Consistency Techniques in Ordinary Differential Equations

Yves Deville and Micha Janssen
({yde,mja}@info.ucl.ac.be)
Université catholique de Louvain,
Pl. Ste Barbe 2,
B-1348 Louvain-la-Neuve, Belgium

Pascal Van Hentenryck
(pvh@cs.brown.edu)
Box 1910
Brown University,
Providence, RI 02912, USA

Abstract.
This paper takes a fresh look at the application of interval analysis to ordinary differential equations and studies how consistency techniques can help address the accuracy problems typically exhibited by these methods, while trying to preserve their efficiency. It proposes to generalize interval techniques into a two-step process: a forward process that computes an enclosure and a backward process that reduces this enclosure. Consistency techniques apply naturally to the backward (pruning) step but can also be applied to the forward phase. The paper describes the framework, studies the various steps in detail, proposes a number of novel techniques, and gives some preliminary experimental results to indicate the potential of this new research avenue.

Keywords: Consistency, continuous problems, differential equations

1. Introduction

Differential equations (DE) are important in many scientific applications in areas such as physics, chemistry, and mechanics to name only a few. In addition, computers play a fundamental role in obtaining solutions to these systems.

THE PROBLEM
A (first-order) *ordinary differential equation* (ODE) system $\mathcal{O}$ is a system of the form

$$\begin{array}{rcl} {u_1}'(t) & = & f_1(t, u_1(t), \ldots, u_n(t)) \\ {u_2}'(t) & = & f_2(t, u_1(t), \ldots, u_n(t)) \\ & \vdots & \\ {u_n}'(t) & = & f_n(t, u_1(t), \ldots, u_n(t)) \end{array}$$

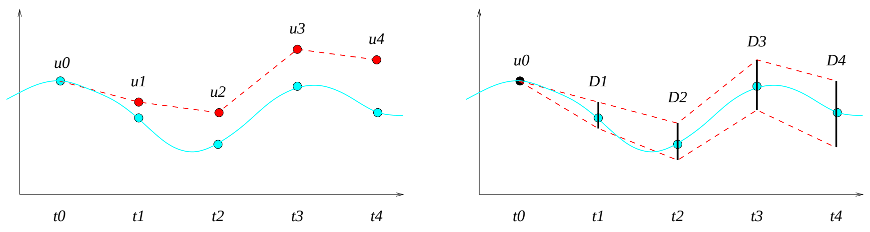

Figure 1. (a) Discrete methods. (b) Interval methods

In the following, we use the vector representation $u'(t) = f(t, u(t))$ or, more simply, $u' = f(t, u)$. We will also assume that function f is sufficiently smooth. Given an initial condition $u(t_0) = u_0$ and assuming existence and uniqueness of a solution, the solution of $\mathcal{O}$ is a function $s^* : \mathbb{R} \to \mathbb{R}^n$ satisfying $\mathcal{O}$ and the initial condition $s^*(t_0) = u_0$. Note that differential equations of order p (i.e., $f(t, u, u', u'', \ldots, u^{(p)}) = 0$) can always be transformed into an ODE by introduction of new variables.

Example 1. Given the ODE $u'(t) = -u^2(t)$, and the initial condition $u(0) = 0.1$, the solution is the function $s^*(t) = \frac{1}{t+10}$.

There exist different mathematical methods for proving the existence and uniqueness of a solution of an ODE system with initial value. But, in practice, a system is generally required, not only to prove existence, but also to produce numerical values of the solution $s^*(t)$ for different values of variable t. Although, for some classes of ODE systems, the solution can be represented in closed form (i.e., combination of elementary functions), it is safe to say that most ODE systems cannot be solved explicitly (Henrici, 1962). For instance, the innocent-looking equation $u' = t^2 + u^2$ cannot be solved in terms of elementary functions!

Discrete variable methods aim at approximating the solution $s^*(t)$ of *any* ODE system, not over a continuous range of t, but only at some points $t_0, t_1, \ldots, t_m$ (see Figure 1.(a)). Discrete variable methods include *one-step methods* (where $s^*(t_j)$ is approximated from the approximation u_{j-1} of $s^*(t_{j-1})$) and *multistep methods* (where $s^*(t_j)$ is approximated from the approximations $u_{j-1}, \ldots, u_{j-p}$ of $s^*(t_{j-1}), \ldots, s^*(t_{j-p})$) (Henrici, 1962). In general, these methods do not guarantee the existence of a solution within a given bound and may suffer from traditional numerical problems of floating-point systems.

Interval Analysis in ODE

Interval techniques for ODE systems were introduced by Moore (Moore, 1966). These methods provide numerically reliable enclosures of the exact solution at points $t_0, t_1, \dots, t_m$ (see Figure 1.(b)). To achieve this result, they typically apply a one-step Taylor interval method and make extensive use of automatic differentiation to obtain the Taylor coefficients (Moore, 1979; Rall, 1980; Rall, 1981; Corliss, 1988; Aberth, 1988). A description and a bibliography of the application of interval analysis to ODE systems can be found in (Berz et al., 1996). An extended bibliography on enclosure methods and related topics is also given in (Bohlender, 1996).

