Rigorously Shadowing Numerical ODE Integrations by Containment

Ph.D. Thesis Proposal of Wayne Hayes

January 27, 1999

This document summarizes the primary contributions of my Ph.D. thesis, and the tasks that remain to be accomplished. It is intentionally terse, without much background included or many references cited. Note: my Qualifying Depth Paper, in case you don't have a copy handy, can be browsed on the Web at "http://www.cs.utoronto.ca/~wayne/research/thesis/depth/". My M.Sc. thesis is nearby, at "...thesis/msc/". Finally, my Research Proposal is at "...thesis/RP.ps.gz".

Abstract

An exact trajectory of a dynamical system lying close to a numerical trajectory is called a *shadow*. We present a general-purpose algorithm for rigorously proving the existence of shadows of numerical ODE integrations of arbitrary dimension. Much of the rigor is provided automatically by interval arithmetic and validated ODE integration software that is freely available. The method is a generalization of a previously published *containment* process that was applicable only to two-dimensional maps. We extend it to handle maps of arbitrary dimension, and finally to ODEs. It involves the building of hypercubes around the discrete numerical trajectory through which the shadow is guaranteed to pass at appropriate times. The proof consists of two steps: first, the rigorous computational verification of a certain property; and second, a simple geometric argument showing that the computed property implies the existence of a shadow. The computational step is almost entirely automated, easily adaptable to any ODE problem, and asymptotically more efficient than previous rigorous methods since it requires a rigorous bound only on the error of the solution, rather than on its variational equation. The algorithm also allows rigorously for the "rescaling" of time, which is a necessary ingredient for successfully shadowing ODEs. Finally, the algorithm is "local", in the sense that it builds the shadow inductively, requiring information only from the most recent integration step, rather than information about "special" points arbitrarily far away on the numerical trajectory or bounds on resolvents along the entire trajectory. The resulting algorithm is capable of shadowing the Lorenz equations longer (to this author's knowledge) than any previously published results, with local errors of 10^{-6} commonly giving shadows of many times 10^4 time units; and with a local error of 10^{-13} commonly giving shadows many times 10^5 time units, and possibly much longer.[1]

1 Introduction

An exact trajectory of a dynamical system lying close to a numerical trajectory is called a *shadow*. Neglecting complications that will be discussed later, possibly the most general formulation of shadowing is one that states the shadowing problem as an optimization or generalized least squares problem: given an ODE

$$y'(t) = f(t, y) \tag{1}$$

and a discrete sequence of points $< (t_i, x_i) >_{i=0}^{N}$ representing an approximate solution, we want to find the solution, over the space of all exact solutions of equation (1), that is closest to the approximate solution in

[1] My network of workstations crashed due to a network failure when many systems had shadows that were between 3×10^5 and 4×10^5 time units long.

some norm:

$$\min_{\{y(t):y'=f(t,y),\quad i=0,\ldots,N\}} ||y(t_i) - x_i||. \tag{2}$$

If this distance is "small", for some reasonable problem-dependent definition of "small", then the resulting exact solution is called a shadow.

Although formulation (2) is succinct, it gives no hint as to how to solve it. Traditional optimization methods appear inadequate, so many methods specific to shadowing have been devised (*eg.,* Chow & Van Vleck 1994, Van Vleck 1995, Coomes, Koçak & Palmer 1994, 1995). The method in this paper relies on a very simple geometrical argument, which is a generalization of the *containment* process first introduced by Grebogi *et al.* (1990).

1.1 Informal Description of Containment

When containment was first introduced (Grebogi *et al.* 1990), it could be applied only to two-dimensional maps. Let

$$\phi : X \to X \tag{3}$$

be a two-dimensional map. We produce a *noisy* trajectory by iterating the map a finite number of times, including the effects of a small, problem- and machine-dependent perturbation function $\delta(\mathbf{p}_i)$:

$$\mathbf{p}_{i+1} = \phi(\mathbf{p}_i) + \delta(\mathbf{p}_i), \quad i = 0, \ldots, N-1, \tag{4}$$

We assume that ϕ is strongly *hyperbolic*, meaning that there are two special directions, called the *expanding* and *contracting* directions. As i increases, solutions separated from each other by a small distance along the expanding direction diverge on average away from each other, while solutions separated along the contracting direction approach each other, on average. The containment process consists of building a parallelogram M_i around each point $\mathbf{p}_i$ of the numerical trajectory such that two sides $C_i^{1,2}$ are separated from each other along the contracting direction, while the other two sides $E_i^{1,2}$ are separated along the expanding direction[2]. In order to prove the existence of a shadow, the image of M_i under $\phi(\cdot)$ must map over M_{i+1} such that $\phi(M_i)$ makes a "plus sign" with M_{i+1} (Figure 1). The property that Grebogi *et al.* define as a "plus sign" is

$$\phi(E_i^j) \cap M_{i+1} = \emptyset, \qquad \phi(M_i) \cap C_{i+1}^j = \emptyset. \tag{5}$$

To ensure this occurs, Grebogi *et al.* require a bound on the second derivative of ϕ, and the expansion and contraction amounts need to be resolvable by the machine precision. The proof of the existence of a true orbit then relies on the following argument. Draw a continuous curve γ_0 in M_0 between the expanding sides E_0^1 to E_0^2. Its image $\phi(\gamma_0)$ is then stretched such that there is a section of $\phi(\gamma_0)$ lying wholly within M_1, and in particular $\phi(\gamma_0)$ leaves M_1 through the expanding sides at both ends. Let γ_1 be the section of $\phi(\gamma_0)$ lying wholly within M_1. Now look at $\phi(\gamma_1)$ in M_2. Repeat this process along the orbit, producing γ_N lying wholly within the final parallelogram M_N. Then any point lying along γ_N, traced backwards, represents a

[2]Note this naming convention is exactly opposite to that of Grebogi *et al.* , and my Research Proposal. Grebogi *et al.* emphasized which direction the sides of M_i were *parallel* to. In higher dimensions, the faces of a hypercube aren't parallel to a unique direction, and it is the direction along which a face is separated from the centre of the hypercube that matters.

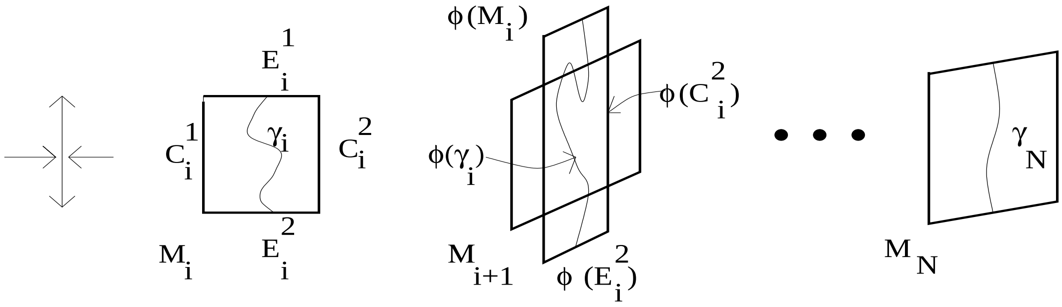

Figure 1: Containment in 2 dimensions, reproduced from Grebogi *et al.* The horizontal direction is contracting, and the vertical direction is expanding.

true trajectory that stays within M_i, $i = N, N-1, \ldots, 1, 0$, and we are done.

