$ as

$$G(x) = \begin{cases} 0, & \text{if } x = 0, \\ x^{-1} \bmod 1, & \text{otherwise,} \end{cases}$$

then this well-known map, from the theory of continued fractions, is chaotic [14,15].

This function is graphed on a torus in Figure 3, and a summary of its most important properties appears below.

1. The orbit $\{x_k\}$ (where $x_{k+1} = G(x_k), k = 0 \ldots \infty$) of every rational initial point x_0 goes to zero in a finite number of iterations. The rationals are dense in $[0, 1]$.
2. Orbits are ultimately periodic if and only if they start from *quadratic irrational* or, trivially, rational initial points. Quadratic irrationals are roots of quadratics with integer coefficients, and are dense in $[0, 1]$. Like the rationals, they are countable, and hence of measure zero. There are an infinite number of different orbits with each period.
3. The map is ergodic and so almost all initial points have orbits dense in $[0, 1]$.
4. The Lyapunov exponent of this map is, for almost all initial points, $\pi^2/(6 \log 2) = 2.3731\ldots$ but is *undefined* for rational initial points and is *different* for each quadratic irrational initial point.

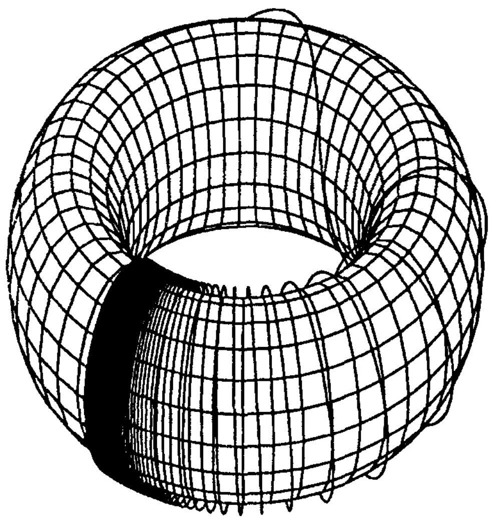

Figure 3. The Gauss map, graphed on a torus.

We thus see formidable numerical difficulties in simulating this map. Floating-point arithmetic is intended to represent arithmetic on a finite subset of the rational numbers, and we see from above that orbits starting at rational points are, in some sense, exceptional, and the 'actual' or almost-sure behaviour of the map is quite different. Yet, paradoxically, numerical simulation gives us, for example, a good approximation to the true almost-everywhere Lyapunov exponent when according to the skeptical view this should be impossible. For example, on an HP28S calculator, starting from $x_0 = 0.73$ and using 10^5 iterations of the floating-point Gauss map, we get a computed Lyapunov exponent equal to 2.36992, which is in error by less than 0.2%. This, in spite of the fact that the true Lyapunov exponent of the Gauss map starting at $x_0 = 0.73$ is not even *defined*—what saves the calculation here is roundoff error! For those who are interested, this leads to a candidate for "the world's worst" algorithm for computing π (see [14]).

The first steps of the resolution of this paradox are contained in [14,15]. Essentially, one can prove an explicit *shadowing* result: the numerically computed orbit starting from x_0 is uniformly close to the exact orbit of G starting at some other initial point y close to x_0. In fact, the proof constructs y explicitly, and shows that its orbit is everywhere within 4ε of the numerically computed orbit, where ε is the machine epsilon of the floating-point system being used for the simulation. Because of the special nature of this problem, this 'backward error' result is very strong, holding for infinite time with an explicit, reasonably tight bound on the shadowing distance, in terms of a known numerical characteristic of the floating-point system being used. The proof uses only the fact that division is accurate in a relative sense. A more typical shadowing result would hold only for a finite time, and the shadowing distance would be more like $\sqrt{\varepsilon}$.

This 'backward error' approach is very commonly used to justify computer simulation of chaotic maps. We will discuss later its application to chaotic differential equations. However, shadowing is not completely satisfactory: one sees immediately that numerical simulations of the Gauss map must inevitably give ultimately periodic orbits, from any initial condition, because the set of floating-point numbers in $[0,1)$ is finite. This means that the y whose actual orbit is shadowing the numerical simulation is a quadratic irrational or rational number, and thus is from a set of zero measure and in particular does not have a dense orbit or the correct Lyapunov exponent. The final resolution of this paradox must somehow account for the fact that the true shadowing orbit behaves like a typical orbit, even though it is not.

Detailed investigation of the logistic map by use of Chebyshev mixing transformations [16] showed that its numerical simulations typically have one very long-period orbit that most initial points are attracted to, possibly after a fairly long transient. A similar analysis for the floating-point Gauss map is sketched below, to see how we can regard the shadowing orbit as being typical.