The major problem of interval methods on ODE systems is the explosion of the size of resulting boxes at points $t_0, t_1, \dots, t_m$. There are mainly two reasons for this explosion. On the one hand, step methods have a tendency to accumulate errors from point to point. On the other hand, the approximation of an arbitrary region by a box, called the wrapping effect, may introduce considerable loss of accuracy after a number of steps. One of the best systems in this area is Lohner's AWA (Lohner, 1987; Stauning, 1996). It uses the Picard iteration to prove existence and uniqueness and to find a rough enclosure of the solution. This rough enclosure is then used to compute correct enclosures using a mean value method and the Taylor expansion on a variational equation on global errors. It also applies coordinate transformations to reduce the wrapping effect.

Goal of the Paper

This paper mainly serves two purposes. First, it provides a unifying framework to extend traditional numerical techniques to intervals providing reliable enclosures. In particular, the paper shows how to extend explicit and implicit, one-step and multistep, methods to intervals. Second, the paper attempts to take a fresh look at the traditional problems encountered by interval techniques and to study how consistency techniques may help. It proposes to generalize interval techniques into a two-step process: a forward process that computes an enclosure and a backward process that reduces this enclosure. In addition, the paper studies how consistency techniques may help in improving the forward process and the wrapping effect.

The techniques are reasonably simple mathematically and algorithmically and were motivated by the same intuitions as the techniques at the core of the Numerica system (Van Hentenryck et al., 1997). In this respect, they should complement well existing methods. But, as was the case for Numerica, only extensive experimental evaluation will determine which combinations of these techniques are useful in practice and which application areas they are best suited for. Preliminary experimental results illustrate the potential benefits.

The techniques presented herein are complementary, and would benefit, those proposed in the ACLP language (Hickey, 1999). ACLP provides an interval-based constraint language allowing higher-order objects, which makes it possible to state differential equations. ODE systems in ACLP are transformed into a set of basic constraints according to the classical (one-step) Taylor interval method. Although the basic constraints are solved by an interval-based constraint solver, this approach is (roughly) equivalent to classical interval techniques for ODE systems.

Contribution

The main contribution of this paper is to take a fresh look at the solving of ODE systems using interval analysis and to show that consistency techniques can play a prominent role in solving these systems in the future. It presents a generic framework consisting of a forward phase (typical in interval analysis) and a backward phase to prune the enclosure (one of the main contributions of this paper). The paper also shows how consistency techniques can help both of these phases to reduce the accuracy problem.

This paper is a framework paper that pioneers some new directions. It is a revised and extended version of (Deville et al., 1998) where the idea of applying consistency techniques for the generation and for the reduction of enclosures was first described. Since the publication of its conference version, several results have indicated the potential of these extensions (Janssen et al., 1999; Nedialkov, 1999). Reference (Janssen et al., 1999) proposes filtering operators based on enclosures of interpolation polynomials to reduce the enclosures, while reference (Nedialkov, 1999) proposes a filtering operator based on a Hermite theorem. As mentioned, these results seem to indicate that the combination of interval analysis and consistency techniques is an interesting avenue for further research.

Organization

The rest of this paper is organized as follows. Section 2 provides the necessary background and notations. Section 3 presents the generic algorithm that can be instantiated to produce the various methods. Section 4 describes how to find bounding boxes. Section 5 describes the forward phase. Section 6 discusses the backward phase, i.e., the pruning component. Section 7 presents some experimental results. Section 8 concludes the paper.

2. Background and Definitions

This paper uses rather standard notations of interval programming. $\mathcal{F}$ denotes the set of $\mathcal{F}$-numbers, $\mathcal{D}$ the set of boxes $\subseteq \mathbb{R}^n$ whose bounds are in $\mathcal{F}$, $\mathcal{I}$ the set of intervals $\subseteq \mathbb{R}$ whose bounds are in $\mathcal{F}$, and D (possibly subscripted) denotes a box in $\mathcal{D}$. Given a real r and a subset A of $\mathbb{R}^n$, $\overline{r}$ denotes the smallest interval in $\mathcal{I}$ containing r and $\Box A$ the smallest box in $\mathcal{D}$ containing A. If a is an $\mathcal{F}$-number, a^+ and a^- denote the next and the previous $\mathcal{F}$-numbers. A canonical interval is an interval of the form $[a, a]$ or $[a, a^+]$, where a is an $\mathcal{F}$-number. A canonical box is a tuple of canonical intervals. If g is a function, $\hat{g}$ and G denote interval extensions of g. We also use $g_i(x)$ and $G_i(D)$ to denote the i^{th} component of $g(x)$ and $G(D)$. As usual, the interval relation $\approx$ is an interval extension of equality, i.e., $D_1 \approx D_2$ if $D_1 \cap D_2 \neq \emptyset$. For vectors, we use the notations $\mathbf{u}_k = \langle u_0, \ldots, u_k\rangle$, $\mathbf{D}_k = \langle D_0, \ldots, D_k\rangle$, and $\mathbf{t}_k = \langle t_0, \ldots, t_k\rangle$.

Because the techniques proposed in this paper use multistep solutions (which are partial functions), it is necessary to define interval extensions of partial functions.

Definition 1. (Interval Extension of a Partial Function) Let g be a (total) function, and h be a partial function. We denote

$$g(D) = \{g(x) \mid x \in D\}$$
$$h(D) = \{h(x) \mid x \in D \text{ and } x \text{ is in the domain of } h\}$$

The interval function G (resp. H) is an *interval extension* of g (resp. h), if forall $D : g(D) \subseteq G(D)$ (resp. $h(D) \subseteq H(D)$).

The solution of an ODE system can be formalized mathematically as follows.