The process just described is not directly applicable to systems with more than 2 dimensions, and Grebogi *et al.* provided no argument for how it could be extended beyond 2 dimensions. It turns out, however, not to be too difficult. We describe the method in 3 dimensions, in which there are two cases:

(i) 1 expanding direction, and 2 contracting (Figure 2). Assume that the z direction is expanding, while the x and y directions are contracting. (We assume, for simplicity of exposition and for ease of drawing, that these three directions are roughly orthogonal, although in practice they need only be resolvable from each other.) Then, analogous to the 2 dimensional argument, we draw *cubes* M_i around the noisy points $\mathbf{p}_i$, and require that $\phi(M_i)$ maps over M_{i+1} so that ϕ stretches M_i into a long, thin tube, a length of which lies wholly in M_{i+1}. Then, precisely as in the 2-dimensional case, we introduce a curve

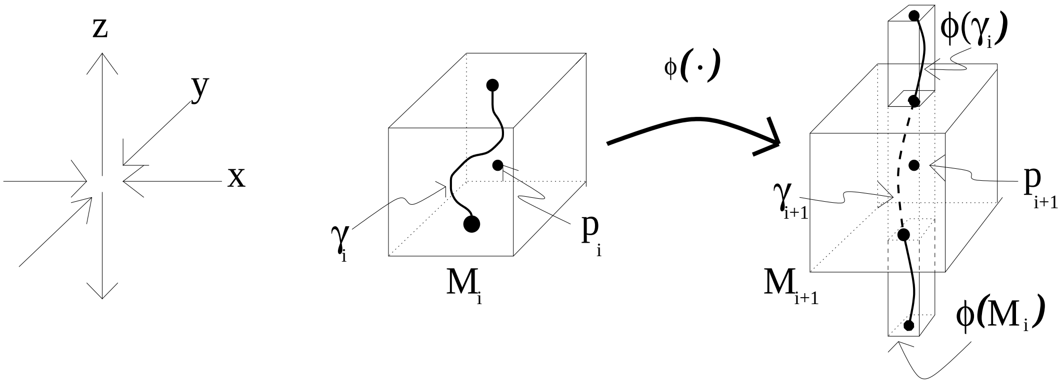

Figure 2: Containment in 3D, case (i): 1 expanding direction and 2 contracting.

γ_i that runs approximately along the expanding (vertical) direction from any point on the top of M_i to its bottom. If $\phi(M_i)$ maps over M_{i+1} as in Figure 2, then we are guaranteed that a contiguous section of $\phi(\gamma_i)$ lies inside M_{i+1}, connecting its top and bottom along the expanding direction. This becomes γ_{i+1}, and by induction γ_N lies inside M_N, and any point on it can be traced backwards to lie inside every M_i back to M_0, and we are done.

(ii) 2 expanding and 1 contracting direction (Figure 3). Assume now that the z (vertical) direction is contracting, while the x and y directions are expanding. We again draw a cube M_i around each noisy point $\mathbf{p}_i$, except now $\phi(M_i)$ maps over M_{i+1} so that ϕ flattens M_i into a thin slice, cutting M_{i+1} into 3 pieces, the middle piece of which contains a contiguous section of $\phi(M_i)$. Now, γ_i must be a *surface*,

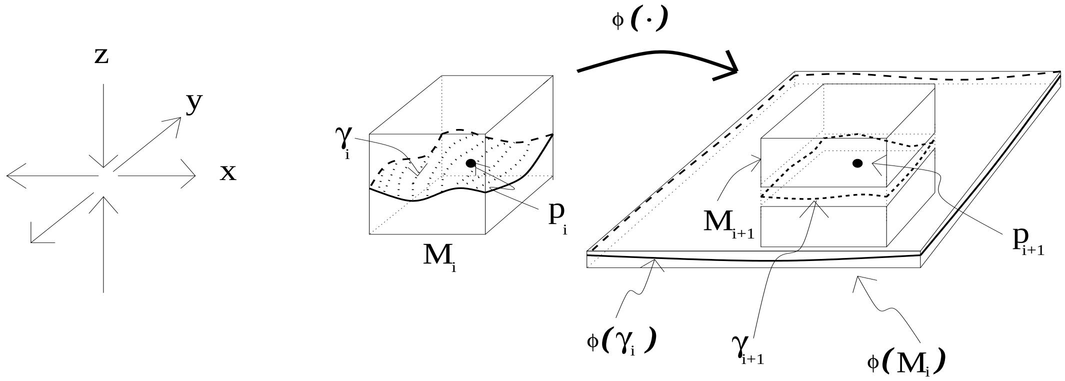

Figure 3: Containment in 3D, case (ii): 2 expanding directions and 1 contracting.

whose boundary connects all of the expanding sides, so that under the mapping, $\phi(\gamma_i)$ is stretched in all the directions it has extent (both horizontal directions), and is "compressed" along the direction it has measure zero (vertical). Then, we are guaranteed that there is a contiguous segment of $\phi(\gamma_i)$ lying wholly in M_{i+1} and connecting all of its expanding sides. We call this surface γ_{i+1}, and by induction γ_N lies wholly within M_N, and any point on γ_N, traced backwards, always lies within M_i all the way back to M_0.

It seems intuitively clear that we can replace "cube" with "hypercube", "surface" with "manifold", and the above argument still applies in arbitrarily high dimension. The crucial point is that γ must have dimension equal to the number of expanding directions, and its border must connect each side of the hypercube that contains a vector parallel to a contracting direction.

More formally, the property we want γ_i to satisfy is that for every co-ordinate in the expanding subspace inside M_i, there exists a co-ordinate in the contracting subspace inside M_i such that the union of the two co-ordinates gives a point on γ_i. In other words, γ_i *covers* the expanding subspace inside M_i.

1.2 Inductive Containment Property

In the following, i will be a fixed index (representing a timestep), and j will vary from 1 to d, d being the dimension of the space we're interested in.

Let $\{\mathbf{v}_i^j \in \Re^d,\ j = 1, \ldots, d\}$ be a set of d linearly independent vectors spanning $\Re^d$. Let V_i be the matrix whose columns are comprised of the above set.

Let $\bar{\mathbf{v}}_i^j = \mathbf{v}_i^j / ||\mathbf{v}_i^j||$. Let A_i be the matrix whose columns are these normalized vectors.

Let $\mathbf{p}_i \in \Re^d$.

Let M_i be a d-dimensional parallelapiped with centre $\mathbf{p}_i$ and edges parallel to $\bar{\mathbf{v}}_i^j$.

Let T_i^j, B_i^j be the faces of M_i ("top" and "bottom" in each dimension), defined by

$$T_i^j \quad = \quad \mathbf{p}_i + \mathbf{v}_i^j + \sum_{k \neq j} \alpha_k \mathbf{v}_i^k, \qquad \alpha_k = [-1, 1]\ \forall k, \tag{6}$$

$$B_i^j \quad = \quad \mathbf{p}_i - \mathbf{v}_i^j + \sum_{k \neq j} \alpha_k \mathbf{v}_i^k, \qquad \alpha_k = [-1, 1] \, \forall k. \tag{7}$$

Let $\phi : \Re^d \to \Re^d$, $\phi \in C^\infty$.