Suppose for convenience that we use four-digit decimal arithmetic. Let F_4 be the set of numbers in $[0,1)$ having four digits after the decimal point, including leading zeros (e.g., 0.0012 is in this set but 0.001234 is not). There are 10^4 numbers in F_4. The numbers in F_4 are the only ones that matter to the four-digit simulation of the Gauss map, because numbers too close to 0, for example, cannot be distinguished from 10^{-4}, as inversion of anything smaller than this will produce a number larger than 10^4, which must have zero fractional part in four-digit decimal arithmetic. Thus, numbers too close to zero will be mapped to zero on the next iteration of G, exactly as 0.0001 will be.

It turns out that there are 14 distinct orbits of G restricted to F_4 by rounding to nearest. If we draw a directed graph on F_4 connecting each point to the one it goes to on iteration of G, we see the orbit structure in detail. It was convenient to use the `networks` package of Maple V, Release 2, for this purpose [17].

A 3-cycle and its basin of attraction are graphed in Figure 4, to give the flavour of a typical numerical orbit. One sees features common to floating-point simulations of many discrete dynamical systems. In particular, entry to the cycle seems to be through one 'dominant' floating-point number, and the inherent symmetry of the true basin of attraction of the true shadowing three-cycle is completely absent from the floating-point simulation. Of course, the true basin of attraction of the true three-cycle is of measure zero in $[0,1]$, whereas here there is a nonzero probability ($p = 0.0016$) of hitting this three-cycle. One expects that with larger sets of floating-point numbers, such short cycles will become more and more 'sparse' and thus less likely to be accidentally computed. The longest cycle for this example is of length 21, with 5.7% of the initial points tending to this cycle, and the longest transient is of length 123, tending to a three-cycle $0.33 \rightarrow 0.303 \rightarrow 0.003 \rightarrow 0.33$. More than 38% of initial points tend to this three-cycle, while nearly 33% of initial points tend to 0. For larger sets of floating-point numbers, one expects the basins of attraction of the longer cycles to dominate. Note that there are *only* cycles, and experiments show that their typical length increases as the precision increases.

This raises the question of just how long such cycles will be. Any map $x \rightarrow g(x)$ induces a numerical simulation $x \rightarrow \hat{g}(x)$ from the finite set of N floating-point numbers to itself. The average length of a cycle of a map chosen *at random* from the set of all N^N such maps is known to be $\sqrt{\pi N/8}+O(1)$ as $N \rightarrow \infty$, and the average length of any orbit is $\sqrt{\pi N/2}+O(1)$ [8, p. 519].

Of course, the map here is not really random—but in some sense just which numerical map is chosen is outside of our control and so it is reasonable to expect that statistical ideas will be useful. Indeed, the typical cycle lengths encountered for floating-point simulations of the logistic map in [16] and for the Gauss map restricted to F_4 agree with this expectation. For IEEE single

$$\begin{tikzcd}
0.8274 \arrow[r] & 0.2086 \arrow[r] & 0.7939 \arrow[r] & 0.2596 \arrow[r] & 0.8521 \arrow[dd] & \\
 & & & & & 0.7604 \arrow[dd] \\
 & & & 0.1394 \arrow[r] & 0.1736 \arrow[ru] & \\
 & & & & & 0.3151 \arrow[lu] \\
0.6084 \arrow[r] & 0.6437 \arrow[r] & 0.5535 \arrow[r] & 0.8067 \arrow[r] & 0.2396 \arrow[uu] & \\
 & & & 0.6913 \arrow[r] & 0.4465 \arrow[u] &
\end{tikzcd}$$

Figure 4. A 3-cycle and its basin under the four-digit floating-point Gauss map.

precision arithmetic, we can thus expect the typical cycle to have about 10^4 elements in it, and for double precision about 10^8. This is borne out by experiments with the Gauss map.

This 'square root' behaviour shows that the cycle shortness may really be more critical than one would expect, and represents a significant effect of floating-point arithmetic.

In [18,19], it is shown that this behaviour is quite general, and a typical scaling would be that the maximum period would usually be of length $O(N^{D/2})$ where D is the *correlation dimension* of the map under study.

Note that this behaviour is not necessarily true for simulations of other chaotic maps. For example, if we take the map $B : [0,1) \to [0,1)$ given by $B(x) = 2x \bmod 1$, then it would be simulated on IEEE machines with a simple shift of the binary digits, with 0's filling in on the right—thus orbits of the simulation would tend to zero for any initial point, and it would not be useful to model the induced numerical map with a randomly chosen map. In [20,21], these issues are taken up in more depth.