Definition 2. (Solution of an ODE System with Initial Value) A *solution* of an ODE system $\mathcal{O}$ *with initial value* $u(t_0) = u_0$ is a function $s^*(t) : \mathbb{R} \to \mathbb{R}^n$ satisfying $\mathcal{O}$ and the initial conditions $s^*(t_0) = u_0$.

In this paper, we restrict attention to ODE systems that have a unique solution for a given initial value. Techniques to verify this hypothesis numerically are given in the paper. Moreover, as mentioned, the objective is to produce (an approximation of) the values of the solution function s^* of the system $\mathcal{O}$ at different points $t_0, t_1, \ldots, t_m$. It is thus useful to adapt the definition of a solution to account for this practical motivation.

Definition 3. (Solution of an ODE System) The *solution* of an ODE system $\mathcal{O}$ is the function

$$s(t_0, u_0, t_1) : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$$

such that $s(t_0, u_0, t_1) = s^*(t_1)$, where s^* is the solution of $\mathcal{O}$ with initial conditions $u(t_0) = u_0$.

The solution of an ODE system $\mathcal{O}$ can be used to obtain the solution of $\mathcal{O}$ at *any* point for *any* initial value.

Some methods for solving ODEs are multistep methods that compute the value at point t_k from values at point $t_{j-k}, \ldots, t_{j-1}$ (for some $k > 1$). Obviously, the values at points $t_1, \ldots, t_{k-1}$ must be computed by some other method. We extend our definition of solution to account for these methods.

Definition 4. (Multistep solution of an ODE) The multistep solution of an ODE system $\mathcal{O}$ is the *partial* function $ms : A \subseteq (\mathbb{R}^k \times (\mathbb{R}^n)^k \times \mathbb{R}) \to \mathbb{R}^n$ defined by

$$\begin{array}{rcll} ms(\mathbf{t}_{k-1}, \mathbf{u}_{k-1}, t) & = & s(t_0, u_0, t) & \text{if } \ u_i = s(t_0, u_0, t_i) \text{ for } \ 1 \leq i \leq k-1 \\ & & \text{undefined} & \text{otherwise} \end{array}$$

where s is the solution of $\mathcal{O}$.

Note that the multistep solution is only defined when $\langle t_0, u_0 \rangle, \ldots, \langle t_{k-1}, u_{k-1} \rangle$ are on the same solution function.
The next definition introduces the concept of bounding box that is fundamental to prove the existence and the uniqueness of a solution to an ODE system over a box and to bound the errors.

Definition 5. (Bounding Box) Let s be the solution of an ODE system $\mathcal{O}$. A box B is a *bounding box* of s in $[t_0, t_1]$ wrt D if, forall $t \in [t_0, t_1]$, $s(t_0, D, t) \subseteq B$.

Informally speaking, a bounding box is thus an enclosure of the solution on the *whole interval* $[t_0, t_1]$. The following proposition is an interesting topological property of solutions.

Theorem 1. Let $\mathcal{O}$ be an ODE system $u' = f(t, u)$ with $f \in \mathcal{C}$ (i.e., f is continuous), let s be the solution of $\mathcal{O}$ (i.e., existence and uniqueness), and let $Fr(D)$ be the frontier of D. Then,

1. $s(t_0, D, t_1)$ is a compact and connected set;
2. $s(t_0, Fr(D), t_1)$ is the frontier of $s(t_0, D, t_1)$.

procedure SOLVE(**in** $\mathcal{O}, D_0, \langle t_0, \ldots, t_m \rangle$, **out** $\langle D_1, \ldots, D_m \rangle$)
Pre: $\mathcal{O}$ is an ODE system, $t_i \in \mathcal{F}$, $D_0 \in \mathcal{D}$
$\quad$ $\mathcal{O}$ has a unique solution s over $[t_0, t_1]$ with $u(t_0) \in D_0$
Post: $s(t_0, D_0, t_i) \subseteq D_i \qquad (1 \leq i \leq m, \quad D_i \in \mathcal{D})$
begin
for each j **in** $1..m$ **do**
begin
$\quad B_{j-1} :=$ BOUNDINGBOX$(\mathcal{O}, D_{j-1}, t_{j-1}, t_j)$;
$\quad D_j :=$ STEP$(\mathcal{O}, \langle t_0, \ldots, t_{j-1} \rangle, \langle D_0, \ldots, D_{j-1} \rangle, t_j, \langle B_0, \ldots, B_{j-1} \rangle)$;
$\quad D_j :=$ PRUNE$(\mathcal{O}, \langle t_0, \ldots, t_j \rangle, \langle D_0, \ldots, D_j \rangle, \langle B_0, \ldots, B_{j-1} \rangle)$;
end ;
end ;

Figure 2. The SOLVE generic algorithm

Proof. Particular case of Theorem 3 (see Appendix). □

As a consequence, $s(t_{j-1}, D_{j-1}, t_j)$ can be computed by considering the frontier of D_{j-1} only.

3. The Generic Algorithm

The interval methods described in this paper can be viewed as instantiations of a generic algorithm. It is useful to present the generic algorithm first and to describe its components in detail in the rest of the paper. The generic algorithm, presented in Figure 2, is parametrized by three procedures: a procedure to compute a bounding box, since bounding-boxes are fundamental in obtaining enclosures, a step procedure to compute forward, and a procedure to prune the enclosures. Procedure BOUNDINGBOX computes a bounding box of an ODE system in an interval for a given box. Procedure STEP computes a box approximating the value of $s^*(t_j)$ given the approximations of $s^*(t_k)$ $(0 \leq k \leq j-1)$ and the bounding boxes $B_0, \ldots, B_{j-1}$. Procedure PRUNE prunes the box D_j at t_j using the previous boxes. The intuition underlying the basic steps of the generic algorithm is illustrated in Figure 3. The fundamental novelty in this generic algorithm is the PRUNE (or backward) component that is a natural place to integrate consistency techniques in traditional interval techniques as recent results have shown (Janssen et al., 1999). The next three sections review these three components. Note however that it is possible to use several step procedures, in which case the intersection of their results is also an enclosure.