Let $E = \{1, \ldots, |E|\}$ be the set of (nominally) expanding directions, and $C = \{|E|+1, \ldots, d\}$ be the set of (nominally) contracting directions.

Let V_i^E be the set of vectors that are (nominally) expanding, and V_i^C be the set of vectors that is (nominally) contracting.

Let $E_i = \{T_i^k, B_i^k | k \in E\}$, the set of expanding faces.

Let $C_i = \{T_i^k, B_i^k | k \in C\}$, the set of contracting faces.

Let Q_i be M_i projected into the expanding subspace, *i.e.*, $Q_i = \mathrm{Span}\{V_i^E\} \cap M_i$,

Let R_i be M_i projected into the contracting subspace, *i.e.*, $R_i = \mathrm{Span}\{V_i^C\} \cap M_i$,

Definition: M_i and M_{i+1} satisfy the (d, E)-*Inductive Containment Property*, or the *(d,E)-ICP* if for all $j \in E$, $\phi(T_i^j)$ is "above" T_{i+1}^j and $\phi(B_i^j)$ is "below" B_{i+1}^j; and for all $j \in C$, T_{i+1}^j is "above" $\phi(T_i^j)$ and B_{i+1}^j is "below" $\phi(B_i^j)$. More formally,

$$\forall j \in E, \left[A_{i+1}^{-1}(\phi(T_i^j) - \mathbf{p}_{i+1})\right]_j \quad > \quad ||\mathbf{v}_{i+1}^j||, \tag{8}$$

$$\forall j \in E, \left[A_{i+1}^{-1}(\phi(B_i^j) - \mathbf{p}_{i+1})\right]_j \quad < \quad -||\mathbf{v}_{i+1}^j||, \tag{9}$$

$$\forall j \in C, \left[A_{i+1}^{-1}(T_{i+1}^j - \mathbf{p}_{i+1})\right]_j \quad > \quad ||\phi(\mathbf{p}_i + \mathbf{v}_i^j) - \phi(\mathbf{p}_i)||, \tag{10}$$

$$\forall j \in C, \left[A_{i+1}^{-1}(B_{i+1}^j - \mathbf{p}_{i+1})\right]_j \quad < \quad -||\phi(\mathbf{p}_i - \mathbf{v}_i^j) - \phi(\mathbf{p}_i)||. \tag{11}$$

To rigorously prove the existence of a shadow, we need to computationally verify the Inductive Containment Property between each pair of adjacent (M_i, M_{i+1}). Then, we need to prove that the ICP implies, for every step, what I'll call the γ-*covering property*:

$$\forall x \in Q_i \exists c \in R_i : \begin{pmatrix} x \\ c \end{pmatrix} \in \gamma_i. \tag{12}$$

This is the theorem that forms the heart of the thesis. I have not yet proven it, although I feel strongly that it is true, and I hope to prove it for the thesis. A bit more discussion of this theorem can be found in the Appendix.

1.3 Interval Arithmetic

In order to verify the Inductive Containment Property as described above using numerical methods, we use interval arthmetic and validated ODE integrations.

Interval arithmetic packages keep upper and lower bounds on floating point computations that are guaranteed to enclose the exact value of the computation. For example, let the symbol $+^\uparrow$ represent floating point addition in which the hardware rounding mode is set to "round up", and $+_\downarrow$ represent floating point addition with the hardware rounding mode set to "round down". To compute an interval that is guaranteed to enclose the exact sum of floating point numbers a and b, we compute

$$\underline{c} = a +_{\downarrow} b,$$

$$\overline{c} = a +^{\uparrow} b.$$

Then the interval $[\underline{c}, \overline{c}]$ is guaranteed to enclose the actual sum. Similarly, we can add two intervals $c = [\underline{c}, \overline{c}]$ and $d = [\underline{d}, \overline{d}]$ by performing

$$[\underline{e}, \overline{e}] = [\underline{c}, \overline{c}] + [\underline{d}, \overline{d}] = [\underline{c} \ +_{\downarrow} \ \underline{d}, \overline{c} \ +^{\uparrow} \ \overline{d}].$$

Similar operations can be provided for subtraction, multiplication, and division, although they are slightly more complex. Interval subroutines can also be provided for computing sin, cos, log, exp, and all the standard mathematical routines, by using interval arithmetic and some clever knowledge about, for example, bounding remainder terms in the series used to compute these functions. A "good" implementation of interval arithmetic is one in which the intervals are kept as small as possible while still enclosing the exact solution. A "bad" one lets the intervals expand too quickly, until the intervals are so large that virtually no information remains about the value of the exact solution.

Interval vectors and matrices can be represented simply by having arrays of interval real numbers. An interval vector may be thought of as an axis-aligned "box" that represents all the real numbers inside the box. An interval whose upper and lower bounds are equal is called a "point" interval.

Throughout this paper, the word "enclosure" will be used to describe the outer bound provided by interval methods, while the word "contain" and its derivatives will be used to describe the bounds provided by the shadowing containment algorithm.

1.4 Validated ODE Integrators

A validated ODE integrator is simply an algorithm that uses, among other things, interval arithmetic to produce an interval vector that is guaranteed to enclose the true solution of an ODE integration starting with a specific set of point or interval initial conditions. There are exactly two sources of error in numerical integrators: roundoff error, which can be accounted for by interval arithmetic, and truncation error, for which we need some theoretical bound on the error in the method. Most validated ODE integrators use Taylor series to bound the truncation error (Nedialkov *et al.* 1997). An impressive accomplishment in this direction has been the development of software that can automatically differentiate code lists (Bendtsen & Stauling, 1996, 1997), making it almost trivial to automatically generate Taylor series.

If the interval vector representing an enclosure of a validated integration is an axis-aligned box at a certain time, its image under the evolution of the ODE is unlikely to remain axis-aligned for very long. Many methods have been devised to account for this (called the "wrapping effect"). We use the software developed by Nedialkov (1999), and inherit his method of representing the enclosure of a solution as

$$\{y_{i+1}\} = \hat{y}_{i+1} + A_{i+1}[r_{i+1}] \supseteq \phi_h(\{y_i\}), \tag{13}$$

where ϕ_h is the solution operator for one timestep of length h, $\{y_i\}$ is the set of points enclosing the image of the previous solution enclosure, $\hat{y}_i$ is a point vector, A_i is a point matrix, and $[r_i]$ is an interval vector enclosing the zero vector. The matrix A_i is usually evolved via the variational equation of the ODE, so that it can keep a tight enclosure around the image of previous enclosures. In our implementation, $[r_i]$ is evolved using (Nedialkov *et al.* 1997)

$$[r_{i+1}] = A_{i+1}^{-1}([S_i]A_i)[r_i] \quad + \quad A_{i+1}^{-1}([z_{i+1}] - z_{i+1}). \tag{14}$$

The first term is responsible for propagating the error from the previous step, while the second term is the new error introduced at the current step, with $[z_i]$ being derived from the remainder term of the Taylor expansion for step i, while z_i is the midpoint of the interval $[z_i]$. It turns out that A_i can be *any* matrix, although some choices are better than others. With any choice of A_i, $[r_i]$ will still be a proper enclosure if it is computed with equation (14). In our implementation, A_{i+1} is the orthogonalization of the image of A_i under the variational equation.[3]

A more thorough discussion of interval arithmetic and validated ODE integration will be included in the thesis.