The square root effect, if present, can be handled by simple brute force, such as by going to double or extended precision: a typical cycle length of about 10^8 is likely long enough for most purposes. For the Gauss map, it seems intuitively clear that a very long transient or very long period orbit must really reflect the nearby presence of 'typical' points. One should be able to prove a theorem showing that if one defined an 'average' Lyapunov exponent for orbits of period T, then this average should tend, in the limit as T goes to infinity, to the Lyapunov exponent of almost all initial points. This might need a careful definition of 'average,' however, and currently I am not aware of any such proof. The very interesting paper [22], which is perhaps the only paper on the subject of the effects of floating-point arithmetic on simulations of chaotic dynamical systems to properly take account of the nonuniformity of the distribution of the floating-point numbers, gives a very general result along these lines. They show that if a map $\tau : [0,1] \to [0,1]$ has a computer implementation $\hat{\tau}$ which has 'very long trajectories,' $O(N)$ in our notation, then these trajectories have histograms which weakly approach the density of the (postulated) absolutely continuous invariant measure of τ. The conclusions of the main theorem of [22] are exactly what we want here: the absolutely continuous invariant measure for the Gauss map is the Gauss measure $\int_A dx/(1+x)$, and if the histogram of a long trajectory of a computer implementation of this map looks like the density of the Gauss measure then the computed Lyapunov exponent will be almost right.

Unfortunately, we do not know if there are orbits which are $O(N)$ long, and in fact by the previous arguments, we expect that the longest orbits are $O(\sqrt{N})$. This seems to be enough to invalidate one of the key steps in the proof of the theorem in [22]. Note, however, that their hypotheses give *sufficient* conditions and not *necessary* conditions—it would be interesting to see if the hypotheses can be weakened to allow use in this context.

For a more algorithmic discussion of the difficulties with shadowing, computable numbers, and 'almost-everywhere' versus actual initial points, see [23]. The essence of the discussion in that book, the work by Gavelek and Erber [16], and the previous paragraphs, however, is that shadowing by itself is not quite enough to allow us to believe in our numerical solutions—we must also have some confidence that the true orbit doing the shadowing is in some sense a typical orbit.

For continuous dynamical systems, as opposed to discrete dynamical systems, the idea of shadowing is theoretically the same. One simply appeals to the 'trivial' ability to calculate a locally accurate solution of a differential equation and then replaces the continuous dynamical system by a noisy discrete dynamical system. That is, in theory one can compute $y(t_k + T)$ to within a given tolerance δ, given the value for $y(t_k)$. Of course, there are great practical difficulties with this, and indeed interval arithmetic is required to actually compute $y(t_k + T)$ accurate to within δ. But waving these difficulties aside, we have reduced the problem from a continuous dynamical system $y' = f(y)$ to iteration of a discrete dynamical system $y(t_k) \to y(t_k + T)$, plus a *uniformly bounded* amount of noise. This is precisely what the shadowing idea from discrete dynamical systems is intended to cover, and again one can, for certain differential equations, show that there are initial points near to the specified one which give trajectories uniformly close to the numerical trajectory. It turns out that these computations are somewhat expensive, and as stated previously, require interval arithmetic because they are, in fact, computer-assisted proofs, and as such, must account for rounding error as well as discretization error. Further, there are difficulties with maps of dimension higher than 3 [24], and hence, with continuous dynamical systems of dimension higher than 3.

One can prove that a slightly weaker shadowing property than the 'pseudo-orbit tracing property' used above holds for *generic* dynamical systems, regardless of the dimension [25,26], and this provides some reassurance that dimension should not be a real barrier. This weaker property, which holds for all topologically transitive systems, explains for example the success of computer simulations of the Hénon map. Plots of the results of simulations of this map on different computers are essentially indistinguishable from each other, even though different roundoff properties of the different implementations of floating-point arithmetic ensure that the two pseudo-orbits diverge from each other (and from the true orbits) after only a very few iterations. The weak tracing property of [25] is enough to explain the similarity of the plots.

Shadowing itself allows some other reassurances, such as its use in [27] to show that upper Lyapunov exponents can be computed reliably for trajectories near hyperbolic sets, but the practical difficulties discussed above remain of interest.

3. DEFECT CONTROL FOR CONTINUOUS DYNAMICAL SYSTEMS

In some sense, the shadowing idea provides a 'post-processing' view of computation. One computes an orbit or trajectory and then tries to show by further computation (see e.g., [28]) or appeal to a theorem (though usually one which hasn't specified the required δ to get the desired ε) that a true orbit was within ε of the computed one. One then hopes that the shadowing orbit was 'typical' and proceeds from there.

There is a more efficient approach that uses a similar idea. Modern differential equation codes provide interpolants, for various practical reasons such as efficient graphical output, 'g-stops,' delays, etc., and since this is the case, it might pay us to think about the interpolated solution (as opposed to merely using it). One can substitute a continuously differentiable interpolant back into the differential equation, for example, and look at the *residual* or *defect* as it is called in the initial-value problem community:

$$\delta(x) = \frac{dx}{dt} - f(x) = \varepsilon v(t), \text{ say,}$$

where $x = x(t)$ is our computed, interpolated solution and ε is our input tolerance.

A trivial rearrangement of that equation gives us

$$\frac{dx}{dt} = f(x) + \varepsilon v(t),$$

and we see that if our error control mechanism can guarantee that $||v(t)|| \leq 1$, then *we have the exact solution to a nearby problem* [29]. For a fuller discussion of this approach for chaotic problems, see [30]. In view of physical perturbations (e.g., trucks passing by the laboratory where wind-tunnel experimenters work on flow-induced vibration problems), we can see that we have the *exact* solution of just as good a mathematical model of the physical problem as the one that was originally written down. Further, $v(t)$ is computable, and we can examine it at our leisure.