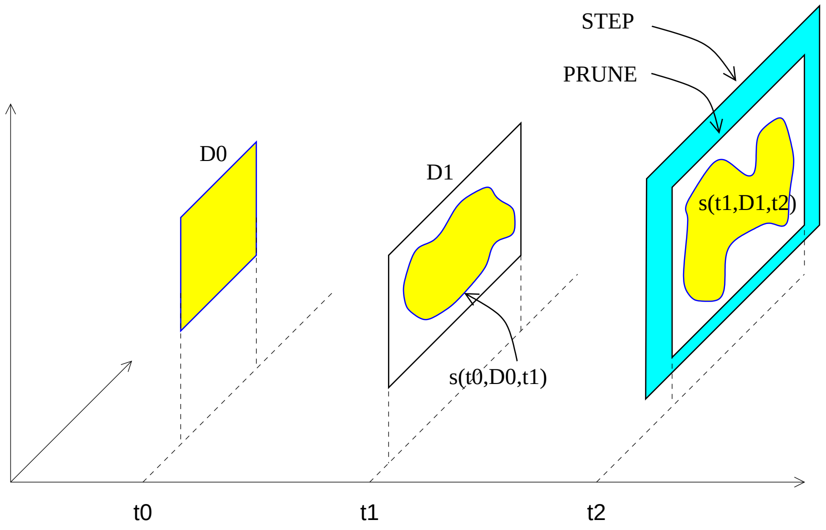

Figure 3. Computing correct enclosures of the solution.

4. The Bounding Box

This section considers how to obtain a bounding box for an ODE system. As will become clear later on, bounding boxes are fundamental to obtain reliable solutions to ODE systems. Bounding boxes will be used to bound the error terms, or more precisely to compute interval extensions of these error terms. The traditional interval techniques to obtain bounding boxes are based on the Picard operator (Hartman, 1964; Moore, 1979).

Theorem 2. (Picard Operator) Let D_0 and B be two boxes such that $D_0 \subseteq B$, let $[t_0, t_1] \in \mathcal{I}$, and let $h = t_1 - t_0$. Let $\mathcal{O}$ be an ODE system $u' = f(t, u)$, where f is continuous and has a continuous Jacobian (i.e first-order partial derivatives) over $[t_0, t_1]$. Let Φ be the transformation (Picard Operator)

$$\Phi(B) = D_0 + [0, h]F([t_0, t_1], B)$$

where F is an interval extension of f.
If $\Phi(B) \subseteq B$, then
1. The ODE system $\mathcal{O}$ with initial value $u(t_0) \in D_0$ has a unique solution s over $[t_0, t_1]$;
2. $\Phi(B)$ is a bounding box of s in $[t_0, t_1]$ wrt D_0.

Theorem 2 can be used for proving existence and uniqueness of a solution and for providing a bounding box (Lohner, 1987; Corliss, 1995).

function BOUNDINGBOX($\mathcal{O}, D_0, t_0, t_1$)
Pre: $\mathcal{O}$ is an ODE system, $t_0, t_1 \in \mathcal{F}$, $D_0 \in \mathcal{D}$
$\mathcal{O}$ has a unique solution s over $[t_0, t_1]$ with $u(t_0) \in D_0$
Post: BOUNDINGBOX $\in \mathcal{D}$
BOUNDINGBOX is a bounding box of s in $[t_0, t_1]$ wrt D_0
begin
{ *Finding a bounding box* }
$BB := D_0$;
while $\Phi(BB) \not\subseteq BB$ **do** $BB :=$ WIDEN(BB) ;
{ *Reducing the bounding box* }
while (size(BB) $-$ size($\Phi(BB)$)) $> \epsilon$ **do** $BB := \Phi(BB)$;
BOUNDINGBOX $:= BB$;
end ;

Figure 4. A BOUNDINGBOX specification and a possible algorithm.

The specification and a typical algorithm for BOUNDINGBOX are given in Figure 4. Notice that D_0 is always included in the bounding box, and that a bounding box is also a first (rough) enclosure of the solution at t_1. If B is a bounding box, $\Phi(B)$ is also a bounding box, included in B. The WIDEN function provides a (strictly) larger box. The algorithm does not necessarily terminate successfully. This occurs when BB becomes too large as we only work with finite intervals and $\mathcal{F}$ is finite. In that case, the step size should be reduced. The existence of the Jacobian of f, not handled in this algorithm, can be checked numerically by evaluating its interval extension over the box. Note also that the Picard operator uses a Taylor expansion of order 1. It can be generalized for higher orders, which may allow the step size to be increased, at the cost of more computation per step.

5. The Step Component

This section describes the STEP component. Step methods are presented in isolation. However, as mentioned previously, they can be used together, since the intersection of their results is also a step method. We concentrate here on *one-step methods*. Extensions to *multistep methods* and the treatment of the wrapping effect are described in the Appendix.