2 Implementation of Containment

2.1 Description

The central idea of containment is that we need to build a paralellapiped M_{i+1} that fits over the image of M_i in such a way that the image of M_i is smaller than M_{i+1} in some directions, and larger than M_{i+1} in other directions. The former we term the *nominally contracting directions*, while the latter are the *nominally expanding directions*. Recall that a validated ODE integration gives an enclosure that provides only an *outer* bound on the image of the previous enclosure. If the only information we have access to is the enclosure, then we can use it only to provide proof of contraction. Refer to figure 4. In both figures 4.a and 4.b, the

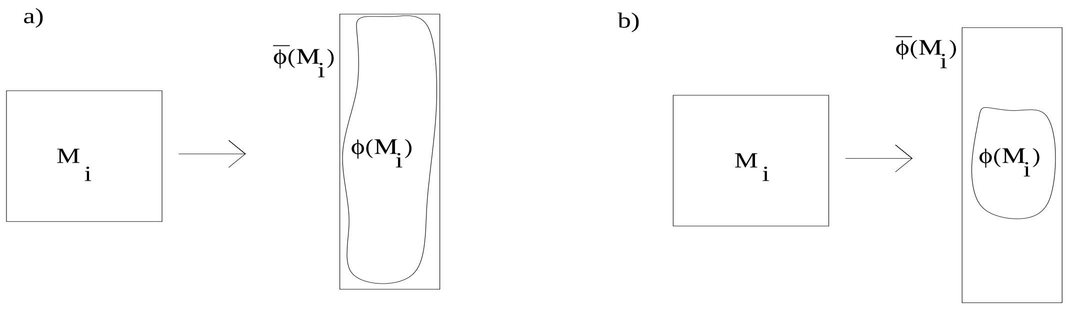

Figure 4: Enclosure methods prove contraction, but not expansion.

rectangle $\bar{\phi}(M_i)$ is a valid enclosure, because it is a superset of $\phi(M_i)$. In both figures, $\bar{\phi}(M_i)$ can be used to prove that $\phi(M_i)$ has contracted in the horizontal direction. However, enclosure methods cannot directly prove expansion, as figure 4.b demonstrates: although $\bar{\phi}(M_i)$ is a valid enclosure of $\phi(M_i)$, it is not a very good one, because the actual image of M_i has not expanded in any direction.

[3] Note that if this orthogonalization didn't take place, then $A_{i+1}^{-1}([S_i]A_i)$ would simply be the identity, to within roundoff error.

To solve this problem, we perform two validated integrations. Refer to figure 5.a. The first integration

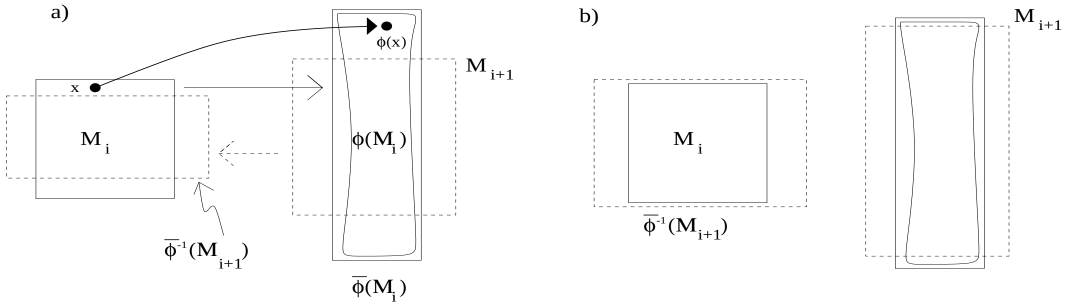

Figure 5: (a). The two validated integrations required to prove containment. (b) Shortcomings of the two-integration method: sometimes it can not prove expansion even if the M_{i+1} is valid.

(solid rectangles) is a forward integration that is used to determine a size that is larger than $\phi(M_i)$ for the sides of M_{i+1} in the nominally contracting directions (the horizontal direction in the figure). The second integration (dashed rectangles) is backwards, and proves backwards contraction in the nominally expanding directions (vertical direction in the figure). We then argue that expansion in the forward direction has occurred, as follows: choose any $x \in M_i - \bar{\phi}^{-1}(M_{i+1})$. Its image is in $\phi(M_i)$, but not in M_{i+1} since x is outside $\bar{\phi}^{-1}(M_{i+1})$, which in turn is an outside bound of $\phi^{-1}(M_{i+1})$. So, $\phi(x) \in \phi(M_i) - M_{i+1}$. Since this is true of any $x \in M_i - \bar{\phi}^{-1}(M_{i+1})$, this proves expansion in the forward direction. Figure 5.b demonstrates that it is possible to choose an M_{i+1} that is not small enough in the vertical direction for the validated integrator to prove contraction in the backwards direction, even though the M_{i+1} does, in fact, satisfy containment. We solve this problem in the algorithm by iteratively shrinking M_{i+1} in the nominally expanding directions until its image under ϕ^{-1} fits inside M_i in those directions.

If the system were uniformly hyperbolic, then the nominally expanding directions would always expand, and the nominally contracting directions would always contract. However, in strongly (but not uniformly) hyperbolic systems, the nominally expanding directions usually expand, but not always; and similarly for the contracting directions. One of the reasons our shadowing algorithm can fail is if a nominally expanding direction contracts too much or for too long a time (figure 6). Then, the expanding direction of M_i becomes

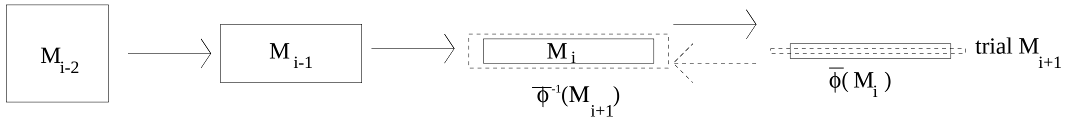

Figure 6: Example of the nominally expanding direction contracting too much for our integrator to prove contraction in the backwards direction.

so small that no backwards integration from M_{i+1} can fit inside M_i in the nominally expanding directions.

There may be other ways to implement this general idea, but with better results. For example, it may be possible to prove both expansion and contraction using a single forward integration if we take advantage of knowledge of the second term in equation (14), which essentially tells us how much uncertainty is introduced to the boundary of the image of $[r_i]$ as a result of new error introduced on this step, both inwards and outwards. Alternatively, there are other implementations of validated ODE integration that we could use that are more expensive, but provide tighter bounds on the solution. We have not yet explored any of these options.

2.2 Implementation in interval arithmetic

As one may have guessed by now, the representation of M_i is $\{y_i\}$ as defined in equation (13).

To computationally verify contraction in the forward direction, it is necessary only to give M_{i+1} the *same* size as $\bar{\phi}(M_i)$ in the nominally contracting directions, since $\bar{\phi}(M_i)$ is already outside $\phi(M_i)$ in those directions (although it need not be strictly outside). Proving expansion is slightly more subtle, because the only operation we are allowed to use in comparing intervals is to ask if one is a subset of another, whereas clearly the containment property we are attempting to prove is not subset enclosure in the full space of the problem. However, it *is* subset enclosure if we restrict ourselves to the expanding directions. Thus, to prove expansion, assume that we have correctly chosen an M_{i+1} that is small enough in the expanding directions (*eg.,* like figure 5.a, not figure 5.b). Then, we integrate M_{i+1} backwards, giving $\bar{\phi}^{-1}(M_{i+1})$ (figure 7). Next, we "collapse" this value down to zero width in the contracting directions, and finally we can ask if this collapsed interval $\bar{\phi}_C^{-1}(M_{i+1})$ is a subset of M_i.