If there is some physically interesting *statistic* (that is to say, numerical quantity computed from the solution) that is insensitive to such perturbations, then we say the problem is 'well-enough conditioned.' If there is no such statistic, then the model is useless for any practical purpose, in view of physical perturbations. In [30], this concept is contrasted with the more usual ideas of 'well-posedness' and 'stability,' and an example of a problem that is not 'well-enough conditioned' is given.

Interval arithmetic is also necessary here [31], if we wish a guarantee that $||v(t)|| \leq 1$, as opposed to an estimate $||v(t)|| \approx 1$. Estimating codes such as PAMETH [32] are roughly as efficient as the best existing nonstiff codes using classical error control strategies, but are not intended for computer-aided proofs. Either approach is more efficient than shadowing, since the solution is constructed in the first place to be the exact solution of a nearby problem.

The technical idea of 'defect control' can be replaced by a simpler, albeit less efficient, point of view: that of merely examining the defect in a computed solution, however attained. We consider this simpler analysis here.

The following example was taken from a discussion on the USENET group sci.math.num-analysis, which took place in December, 1991. The spontaneous and informal discussion there will be paraphrased here. The discussions themselves are available by e-mail to `rcorless@uwo.ca`, but the context of the discussions has not been archived.

In a thread started by Björn Alsberg, who asked about numerical integration of the Lorenz equations in Matlab, C. Moler made the following observations, and gave a MATLAB program to graph the solution by Euler's method with fixed step-sizes.

> But I think a little discussion from a broader view point might be of interest to this group. The Lorenz attractor is of such great interest precisely because it is, in a sense, [an ill-conditioned] problem. Small changes in the data, including those introduced by numerical approximations, eventually lead to arbitrarily large changes in the solution. However, the altered solution has the same qualitative behavior as the exact solution. Any numerical solution is destined to be totally inaccurate, but its graph still looks the same.
>
> With this in mind, here is my MATLAB program to study Björn's instance of Lorenz's equation. It does not use ode23 or ode45 or any other modern, general purpose ode routine. It uses Euler's method with fixed step size! The result is lousy accuracy, but fascinating graphics and, I think, good insight into the nature of the Lorenz attractor [33].

These remarks are correct. The *trajectories* of the Lorenz system are, for certain parameter values, ill-conditioned, in that arbitrarily small changes in the initial conditions *or in the differential equations* will cause $O(1)$ changes in the trajectories of the solution after exponentially short times. That is, errors of size ε will be amplified to $\varepsilon \exp(\lambda T)$ after time T, where $\lambda > 0$ is the largest Lyapunov exponent.

The remarks are also correct in saying that the graph of the solution appears to the eye to be *unchanged* under perturbations of the initial conditions or under perturbations of the differential equations. We here examine the perturbation of the differential equation that occurs when we solve this system by Euler's method with fixed step-size.

For Euler's method, the natural interpolant is piecewise linear. This is not C^1. To get a C^1 interpolant, we can use cubic Hermite interpolation but this only adds complexity to the example. It turns out that the size (e.g., infinity norm) of the defect for any C^1 interpolant is not much different than the size of the (discontinuous) defect that arises from the piecewise linear interpolant. Therefore, we set

$$x(t) = x_n + (t - t_n)f(x_n) \qquad \text{for } t_n \le t < t_{n+1}.$$

We can sample the defect at t_{n+1}^- to get

$$\delta(t_{n+1}^-) = f(x_n) - f(x_{n+1}),$$

which gives us, asymptotically as $h \to 0$, the maximum value of the defect on $t_n \le t < t_{n+1}$. This sampling also has the advantage that we require no extra computations, for this simple method. For more complicated methods, say a higher-order Runge-Kutta method, extra stages are typically required to estimate the defect [29]. In the case of Adams PECE methods, a 'free' estimate is available from quantities already computed by modern codes [34]. For Taylor series methods, we can sample at t_{n+1}^- as we did above for Euler's method, and this is simple and effective. If we wish a more accurate estimate of the defect than we gain from just this one sample, we can, of course, sample it at more points, at a cost of one function evaluation per sample (plus some interpolation costs, but if the function is complicated then this will be trivial in comparison).