In a one-step method, an enclosure D_j is obtained from the enclosure D_{j-1} and a bounding box. It is thus a simplified version of our STEP component, and it is specified in Figure 5. The STEP function computes

function $\text{STEP}(\mathcal{O}, t_0, D_0, t_1, B_0)$
Pre: $\mathcal{O}$ is an ODE system, $t_0, t_1 \in \mathcal{F}$, $D_0 \in \mathcal{D}$
$\mathcal{O}$ has a unique solution s over $[t_0, t_1]$ with $u(t_0) \in D_0$
B_0 is a bounding box of s in $[t_0, t_1]$ wrt D_0
Post: $s(t_0, D_0, t1) \subseteq \text{STEP}$
$\text{STEP} \in \mathcal{D}$

Figure 5. Specification of the STEP component for one-step methods.

an interval extension of the solution s. Such an interval extension will be called an interval solution. [1]

Definition 6. (Interval Solution of an ODE System) Let s be the solution of an ODE system $\mathcal{O}$. An *interval solution* of $\mathcal{O}$ is an interval extension S of s, i.e.,

$$\forall t_0, t_1 \in \mathcal{F}\ ,\ D_0 \in \mathcal{D} : s(t_0, D_0, t_1) \subseteq S(t_0, D_0, t_1)$$

There exist different families of interval solutions depending on the underlying numerical method (explicit or implicit) and on the computation approach (direct or piecewise).

5.1. *Explicit One-Step Methods*

We first describe traditional numerical methods, move to traditional interval methods, and propose improvements which can be obtained from consistency techniques.

TRADITIONAL NUMERICAL METHODS
To understand traditional interval methods, it is useful to review traditional numerical methods. In explicit one-step methods, the solution s of an ODE system $\mathcal{O}$ is viewed as the summation of two functions.

Definition 7. (Explicit One-Step Solution) An *explicit one-step solution* of an ODE system $\mathcal{O}$ is the solution s of $\mathcal{O}$, expressed in the form

$$s(t_0, u_0, t_1) = sc(t_0, u_0, t_1) + e(t_0, u_0, t_1).$$

where the function sc is computable while the function e is not.

As a consequence, a traditional numerical method based on an explicit one-step method is an algorithm of the form

[1] As usual, interval solutions could also be defined on particular subsets of $\mathcal{F}$ and $\mathcal{D}$.

```
forall(i in 1..m)
   u_i := sc(t_{i-1}, u_{i-1}, t_i);
```

This algorithm tries to approximate the solution $s^*(t)$ for an initial value $u(t_0) = u_0$.

Example 2. (Taylor Method) The Taylor method is one of the best known explicit one-step methods where the functions sc and e are given by the Taylor expansion of a given order p, i.e.,

$$\begin{array}{rcl} sc_T(t_0, u_0, t_1) & = & u_0 + hf^{(0)}(t_0, u_0) + \frac{h^2}{2} f^{(1)}(t_0, u_0) + \dots \\ & & + \frac{h^p}{p!} f^{(p-1)}(t_0, u_0) \\ e_{T_i}(t_0, u_0, t_1) & = & \frac{h^{p+1}}{(p+1)!} f_i^{(p)}(\xi_i, s(t_0, u_0, \xi_i)) \end{array}$$

where, for a given u_0, we have $t_0 < \xi_i < t_1$, $1 \leq i \leq n$.

In the Taylor method, only sc_T is computed as the error function e_T is uncomputable.

Direct Interval Extensions

Traditionally, interval solutions are often constructed by considering an explicit one-step solution $s(t_0, u_0, t_1) = sc(t_0, u_0, t_1) + e(t_0, u_0, t_1)$, by taking an interval extension SC of sc and by using a bounding box to bound the error function e to obtain a function of the form $S(t_0, D_0, t_1) = SC(t_0, D_0, t_1) + E(t_0, D_0, t_1)$.

Definition 8. (Direct Explicit One-Step Interval solution) Let $s(t_0, u_0, t_1) = sc(t_0, u_0, t_1) + e(t_0, u_0, t_1)$ be an explicit one-step solution of an ODE system $\mathcal{O}$. A *direct explicit one-step interval solution* of s is an interval solution S of the form

$$S(t_0, D_0, t_1) = SC(t_0, D_0, t_1) + E(t_0, D_0, t_1).$$

where SC is an interval extension of sc, and E is an interval extension of e.

Example 3. (Taylor Interval Solution) The Taylor Interval Solution of order p of an ODE system $\mathcal{O}$ is defined as

$$\begin{array}{rcl} S_T(t_0, D_0, t_1) & = & D_0 + hF^{(0)}(t_0, D_0) + \frac{h^2}{2} F^{(1)}(t_0, D_0) + \dots \\ & & + \frac{h^p}{p!} F^{(p-1)}(t_0, D_0) + E_T(t_0, D_0, t_1) \\ E_T(t_0, D_0, t_1) & = & \frac{h^{p+1}}{(p+1)!} F^{(p)}([t_0, t_1], B_0) \end{array}$$

where $h = t_1 - t_0$, B_0 is a bounding box of s in $[t_0, t_1]$ wrt D_0, and the interval functions $F^{(j)}$ are interval extensions of functions $f^{(j)}$ inductively defined as follows (Moore, 1966) ($f^{(j)}$ is the "total j^{th} derivative of f wrt t")

$$\begin{array}{rcl} {f_i}^{(0)}(t,u(t))) & = & f_i(t,u(t)) \\ {f_i}^{(j)}(t,u(t)) & = & \dfrac{\partial {f_i}^{(j-1)}(t,u(t))}{\partial t} + \displaystyle\sum_{1\le m\le n} \dfrac{\partial {f_i}^{(j-1)}(t,u(t))}{\partial u_m} {f_m}^{(0)}(t,u(t)) \end{array}$$

More information on automatic generation of the value of these functions can be found in (Moore, 1979; Rall, 1980; Rall, 1981; Corliss, 1988; Aberth, 1988).