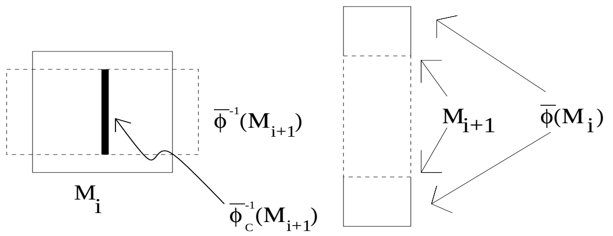

Figure 7: "Collapsing" intervals in the contracting direction to verify expansion.

2.3 Complications

There are several complications that require us to modify the M_i that we choose to verify containment. In particular, whenever we discover a new source of numerical error, we must bound it, and then increase the minimum distance between the faces of $\phi(M_i)$ and the corresponding face of M_{i+1}. Increasing this distance ensures that the Inductive Containment Property (equations (8–11)) is maintained. For the duration of this section, we will call this distance $\delta(M_i, M_{i+1})$.

2.3.1 Verifying directions and their images

The first complication is that the validated ODE integrator being used (Nedialkov 1999) only guarantees that the set $\{y_{i+1}\}$ (see equation (13)) encloses the image of the previous set $\{y_i\}$; it does not make any guarantees about which corners of $[r_{i+1}]$ correspond to which corners of $[r_i]$. This makes it impossible to guarantee inequalities (8–11) without first verifying that each face of $\phi(M_i)$ is placed in the correct position with respect to its corresponding face in M_{i+1}. (Of course, since A_i is evolved via the variational equation, we *do* expect the faces to behave in an orderly manner, but for the algorithm to be rigorous, we need some way to verify it.) Fortunately, it is not hard to verify. Recalling equation (14), we note that the second term is designed to correctly account for Tayler series truncation error for all points in $[r_i]$ and $[r_{i+1}]$. In particular, we can verify it for the centres of the faces defined in equations (6–7). Let c be the centre of a face of M_i; *i.e.*, a point vector in $[r_i]$ such that all the components but one have been set to zero, and the remaining component has its interval top and bottom both set to one or the other of the top or the bottom. All we want to do is verify what we already strongly suspect about what the validated integrator gives us: namely, that the centre of a particular face of $[r_{i+1}]$ is very close to the image of the centre of the same face of $[r_i]$. To do this, we compute the image of c using interval arithmetic,

$$[c'] = A_{i+1}^{-1}([S_i]A_i)c \ + \ A_{i+1}^{-1}([z_{i+1}] - z_{i+1}), \qquad c \in [r_i],$$

and verify that the centre of the corresponding face in $[r_{i+1}]$ is enclosed by $[c']$. Finally, we need to increase $\delta(M_i, M_{i+1})$ by the width of the interval $[c']$. (This step is performed before the orthogonalization of A_i.)

2.3.2 Orthogonalization of A_i

As mentioned previously, the matrix A_{i+1} used in equation (14) is not directly the image through the variational equation of A_i. Instead, it is an orthonormalized version of this image. This orthonormalization is computed with the Gram-Schmidt algorithm. Recall that the Gram-Schmidt process involves taking a set of vectors $\{\mathbf{v}^j\}_{j=1}^d$, and for each $\mathbf{v}^j$, subtracting away the components of $\mathbf{v}^j$ that lie in the space spanned by $\{\mathbf{v}^k\}_{k=1}^{j-1}$, giving $\mathbf{v}'^j$. This means that at each step, the new set of vectors $\{\mathbf{v}'^k\}_{k=1}^{j-1}$ spans the same subspace as the old set $\{\mathbf{v}^k\}_{k=1}^{j-1}$. Since we only take the contracting set when verifying containment, we are guaranteed that contraction has occurred in the subspace we are concerned with, even though the Gram-Schmidt algorithm has changed the directions of the edges, and thus the shape, of M_{i+1}. This is illustrated in two dimensions in figure 8. We see that in both parts (b) and (c) of the figure, we can build an M_{i+1} that is inside the corners of $\phi(M_i)$ in the expanding directions, and outside the corners of $\phi(M_i)$ in the contracting directions, regardless of how the Gram-Schmidt process chooses to build an orthogonal co-ordinate system.

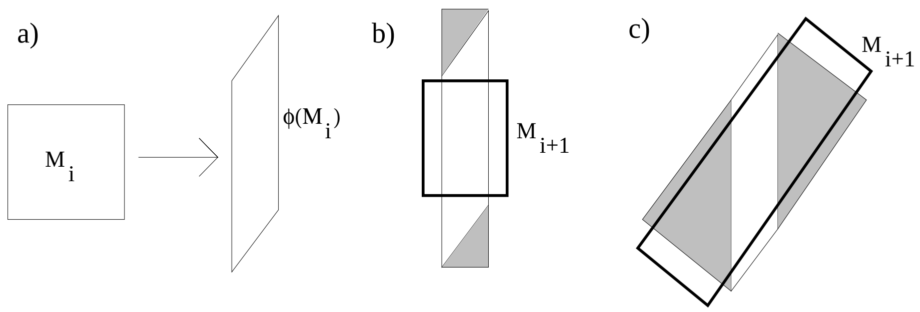

Figure 8: (a) M_i and its image $\phi(M_i)$. The nominal expanding direction is the vertical direction. (b) The Gram-Schmidt process fixes the vertical (expanding) direction and builds a second direction orthogonal to it. The result is an orthogonal box consisting of both the shaded and unshaded regions. Finally, we build an M_{i+1} that satisfies containment. (c) The Gram-Schmidt process fixes the bottom-left to top-right diagonal (contracting) direction and builds a new direction orthogonal to it, top-left to bottom-right diagonal. Again, we build an M_{i+1} that satisfies precisely the same kind of containment as (b).

2.4 Rescaling time

Consider the one-dimensional ODE

$$y''(t) = 0.$$

In physical terms, this represents a particle in straight-line motion with a constant velocity, say v_0. Now, assume we have a noisy trajectory of this system:

$$v(t) = \begin{cases} v_0, & t < 0. \\ v_0 + \delta, & t \geq 0. \end{cases}$$

Clearly, this noisy trajectory has no shadow that lasts any significant amount of time, because any true trajectory has a constant velocity, and will diverge linearly in time away from the noisy trajectory in at least one of the intervals $t < 0$ or $t > 0$.

Our description of containment thus far has ignored the numerical errors in time that occur in numerical ODE solutions, and thus is more applicable to maps than to ODEs. There is a fundamental difference between a discrete map and a discrete solution to an ODE. Local errors of the former are restricted to being "space like"—there is no concept of the passage of time between iterations of the map. The latter, however, can have errors in space as well as time. That is, the numerical error in the length of each timestep can accumulate, leading the numerical solution to have a slightly different time scale than the real system. In other words, we want to allow the numerical solution to be shadowed if it closely follows the *path* of a true solution, but at time t it is allowed to be up to ϵt time units ahead of, or behind, the true solution. This linear growth of errors is due to a lack of hyperbolicity in the direction of the flow in phase space. For large t, this can be a significant difference, so a shadowing algorithm which does not take the re-scaling of time into account is likely to grossly underestimate the length of the shadow. Coomes, Koçak, and Palmer (1994,1995) dramatically demonstrate this when they show that their "map method" is capable of shadowing the Lorenz equations for only 10^1 time units, while their "transversal" method which rescales time can find shadows lasting 10^5 time units.