What follows is a modification of the originally posted Matlab code to plot the defect. This updated version corrects a bug and now uses Matlab version 4 to give a three-dimensional plot.

```
%
% The defect in Euler's method solution to the Lorenz equations.
% Matlab version 4.0
% Cleve Moler, 1993.
%
x = [.06735 1.8841 15.7734]';
h = .001;
figure(1)
clf reset
xp = plot3(0,0,0);
set(xp,'erasemode','none')
set(gca,'box','on')
axis([-10 10 -20 20 0 40])
view(-10,40)
figure(2)
clf reset
dp = plot3(0,0,0);
set(dp,'erasemode','none')
set(gca,'box','on')
axis([-.5 .5 -1 1 -.5 1.5])
view(-10,40)
kmax = 2000
imax = 10
A = [ -3 3 0; 26.5 -1 -x(1); 0 x(1) -1];
X = zeros(3,kmax);
```

```
D = zeros(3,kmax);
oldxdot = A*x;
k = 0;
while k < kmax
   if rem(k,200) == 0, k, end
   k = k + 1;
   for i = 1:imax
      A(2,3) = -x(1);
      A(3,2) = x(1);
      xdot = A*x;
      x = x + h*xdot;
      d = oldxdot - xdot;
      oldxdot = xdot;
   end
   set(xp,'xdata',x(1),'ydata',x(2),'zdata',x(3))
   set(dp,'xdata',d(1),'ydata',d(2),'zdata',d(3))
   drawnow
   X(:,k) = x;
   D(:,k) = d;
end
save
figure(1)
clf reset
plot3(X(1,1:k),X(2,1:k),X(3,1:k))
set(gca,'box','on')
axis([-10 10 -20 20 0 40])
view(-10,40)
print -deps defect1.eps
figure(2)
clf reset
plot3(D(1,1:k),D(2,1:k),D(3,1:k))
set(gca,'box','on')
axis([-.5 .5 -1 1 -.5 1.5])
view(-10,40)
print -deps defect2.eps
```

This Matlab script deserves some comment, as it uses some new features of Version 4.0. The command `figure` opens a new window. The command `figure(n)` makes the n^{th} window the current one. The handles `xp` and `dp` essentially point to the internal data structures for the two plots in the two windows. By referencing the handles, you can change many of the attributes of the plots. In this script, we are just changing the point in 3-space which is plotted. Since we have set `erasemode` to `none`, each new point is plotted, but the old ones are not erased—they just stay on the screen, although they are forgotten in the plot data structure. Since only a very small area of the plot has changed, it can be updated on the screen very quickly. `drawnow` triggers an X-event so the revised figure is actually drawn. All this allows us to see the plots during their computation. They have no effect on the final hardcopy.

We only plot and save every 10^{th} point. This is good enough resolution for the plots and, if we plotted every point, the resulting PostScript files would be 10 times as large. Finally, after the computation is complete, we redraw the graphs, this time with lines instead of dots, and produce hardcopy of this final result.

Figure 5 shows the Lorenz orbit itself and the defect is shown in Figure 6. We notice two things immediately:

(1) The defect is large—bigger than 1 occasionally—and Euler's method is adding a *large* perturbation to the Lorenz system. Decreasing h decreases the defect, asymptotically linearly. (This is because Euler's method is a first order method: the defect is asymptotically $O(h)$, and for a well-conditioned problem—which the Lorenz system is not—this would imply the global error was $O(h)$ as well.)

(2) The defect has structure. It is clearly correlated with the solution—we have a miniature but distorted version of the Lorenz mask produced, plus some other structure. This has some serious implications, as correlations can be e.g., "resonant." An explanation of this correlation is offered in [35].

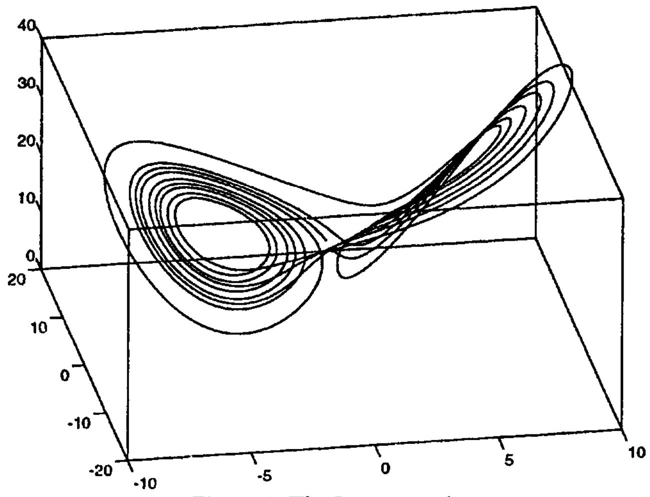

Figure 5. The Lorenz mask.

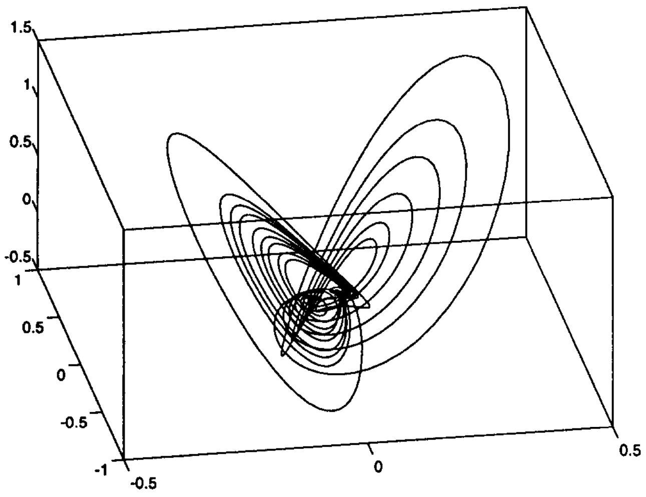

Figure 6. The defect in the solution of the Lorenz equations by Euler's method.