It is worth noticing that a bounding box B_0 is used here to obtain an interval extension E_T of the error function e_T. The Taylor interval method, based on the Taylor interval solution, is the classical interval method for solving ODE (Moore, 1966).

Other classical methods such as Runge-Kutta can be turned into interval solutions. However, as the error term contains the Taylor error term, these interval methods do not usually provide better enclosures.

Mean Value Form

In an explicit interval solution, intervals are growing as $D_0 \subseteq S(t_0, D_0, t_1)$. Mean value forms have been proposed to use contraction characteristics of functions and may (and usually do) return smaller intervals. From an explicit one-step solution

$$s(t_0, u_0, t_1) = sc(t_0, u_0, t_1) + e(t_0, u_0, t_1)$$

we may apply the mean value theorem on $sc(t_0, u, t_1)$ (on variable u) to obtain

$$\begin{array}{rcl} s_i(t_0,u,t_1) & = & sc_i(t_0,m,t_1) + \sum_{j=1}^{n} \left(\frac{\partial sc_i}{\partial (u)_j}\right)(t_0,\xi_i,t_1)\ (u_j - m_j) \\ & & +e_i(t_0,u,t_1) \end{array}$$

for some ξ_i between u and m $(1 \le i \le n)$. As a consequence, any interval solution of s may serve as a basis to define a new interval solution.

Definition 9. (MVF solution of an ODE system) Let D be a box $\langle I_1, \ldots, I_n\rangle$, m_i be the center of I_i, and $S_M = SC_M + E_M$ be an interval solution of an ODE system $\mathcal{O}$. The *MVF solution* of $\mathcal{O}$ in D wrt S_M, denoted by $\tau_M(t_0, D, t_1)$, is the interval solution

$$\begin{array}{rl} SC_M(t_0, \langle \overline{m_1}, \ldots, \overline{m_n}\rangle, t_1) & + \ \sum_{i=1}^{n} \widehat{\left(\frac{\partial sc}{\partial (u)_i}\right)}(t_0, D, t_1)\ (I_i - \overline{m_i}) \\ & + \ E_M(t_0, D_0, t_1) \end{array}$$

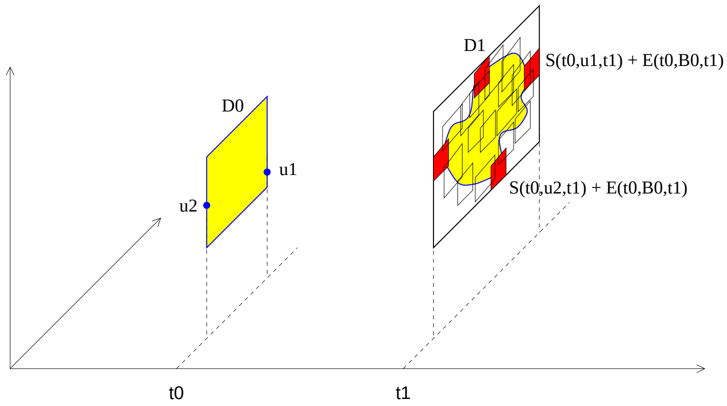

Figure 6. A Piecewise Interval Solution

In the above definition, the interval function $\widehat{\left(\frac{\partial sc}{\partial (u)_i}\right)}$ can be evaluated by automatic differentiation, during the evaluation of $SC(t_0, D, t_1)$.

Mean value form of Taylor expression has already been used in (Lohner, 1987).

PIECEWISE INTERVAL SOLUTION

Direct interval techniques propagate entire boxes through interval solutions. As a consequence, errors may tend to accumulate as computations proceed. This section investigates a new variety of techniques inspired by, and using, consistency techniques that can be proposed to reduce the accumulation of errors. The main idea, which is used several times in this paper and was inspired by box-consistency, is to propagate small boxes as illustrated in Figure 6.

Definition 10. (Piecewise Explicit One-Step Interval Solution) Let $s(t_0, u_0, t_1) = sc(t_0, u_0, t_1) + e(t_0, u_0, t_1)$ be an explicit one-step solution to an ODE system $\mathcal{O}$. A *piecewise explicit one-step interval solution* of s is a function $S(t_0, D, t_1)$ defined as

$$S(t_0, D_0, t_1) = \square\{SC(t_0, \overline{u_0}, t_1) \mid u_0 \in D_0\} + E(t_0, D_0, t_1)$$

where SC is an interval extension of sc, and E is an interval extension of e.

Piecewise interval solutions of an ODE system are not only a theoretical concept: they can in fact also be computed. The basic idea here is to express piecewise interval solution as unconstrained optimization problems.

Proposition 1. Let $s(t_0, u_0, t_1) = sc(t_0, u_0, t_1) + e(t_0, u_0, t_1)$ be an explicit one-step solution to an ODE system $\mathcal{O}$. A piecewise explicit one-step interval extension of s is a function $S(t_0, D, t_1)$ defined as

$$\begin{aligned} S_i(t_0, D_0, t_1) \; = \; & [min_{u \in D_0} SC_i(t_0, u, t_1), max_{u \in D_0} SC_i(t_0, u, t_1)] \\ & + E_i(t_0, D_0, t_1) \quad (1 \leq i \leq n) \end{aligned}$$

where SC is an interval extension of sc, and E is an interval extension of e.