In the containment context, a rescaling of time can be described naturally as follows: in containment as described so far, we have a box M_i through which a shadow passes. Part of the extent of M_i, of course, lies in the direction of motion. It is easiest if we assume that one of the directions $\mathbf{v}_i^j$ of M_i lies along $y'(t_i)$; this can be easily arranged, as described below. If we can prove that the shadow passes through the hyperplane H_i perpendicular to the direction of motion at the centre of M_i some time near to the time computed for the validated solution for M_i, then we can collapse M_i to H_i, *i.e.,* to size zero in the direction of motion. See figure 9. The sole complication to this is that it is possible that the shadow might be in a far corner of

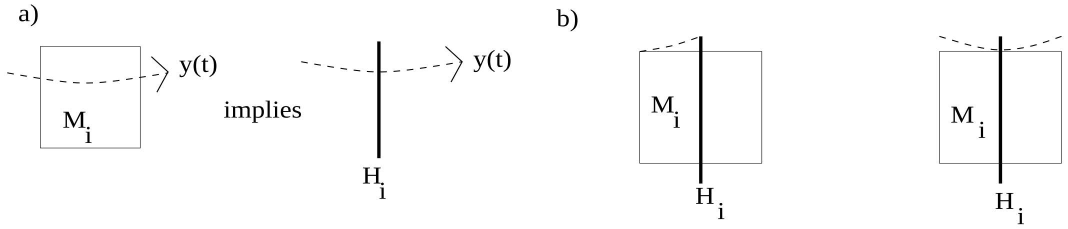

Figure 9: Collapsing M_i in the direction of motion to facilitate the rescaling of time: (a) A shadow passing through M_i will pass through the hyperplane H_i at the centre of M_i that is perpendicular to the direction of motion…(b) except that we need to account for a shadow that is at the edge of the box, that leaves the box before getting to H_i.

M_i, and may leave M_i before it gets to H_i, as shown in figure 9.b. To account for this, we need a bound

B on $y''(t)$ inside M_i (which we can get automatically using TADIFF or FADBAD—Bendtsen and Stauling 1996, 1997), and a bound on the amount of time Δt it takes $y(t)$ to cross M_i. Then, we need to make H_i bigger than M_i in the directions perpendicular to $y'(t_i)$ by an amount $B(\Delta t)^2/2$.

Finally, the easiest way to implement this is to keep one of the $\mathbf{v}_i^j$ of M_i, say $\mathbf{v}_i^1$, very closely parallel to $y'(t_i)$. Since its vector will be the first column of A_i, its direction will not be affected by the Gram-Schmidt process, and thus it will be propagated very closely to $y'(t_{i+1})$. To account for the slow buildup of error that would allow $\mathbf{v}_i^1$ to drift away from $y'(t_i)$, we reset $\mathbf{v}_i^1$ to be parallel to the computed $y'(t_i)$ at each timestep. This corresponds to rotating M_i by a small angle θ, computed by solving $\cos(\theta) = \mathbf{v}_i^1 \cdot y'(t_i)$, where $\mathbf{v}_i^1$ is the vector computed via evolution of the ODE from the previous timestep, and $y'(t_i)$ is the value of y' computed directly from the right hand side of the ODE at the current timestep. The furthest distance a point in M_i will move as a result of this rotation is $r\theta$, where r is distance of the furthest corner in M_i from the centre of M_i. Thus, we increase $\delta(M_i, M_{i+1})$ by $r\theta$.

This concludes the description of containment as we have implemented it.

3 Preliminary Results

The only ODE for which I have substantial results is the Lorenz equations,

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} = \begin{pmatrix} \sigma(y-x) \\ \rho x - y - xz \\ xy - \beta z \end{pmatrix} \tag{15}$$

There are several other examples in the papers listed in the bibliography, and since it is so easy to change the ODE in our software, I intend to attempt shadowing them all, to have a good sample of comparisons to previous results.

I am aware of only two papers that attempt shadowing of the Lorenz equations using a rescaling of time: Coomes, Koçak & Palmer (1995), and Van Vleck (1995).

With no rescaling of time, Van Vleck gives 2 examples of shadows with local error of about 10^{-5} lasting 1.04 and 1.38 time units; Coomes *et al.* have 6 examples with local error of about 10^{-13} lasting 9.7, 9.8, 9.9, 9.9, 86, and 126 time units. I have simulated hundreds of shadows with various local errors. With local errors of about 10^{-5}, my shadows last between 1 and 30 time units, with a median and mean of about 20. With local errors of 10^{-13}, I have shadows lasting between 10 and 1000 time units, again with a mean and median about halfway through that range. Thus, it appears that, without a rescaling of time, the containment method is capable of finding shadows that are about an order of magnitude longer than other methods.

With a rescaling of time, Van Vleck gives many examples of shadows (again with a local error of about 10^{-5}) ranging from 10^2 to 10^4 time units; Coomes *et al.* (with a local error of 10^{-13}) give 6 examples of shadows lasting 10^5 time units. The corresponding numbers for the containment algorithm are 10^2 to 10^5 for local errors of 10^{-5}, and 10^2 to at least several times 10^5 for local errors of 10^{-13}. The latter numbers are lower bounds because there was a network crash that halted about 30 of my shadowing runs when all had shadows with lengths in the range 3–4$\times 10^5$. These results are summarized in the following table.

Author, local error	Map Method	Rescale Time
VV, 10^{-5}	1–2	$10^2 \sim 10^4$
Hayes, 10^{-5}	$10 \sim 30$	$10^2 \sim 10^5$
CKP, 10^{-13}	$10 \sim 100$	$\geq 10^5$
Hayes, 10^{-13}	$10 \sim 1000$	$\geq$ 3–4$\times 10^5$

Figure 10 shows two results of shadow lengths, including the rescaling of time. The first is for 81 orbits with local error of approximately 10^{-6}, and the second for 81 orbits with local error of approximately 10^{-13}. The sharp increase in shadow lengths occuring just left of centre in the first figure is probably due to the fact