One wonders if a time-dependent perturbation to the Lorenz equations is physically reasonable. The Lorenz equations themselves are a severe truncation of a complicated fluid-dynamic model, and it is not clear what this time-dependent perturbation means in the original context. However, the Lorenz equations *do* model, for example, the behaviour of an analog computer set up to solve the Lorenz equations, or a laboratory water wheel [36], and in those contexts a time-dependent perturbation might make sense.

Are the Lorenz equations "really" chaotic? One can say that for finite times, some equations that are uniformly within 10^{-6} of the Lorenz system have positive Lyapunov exponents [30]. As an aside, it has been shown [37] by using rigorous, arbitrary-precision interval arithmetic calculations that homoclinic orbits actually exist in the solutions of the Lorenz equations themselves. Additionally, though I have not yet seen the works in question, there are apparently computer-assisted proofs that the Lorenz equations themselves are truly chaotic in the neighbourhood of certain parameter values [38]. One expects that these works have also used interval arithmetic.

More simply, though, it has been suggested [39] that the practical definition of chaos should be that a system is chaotic if enough sufficiently nearby systems have positive Lyapunov exponents. In view of physical perturbations, (e.g., the effect of Jupiter, or passing trucks, or whatever), this is only sensible. Thus one can certainly say that according to this pragmatic definition of chaos, the Lorenz equations are chaotic.

One might wish for a more precise definition of 'enough sufficiently nearby systems,' such as 'generic systems in arbitrarily small neighbourhoods' of the original dynamical system. It might be possible to prove such a statement, for a specific problem such as the Lorenz system. In practice, with the number of numerical simulations of the Lorenz system worldwide being probably in the hundreds of thousands, with tolerances as small as 10^{-13} or better, one can say here that a large amount of experimental evidence shows that there are lots of systems 'near' to the Lorenz system that have positive Lyapunov exponents. Hence, in a practical sense, the Lorenz system really is 'chaotic.'

As another example, in [40] it is proved that Euler's method applied to a central force problem introduces chaos (that is not present in the exact solution)—for any stepsize $h > 0$, using exact arithmetic. One could conclude from this result that the discretization is bad, in some sense. As argued here, one could conclude instead that the *problem* is bad—the central force problem is surrounded by chaotic problems, so it is possible that real-life perturbations will be exponentially amplified, not just numerical perturbations.

4. THE ISSUE OF CORRELATION AND HIGHER-DIMENSIONAL PERTURBATIONS

The remaining problem with shadowing—'is this a typical orbit?'—has a counterpart in defect control: is the computer-introduced perturbation physically reasonable? Certainly in some cases it will not be—the invariants or symmetries inherent in the physical problem may be varied or broken, giving nonphysical results. For example, in Hockett's problem above [40], we may instead be interested in the perturbations of the problem that arise from using symplectic methods instead of Euler's method, which give (nearly) exact solutions of nearby Hamiltonian problems. This relates to the idea of the appropriate norm to measure the size of the defect in.

Further, we denoted the defect by $\delta(t) = \varepsilon v(t)$ above, but in fact, the error-control mechanism adjusts the stepsize based on information about the solution itself, and we might be better off thinking about the defect as $\delta = \delta(t, x)$ or $\delta = \delta(t, x, \dot{x})$ or even $\delta = \delta(t, x, \dot{x}, \ddot{x})$, which last gives a singular perturbation of the original problem. Worse, if we are solving a partial differential equation by, e.g., the method of lines, we might be better off modelling δ as $\delta(t, x, \dot{x}, x_y)$, including a spatial derivative. This is pursued further in [35].

5. OBJECTIONS

Not everyone is happy with the idea of 'changing the problem.' Some are simply prejudiced—they are happy in the habit of thinking of forward error $x - x_{\text{true}}$ as the quantity they are interested in, and don't want to change. We see some of this resistance even in the area of polynomial roots or the solution of linear systems, where the idea of backward error analysis has been used successfully for more than thirty years.

Other, more rational people (such as writers of general purpose computer codes) have more valid objections: they write *general-purpose* software, and as such obviously *cannot* take into account the context of the problem being solved, which may not even exist at the time the code is being written. There, the model of trying for 'small global error' has proved very successful, for well-conditioned problems. It is unfortunate that the trajectories of chaotic problems are ill-conditioned, and in that case the goal of 'small global error' is unattainable without extra cost. See [28] for an example of how shadowing computations can help in this case, however.