Note that these minimization problems must be solved globally to guarantee reliable solutions. In (Deville et al., 1998), we discuss how a system like NUMERICA may be generalized to solve these problems. The efficiency of the system of course depends on the step size, on the size of D_0, and on the desired accuracy. It is interesting to observe that the function SC does not depend on the error term and hence methods that are not normally considered in the interval community (e.g., Runge-Kutta method) may turn beneficial from a computational standpoint. It is of course possible to sacrifice accuracy for computation time by using projections, the fundamental idea behind consistency techniques. For instance, interval methods are generally very fast on one-dimensional problems, which partly explains why consistency techniques have been used successfully to solve systems of nonlinear equations.

Definition 11. (Box-Piecewise Explicit One-Step Interval Solution) Let $s(t_0, u_0, t_1) = sc(t_0, u_0, t_1) + e(t_0, u_0, t_1)$ be an explicit one-step solution to an ODE system $\mathcal{O}$. A *box-piecewise explicit one-step interval solution* of s wrt dimension i is a function $S^i(t_0, D, t_1)$ defined as

$$\begin{aligned} & S^i(t_0, \langle I_1, \ldots, I_n \rangle, t_1) = \\ & \quad \square \{ SC(t_0, \langle I_1, \ldots, I_{i-1}, \overline{r}, I_{i+1}, \ldots, I_n \rangle, t_1) \mid r \in I_i \} + E(t_0, D_0, t_1) \end{aligned}$$

where SC is an interval extension of sc, and E is an interval extension of e. The *box-piecewise explicit one-step interval solution* of s wrt E and B_0 is the function

$$S(t_0, D_0, t_1) = \cap_{i \in 1..n} S^i(t_0, D_0, t_1)$$

Each of the interval solutions reduces to a one-dimensional (interval) unconstrained optimization problem. The following property is a direct consequence of the use of interval extensions in the (box-)piecewise approaches.

Proposition 2. ((Box-)Piecewise Explicit One-Step Interval Solution) The piecewise and box-piecewise one-step interval solutions are interval solutions.

In essence, box-piecewise solutions safely approximate a multi-dimensional problem by the intersection of many one-dimensional problems. Of course, it is possible, and probably desirable, to define notions such as box(k)-piecewise interval solutions where projections are performed on several variables. Finally, notice that in the above definitions, an interval extension $E(t_0, D_0, t_1)$ of the error function e is needed. Such an extension will use a bounding box over the whole box D_0. More precise interval solutions could be obtained if local error functions using local bounding boxes were considered in the above definitions. It is easy to generalize our definitions to integrate this idea.

5.2. Implicit One-Step Methods

This section considers implicit one-step methods. It first reviews traditional numerical methods and shows how they can be generalized to obtain interval methods. The presentation essentially follows the same lines as the previous section.

Traditional Numerical Methods

In implicit one-step methods, the solution of ODE $\mathcal{O}$ is viewed as the solution of an equation.

Definition 12. (Implicit One-Step Solution) An *implicit one-step solution* to an ODE system $\mathcal{O}$ is the solution s of $\mathcal{O}$, expressed in the form $s(t_0, u_0, t_1) = u_1$ where u_1 is the solution of an equation $u_1 = sc(t_0, u_0, t_1, u_1) + e(t_0, u_0, t_1)$.

Since the error term cannot be computed in general, the above equation is replaced in practice by its approximation $u_1 = sc(t_0, u_0, t_1, u_1)$. As a result, an implicit one-step method is an algorithm of the form

```
forall(i in 1..m)
   u_i := solve(u_i =sc(t_{i-1},u_{i-1},t_i,u_i));
```

where *solve(S)* returns an element x in *Solution(S)*, the set of solutions of S.

Example 4. (Trapezoid Method) The trapezoid method is an implicit one-step method that consists of solving, at each step, an equation of the form $u_1 = u_0 + \frac{h}{2}(f(t_0, u_0) + f(t_1, u_1))$.

Interval Methods

We now show how to generalize implicit one-step methods to intervals. The basic idea is to replace the search for a solution to a system of equations by a search for the solutions of a set of interval equations. The resulting interval solution can then be used as in explicit methods.

Definition 13. (Direct Implicit One-Step Interval Solution) Let $s(t_0, u_0, t_1) = u_1$ where u_1 is the solution of the equation $u_1 = sc(t_0, u_0, t_1, u_1) + e(t_0, u_0, t_1)$ be an implicit one-step solution of an ODE system $\mathcal{O}$. Let SC be an interval extension of sc and E be an interval extension of e. A *direct implicit one-step interval solution* of s is an interval function $S(t_0, D_0, t_1) = D_1$ where

$$D_1 = \Box\{D \subseteq B_0 \mid D \text{ is canonical } \& \ D \approx SC(t_0, D_0, t_1, D) + E(t_0, D_0, t_1)\}$$

and B_0 is a bounding box of s in $[t_0, t_1]$ wrt D_0. As defined in Section 2, canonical boxes are the smallest representable boxes.

Note that this definition amounts to finding all solutions of an "interval equation" in a box. The definition uses the bounding box as the initial search space. However, any step method can be used instead to provide a smaller search space.