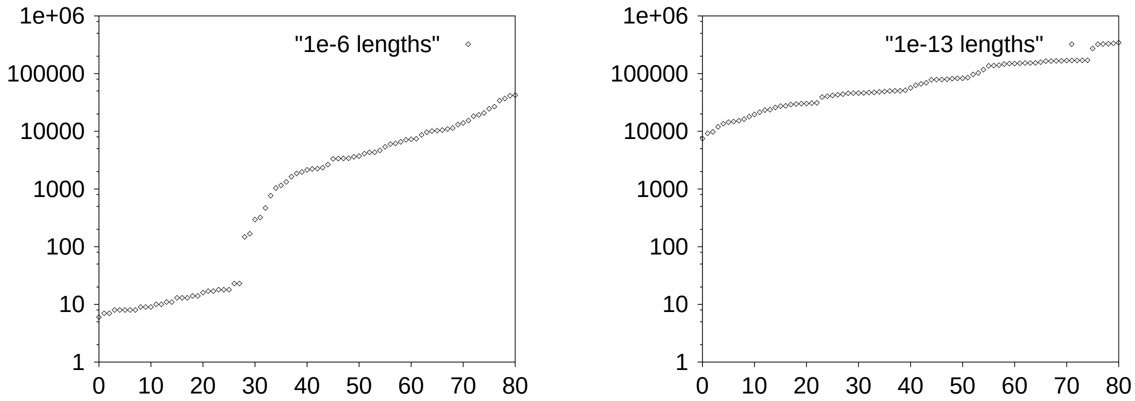

Figure 10: Distribution of shadow lengths. Each figure shows a sorted list of shadow lengths for 81 simulations of the Lorenz equations. The horizontal axis is simply a label for each shadow; the vertical axis is its length. The magnitude of the noise in the noisy orbits is about 10^{-6} in the left graph, and 10^{-13} in the right.

that, other than choosing one of the directions of M_0's edges to be parallel to $y'(0)$, the directions of the sides of M_0 are currently chosen at random. This means that, much of the time, we choose nominally expanding and contracting directions that are not sufficiently close to the true expanding and contracting directions. Thus, many shadows fail early on due to this problem. However, if our nominally chosen directions are (by luck) close enough to the true ones, then we get over this hump to get much longer shadows. There is probably a smarter way to choose the initial M_0, but I'm not sure what it is.

In addition, my shadowing distances (*i.e.,* the maximum distance between the shadow and the numerical trajectory) are comparable to the above authors: for orbits with noise 10^{-5} and 10^{-13}, both the above authors and myself find shadowing distances of approximately 10^{-4} and 10^{-8}, respectively.

4 Discussion

The containment method has several advantages over other rigorous shadowing methods. First, it uses off-the-shelf software for interval arithmetic, validated ODE integration, and automatic differentiation. This software alleviates much of the tedium and manual labour that, it appears to me, would be involved in adapting previous shadowing algorithms to new ODEs. For example, until recently, the only ODE I had tried to shadow was the N-body problem. It took only a few hours of work to modify the code to shadow the Lorenz equations, and much of that work will not need to be repeated when I try other equations. Eventually, I see no reason why changing the ODE to shadow will be as easy as changing the ODE to integrate in any off-the-shelf integrator. One consequence of this ease of use is that it requires the practitioner to have no deeper a knowledge of the system than the defining ODE.[4]

A disadvantage of this method is that, as it is currently implemented, it is not capable of taking an *a priori* noisy solution, and trying to find a shadow of that noisy solution. (Or, in other words, I'm not currently capable of doing *a posteriori* shadowing of noisy solutions, like I did in my Master's Thesis.) It is currently intimately intertwined with Nedialkov's validated ODE integrator, and as such the only noisy solutions it can start with are those produced by Nedialkov's Taylor series integrator. I believe it should not be too difficult to adapt the method to use an externally-generated noisy orbit, although I have not carefully considered this problem yet.

On the other hand, one could consider this implemenation as a generalization of validated ODE integration. Traditional validated ODE integration involves finding a bound enclosing *the* true solution that starts at $t = 0$ at the exact point initial condition given. Since numerical solutions are not exact any time after time zero, it seems rather artificial to insist that they be exact *at* time zero. Considering that shadows can last many orders of magnitude longer than validated ODE integrations (for example, Nedialkov's integrator typically finds a validated solution of the Lorenz equations for about 25 time units), perhaps it is better to say that numerical trajectories are valid approximations of true trajectories as long as one drops the requirement that they be exactly valid at time 0, and treats time zero on an equal footing with every other time (Murdock 1995).

The containment algorithm is also more efficient than other algorithms, because it requires a rigorous bound only on the solution, rather than on its variational equation. This may have a tremendous impact on efficiency for large systems.[5] If I were to compute rigorous bounds on the variational equation, I would not be restricted to explicitly building the M_i boxes. Instead, I could implicitly build boxes, and keep track of their width explicitly. This would alleviate the problem depicted in figure 6, because instead of using an interval (consisting of two nearby floating point numbers) to store the dimensions of M_i, I could use a *single* floating point number to store the width of the box. Right now, the size of the box cannot be any smaller than the machine epsilon; with rigorous bounds on the variational equation, my implicit M_i's could shrink as small as the smallest machine representable number.

The containment algorithm is also *local*, in the sense that it does not need to remember any further back than the previous timestep in order to inductively verify the existence of the shadow on the next timestep. In this sense, it is much more like a standard initial value problem integrator than other rigorous shadowing algorithms, which more closely resemle boundary value solvers and Newton's methods. These latter methods are more global in nature, which means they are more memory intensive, more monolithic, and less dynamic.

[4] Some may consider this a disadvantage.

[5] I'm not absolutely certain about this. But from my reading of all the rigorous shadow papers I've seen, they all appear to require rigorous bounds on the variational equation.

5 Further Work

5.1 For the thesis

The most important unfinished work is the main proof of containment: that the Inuductive Containment Property, which is what I have implemented, actually implies the existence of a shadow. I am virtually certain that it does, but my lack of ability for converting my strong geometrical intution into a formal proof seems to be a major stumbling block. I discuss some of the ideas I've tried in the Appendix. The discussion of the "complications" (section 2.3) also needs to be vastly improved, to show that they do not destroy the Inductive Containment Property.

Beyond that, the only other substantial research I feel is necessary for this thesis is to apply the containment algorithm to a host of problems, to see how it compares to previous work.

The thesis itself, of course, will contain a more thorough background discussion similar to the Qualifying Depth paper, and a broader discussion of what niche this research fills.

5.2 After the thesis

The original intent of this work was to be able to shadow N-body integrations similar to those of Quinlan & Tremaine (1992). Unfortunately, this algorithm does not seem to be able to find very long shadows of these systems, even though Quinlan & Tremaine (and later Hayes & Jackson (1997)) presented apparently strong evidence that long shadows exist. It appears that the difficulty lies in the N-body problem simply requiring so much numerical computation to advance the solution by one timestep that the interval package can not maintain tight bounds on the computations, and the intervals grow more quickly than the hyperbolicity of the problem can shrink them in the contracting direction. This hypothesis is corroborated by the observation that Nedialkov's package can find a *longer* traditional enclosure than I can find a shadow via containment, and by the fact that the shadow lengths I find are indepnedent of the local error I choose. Other avenues that could be followed to study the shadowability of N-body solutions include shadowing particles in synthetic galactic potentials (Merrit & Valluri 1996), or orbits of maps (not ODEs) constructed to emulate galactic potentials (Touma & Tremaine 1997). Also, I would like to attempt shadowing of long-term solar system integrations (*eg.,* Wisdom & Holman 1991).

The current work brings us no closer to understanding what glitches are, why they are caused, or how it may be possible to avoid them. (Recall that a "glitch" is the term used by Grebogi *et al.* to describe the event that causes our shadowing algorithm to fail in finding a shadow. Whether that event is more closely connected to the noisy trajectory or to the shadowing algorithm, or to some other factor, is currently an entirely open question.)