An interesting objection to the backward error idea is that some problems are 'forward-stable' but not 'backward-stable' [41]—that is, for such problems, there exist approximations which give good forward error, but no acceptable ones which give good backward error. Stiff problems (and perhaps DAE's) may be viewed as falling into this category: it is possible to have a solution (computed with a stiffly-stable method) with a large defect in the usual norms, but small global error. In fact, this may be a useful definition of stiffness. If this definition is accepted, we see a certain rough anti-symmetry with chaotic problems: a chaotic problem is one where a solution with good backward error may be easily computed with explicit methods while a solution with good forward error is too expensive; on the other hand, a stiff problem is one where a solution with good forward error is easily computed using implicit methods, while a solution with good backward error is too expensive.

This anti-symmetry is only rough: it may be that there are problems which partake of both properties, having some very large and negative Lyapunov exponents and some small and positive ones: while the trajectory is moving towards an attractor, it might be that some degree of implicitness will be useful to efficiently solve the system, to locate and stay on the attractor—in which case one might call the problem stiff—but once on the attractor care must be taken to get good backward error, and this is likely to be an important restriction on the stepsize.

Further objections to defect control include the fact that we have added a time-dependent perturbation to a (perhaps) initially autonomous problem, and thus made it more complicated—putting the solution in a higher-dimensional space, in effect. This is indeed a complication, but one must remember that physical perturbations of autonomous problems can be time-dependent, and so this objection is not necessarily insuperable. Another objection is that the numerical solution usually merely goes on for a very long time—one wants an infinite time solution, which numerically is difficult unless one is near an equilibrium or limit cycle. But here again the use of infinite time is only a mathematical model of physical reality—one is usually only interested in infinite time as a 'first approximation' or simple model for 'very long times,' and so whether one simulates for a billion years instead of forever usually makes little practical difference. It is different, of course, if the infinite-time behaviour is easily available, through asymptotics or other approximations. Then, infinite time results provide useful approximations for the lengthy times we are usually interested in practically. See [42] for a discussion of available infinite-time backward error results for dynamical systems.

Finally, the choice of norm appropriate for the defect is clearly problem-dependent. This requires more complicated user interfaces, so the user can tell the computer code what norm to use, and concomitantly requires the user to be sophisticated enough to tell the code to use the right one. For differential-algebraic equations the norm will have to take into account the greater rigidity of algebraic constraints, for example, and raises the question of whether or not the chosen norm of the defect can be controlled at all by choosing the stepsize, order, or method. It is possible that an inappropriate underlying scheme might be detected by the noncontrollability of the defect, and this may open a fruitful line of investigation for numerical analysts.

6. CONCLUDING REMARKS

Numerical simulations of chaotic dynamical systems can be relied on when they are the exact solutions of problems 'sufficiently near' to the mathematical model of the physical problem under study, in the light of physical perturbations (whose effect has to be studied anyway). Ensuring that the problem really is 'sufficiently near' and deciding just what sense of 'near' is appropriate is not always easy, but at least this is now an applied mathematical modelling question and no longer one of study of floating-point arithmetic or discretization, though knowledge of each will obviously still be useful.

REFERENCES

1. G.V. Parkinson, Phenomena and modelling of flow-induced vibrations of bluff bodies, *Prog. Aerospace Sci* **26**, 169–224 (1989).
2. R.M. Corless, Bifurcation in a flow-induced vibration model, In *Proceedings of the Workshop on Normal Forms and Homoclinic Chaos*, (Edited by W.F. Langford and W. Nagata), AMS Fields Institute Communications, Waterloo, Ont., Canada, (1994).
3. R.M. Corless and G.V. Parkinson, A model of the combined effects of vortex-induced vibration and galloping, *Journal of Fluids and Structures* **2**, 203–220 (1988).
4. R.M. Corless and G.V. Parkinson, A model of the combined effects of vortex-induced vibration and galloping, Part II, *Journal of Fluids and Structures* **7**, 825–848 (1993).
5. C.W. Van Atta and M. Gharib, Ordered and chaotic vortex streets behind circular cylinders at low Reynolds numbers, *J. Fluid Mech.* **174**, 113–133 (1987).
6. D.J. Olinger and K.R. Sreenivasan, Universal dynamics in the wake of an oscillating cylinder at low Reynolds number, In *Proc. Int'l Symp. on Flow-Induced Vibration and Noise,* Chicago, Vol. 7, 1–30, (1988).
7. R.M. Corless, Chaos in a flow-induced vibration model, In *Proc. Int'l Symp. on Flow-Induced Vibration and Noise,* Chicago, Vol. 7, 77–86, (1988).
8. D.E. Knuth, *The Art of Computer Programming*, Vol. 2, 2nd ed., Addison-Wesley, Reading, MA, (1973).
9. P. Henrici, *Elements of Numerical Analysis*, Wiley, New York, (1964).
10. M. Yamaguti and S. Ushiki, Chaos in numerical analysis of ordinary differential equations, *Physica D* **3** (3), 618–626 (1981).
11. R.M. Corless, C. Essex and M.A.H. Nerenberg, Numerical methods can suppress chaos, *Phys. Lett. A* **157**, 27–36 (1991).
12. E.N. Lorenz, Deterministic nonperiodic flow, *J. Atmos. Sci.* **20**, 130–141 (1963).
13. D. Greenspan, Computer power and its impact on applied mathematics, In *Studies in Applied Mathematics,* (Edited by A.H. Taub), Vol. 7, pp. 65–89, Mathematical Association of America, (1971).
14. R.M. Corless, Continued fractions and chaos, *Amer. Math. Monthly* **99** (3), 203–215 (1992).
15. R.M. Corless, G.W. Frank and J.G. Monroe, Chaos and continued fractions, *Physica D* **46**, 241–253 (1990).
16. D. Gavelek and T. Erber, Shadowing and iterative interpolation for Čebyšev mixing transformations, *J. Comp. Phys.* **101**, 25–50 (1992).
17. B.W. Char, K.O. Geddes, G.H. Gonnet, B.L. Leong, M.B. Monagan and S.M. Watt, *The Maple V Language Reference Manual*, Springer-Verlag, New York, (1992).
18. C. Beck, Scaling behaviour of random maps, *Physics Letters A* **136** (3), 121–125 (March 1989).
19. C. Beck and G. Roepstorff, Effects of phase space discretization on the long-time behaviour of dynamical systems, *Physica* **25D**, 173–180 (1987).
20. P. Diamond, Iterated maps on discretized meshes of the unit interval, *Computers Math. Applic.* **21** (8), 57–63 (1991).
21. P. Diamond and P. Kloeden, Spatial discretization of mappings, *Computers Math. Applic.* **25** (6), 85–94 (1993).
22. P. Gora and A. Boyarsky, Why computers like Lebesgue measure, *Computers Math. Applic.* **16** (4), 321–329 (1988).
23. J.L. McCauley, *Chaos, Dynamics, and Fractals, an Algorithmic Approach to Deterministic Chaos*, Cambridge University Press, Cambridge, (1993).
24. C. Grebogi, personal communication.
25. R.M. Corless and S.Yu. Pilyugin, Approximate and real trajectories for generic dynamical systems, *J. Math. Anal. & Applic.* (1993) (to appear).
26. S.Yu. Pilyugin, The space of dynamical systems with the C^0 topology, *Springer-Verlag Lecture Notes in Mathematics*, No. 1571, Springer-Verlag, (1994).
27. R.M. Corless and S.Yu. Pilyugin, Evaluation of upper Lyapunov exponents on hyperbolic sets, *J. Math. Anal. & Applic.* (1993) (to appear).
28. S.-N. Chow and E.S. Van Vleck, A shadowing lemma approach to global error analysis for initial value ODEs, *SIAM J. Sci. Comp.* (to appear).
29. W.H. Enright, A new error-control for initial value solvers, *Appl. Math. Comput.* **31**, 288–301 (1989).

30. R.M. Corless, Defect-controlled numerical methods and shadowing for chaotic differential equations, *Physica D* **60**, 323–334 (1992).
31. R.M. Corless and G.F. Corliss, Rationale for guaranteed ODE defect control, In *Proceedings of SCAN 1991: International Symposium on Computer Arithmetic and Scientific Computing*, Oldenburg, Oct. 1–4, (Edited by J. Herzberger), IMACS Annals on Computing and Applied Mathematics, (1991).
32. W.H. Enright, PAMETH: Fortran subroutine for defect-controlled Runge-Kutta solution of ODE's, (private communication), (1990).
33. C. Moler, The Lorenz equations in Matlab, posting to sci.math.num-analysis, text available by e-mail to `rcorless@uwo.ca` (1991).
34. D.J. Higham, Defect estimation in Adams PECE codes, *SIAM J. Sci. Stat. Comput.* **10** (5), 964–976 (1989).
35. R.M. Corless, Error backward, In *Proc. Chaotic Numerics*, (Edited by P. Kloeden and K. Palmer), AMS, (1993) (to appear).
36. E.N. Lorenz, On the prevalence of aperiodicity in simple systems, In *Proc. Global Analysis, Calgary, Springer Lecture Notes in Mathematics*, Vol. 755, pp. 53–75, (1978).
37. B. Hassard and J.-H. Zhang, A homoclinic orbit of the Lorenz system by precise shooting, In *Proc. Fields Institute Workshop on Normal Forms and Homoclinic Chaos*, (Edited by W.F. Langford and W. Nagata), (1993) (to appear).
38. J.K. Hale, private communication, (1993).
39. W.H. Enright, private communication, (1990).
40. K. Hockett, Chaotic numerics from an integrable Hamiltonian system, *Proc. AMS* **108** (1), 271–281 (1990).
41. J. Demmel, private communication, (1991).
42. T. Eirola, Aspects of backward error analysis of numerical ODEs, *J. Comp. Appl. Maths.* **45**, 65–73 (1993).