A Appendix: Discussion and preliminary proofs of simplified systems

First, I believe that the two and three-dimensional proofs elucidated in and around figures 1, 2, and 3 are rigorous enough as they are, with the possible exception that I'd need to prove that no holes in γ_i are introduced between steps. (I *do* have a proof that holes aren't introduced, thanks to the help of Prof. Paul

Selick of the Math Dept. I have not included it because it is meaningless to say that γ_i has no holes if I can't even prove γ_i exists for all i.) The only reason I'm not entirely convinced that the same kind of argument holds in higher dimensions is because I can't draw four-dimensional (or five- or seventeen-dimensional) pictures to convince myself; nor can I see any way to extend the 3 dimensional arguments geometrically to higher dimensions. Note that if you accept these figures and their accompanying argument as good enough for the correctness of the containment algorithm, and you accept that I have listed all the complications and handled them correctly, then the shadows of the Lorenz equations are as rigorous as they need to be, because the Lorenz system is three-dimensional.

Here is an algebraic proof of the two-dimensional containment argument, in the case that no rotation or skewing occurs, and M_i and M_{i+1} are the same size. Refer to figure 11. Note that all of M_i, $\phi(M_i)$, and M_{i+1} are axis-aligned and centred on the origin (they have been drawn side-by-side for clarity only). So, the

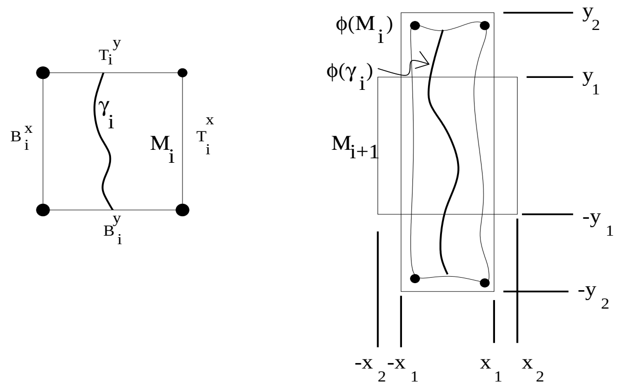

Figure 11: Simple, axis-aligned containment in 2 dimensions. The horizontal direction is contracting, and the vertical direction is expanding.

y-axis direction is the expanding direction, and the x-direction is contracting. We assume (without reference to the picture) that the γ-covering property is true at step i inside M_i:

$$\forall y \in [-y_1, y_1] \exists x \in [-x_2, x_2] : (x, y) \in \gamma_i.$$

Now, we assume that we've verified computationally that M_i and M_{i+1} satisfy the Inductive Containment Property, equations (8–11). Note that $\forall i, \ \ A_i = I$. Namely,

$$\begin{aligned} \phi(T_i^y) &> y_1 && (16) \\ \phi(B_i^y) &< -y_1 && (17) \\ x_2 &> \phi((x_2, 0)) && (18) \\ -x_2 &< \phi((-x_2, 0)) && (19) \end{aligned}$$

With the above, we can derive that the γ-covering property is satisfied in M_{i+1}, as follows. Since $\phi(T_i^y) > y_1$, we know that the top of $\phi(\gamma_i)$ is above y_1; similarly, since $\phi(B_i^y) < -y_1$, the bottom of $\phi(\gamma_i)$ is below $-y_1$. Furthermore, since $[\phi((-x_2, 0)), \phi((x_2, 0))] \subset [-x_1, x_1] \subset [-x_2, x_2]$, $\phi(\gamma_i)$ does not leave M_{i+1} through the sides. Since γ_i is continuous (and we assume ϕ is also continuous),

$$\forall y \in [top(\phi(\gamma_i))_y, bottom(\phi(\gamma_i))_y] \exists x \in [-x_1, x_1] : (x, y) \in \phi(\gamma_i).$$

Finally, since $[-y_1, y_1] \subset [top(\phi(\gamma_i))_y, bottom(\phi(\gamma_i))_y]$ and $[-x_1, x_1] \subset [-x_2, x_2]$,

$$\forall y \in [-y_1, y_1] \exists x \in [-x_2, x_2] : (x, y) \in \phi(\gamma_i) \Longrightarrow \forall y \in [-y_1, y_1] \exists x \in [-x_2, x_2] : (x, y) \in \gamma_{i+1},$$

and we are done.

It seems simple enough, but I suspect that the same argument in higher dimensions, taking account of rotating, skewing, puslating boxes is going to be very, very messy.

References

[1] C. Bendtsen and O. Stauling. FADBAD, a flexible C++ package for automatic differentiation, using the forward and backward methods. Technical Report IMM-REP-1996-17, Department of Mathematical Modelling, Technical University of Denmark, 2800 Lyngby, Denmark, August 1996.

[2] C. Bendtsen and O. Stauling. TADIFF, a flexible C++ package for automatic differentiation, using taylor series expansion. Technical Report IMM-REP-1997-07, Department of Mathematical Modelling, Technical University of Denmark, 2800 Lyngby, Denmark, April 1997.

[3] Shui-Nee Chow and Erik S. Van Vleck. A Shadowing Lemma Approach to Global Error Analysis for Initial Value ODEs. *SIAM Journal on Scientific Computing*, 15(4):959–976, July 1994.

[4] Brian A. Coomes, Hüseyin Koçak, and Kenneth J. Palmer. Shadowing orbits of ordinary differential equations. *Journal of Computational and Applied Mathematics*, 52:35–43, 1994.

[5] Brian A. Coomes, Hüseyin Koçak, and Kenneth J. Palmer. Rigorous computational shadowing of orbits of ordinary differential equations. *Numerische Mathematique*, 69:401–421, 1995.

[6] Celso Grebogi, Stephen M. Hammel, James A. Yorke, and Tim Sauer. Shadowing of Physical Trajectories in Chaotic Dynamics: Containment and Refinement. *Physical Review Letters*, 65(13):1527–1530, 24 September 1990.

[7] Wayne Hayes and Kenneth R. Jackson. Global error measures for the large gravitatonal n-body problem. In David A. Clarke and Michael J. West, editors, *The 12th "Kingston Meeting": Computational Astrophysics*, volume 123 of *ASP Conference Series*, pages 237–239. Astronomical Society of the Pacific, 1997.

[8] David Merritt and Monica Valluri. Chaos and mixing in triaxial stellar systems. *The Astrophysical Journal*, 471:82–105, 1996.

[9] James Murdock. Shadowing multiple elbow orbits: an application of dynamical systems to perturbation theory. *J. Diff. Eq.*, 119(1):224–247, 1995.

[10] N. S. Nedialkov, K. R. Jackson, and G. F. Corliss. Validated Solutions of Initial Value Problems for Ordinary Differential Equations. Submitted to SIAM Review, February, 1997.

[11] Gerald D. Quinlan and Scott Tremaine. On the reliability of gravitational N-body integrations. *Monthly Notices of the Royal Astronomical Society*, 259:505–518, 1992.

[12] J. Touma and S. Tremaine. A map for eccentric orbits in triaxial potentials. *Monthly Notices of the Royal Astronomical Society*, 292:905, 1997.

[13] Eric S. Van Vleck. Numerical Shadowing Near Hyperbolic Trajectories. *SIAM Journal on Scientific Computing*, 16(5), 1995.

[14] Jack Wisdom and Matthew Holman. Symplectic maps for the n-body problem. *The Astronomical Journal*, 102:1528–1538, October 1991.