Example 5. (Trapezoid Interval Method) The trapezoid interval solution of the trapezoid method requires the solving of the interval-valued equation

$$D \approx SC(t_0, D_0, t_1, D) + E(t_0, D_0, t_1)$$

with

$$\begin{aligned} SC(t_0, D_0, t_1, D) &= D_0 + \tfrac{h}{2}(F(t_0, D_0) + F(t_1, D)) \\ E(t_0, D_0, t_1) &= \tfrac{h^3}{12} F^{(2)}([t_0, t_1], B_0) \end{aligned}$$

where $h = t_1 - t_0$, F is an interval extension of f, and B_0 is a bounding box of s in $[t_0, t_1]$ wrt D_0.

It is possible to improve this result by incorporating the idea of piecewise interval solution proposed earlier. Coarser extensions can be defined in a similar way as well.

Implicit methods based on Taylor expression have already been developed in (Rihm, 1999).

Definition 14. (Piecewise Implicit One-Step Interval Solution) Let $s(t_0, u_0, t_1) = u_1$ where u_1 is the solution of the equation $u_1 = sc(t_0, u_0, t_1, u_1) + e(t_0, u_0, t_1)$ be an implicit one-step interval solution of an ODE system $\mathcal{O}$. Let SC be an interval extension of sc and E be an interval extension of e. A *piecewise implicit one-step interval solution* of s is an interval function $S(t_0, D_0, t_1) = D_1$ where

$$\begin{aligned} D_1 = \Box\{D \in B_0 \mid \ & D \approx SC(t_0, D_c, t_1, D) + E(t_0, D_0, t_1) \\ & \& \ D_c \subseteq D_0 \ \& \ D, D_c \text{ are canonical } \} \end{aligned}$$

and B_0 is a bounding box of s in $[t_0, t_1]$ wrt D_0.

6. The Pruning Component

As mentioned earlier, the major problem of interval methods for solving ODE systems is the explosion of the size of the enclosures at points $t_0, t_1, \ldots, t_m$. One of the main reasons for this explosion is that step methods have a tendency to accumulate errors from point to point. Basically, applying a step method at t_j to a canonical box produces a box at t_{j+1} which is not necessarily canonical. Finding pruning methods for reducing the size of enclosures is thus essential for practical applications of interval-based methods for solving ODE.

This section describes how to use consistency techniques to prune the enclosures. Section 6.1 recalls how pruning takes place in nonlinear programming and shows that the main difficulty in ODE systems is in finding ways of determining that a box cannot contain a solution. Algorithms to do so are called *filters* in this paper and are defined formally in Section 6.2. Section 6.3 then defines box-consistency for ODE systems in terms of filters. The remaining sections presents various possible filters.

6.1. Pruning in Nonlinear Programming

In nonlinear programming, a constraint $c(x_1, \ldots, x_n)$ can be used almost directly for pruning the search space (i.e., the cartesian products of the intervals I_i associated with the variables x_i). It suffices to take an interval extension $C(X_1, \ldots, X_n)$ of the constraint. Now if $C(I'_1, \ldots, I'_n)$ does not hold, it follows, by definition of interval extensions, that no solution of c lies in $I'_1 \times \ldots \times I'_n$. This basic property can be seen as a filtering operator that can be used for pruning the search space in many ways, including box(k)-consistency as in NUMERICA (Van Hentenryck et al., 1997; Van Hentenryck, 1998b). Recall that a constraint C is box(1)-consistent wrt $I_1, \ldots, I_n$ and x_i if the condition

$$C(I_1, \ldots, I_{i-1}, [l_i, l_i^+], I_{i+1}, \ldots, I_n) \wedge C(I_1, \ldots, I_{i-1}, [u_i^-, u_i], I_{i+1}, \ldots, I_n)$$

holds where $I_i = [l_i, u_i]$. The pruning algorithm based on box(1)-consistency reduces the interval of the variables without removing any solution until the constraint is box(1)-consistent wrt the intervals and all variables. Stronger consistency notions, e.g., box(2)-consistency, are also useful for especially difficult problems (Van Hentenryck, 1998a). It is interesting here to distinguish the filtering operator, i.e., the technique used to determine if a box cannot contain a solution, from the pruning algorithm that uses the filtering operator in a specific way to prune the search space.

6.2. Filters in ODE

Let us now define what a filter is in the context of ODE systems.

Definition 15. (Filter of an ODE System) A *filter* or *filtering operator* of an ODE system $\mathcal{O}$ is an interval constraint FL such that
if $\forall\, 1 \leq i \leq k: \; s(t_0, u_0, t_i) \in D_i$
then $FL(\mathbf{t}_k, \mathbf{D}_k)$ holds

Given boxes $\mathbf{D}_k$ at $\mathbf{t}_k$, the objective of a filter is thus to test the existence of a solution of $\mathcal{O}$ that goes through all the boxes. The number k of boxes has to be defined in actual instances of the filtering operator. How can we use a filter to obtain tighter enclosures of the solution? A simple technique consists of pruning the last enclosure produced by the forward process. A subbox $D \subseteq D_k$ can be pruned away if the condition

$$FL(\mathbf{t}_k, \langle D_0, \ldots, D_{k-1}, D \rangle)$$

does not hold.

6.3. Box-Consistency for ODE

We are now in position to define box consistency for ODE, aiming at pruning the enclosures without losing any solution.

Definition 16. (Interval Projection of an ODE System) An *interval projection ODE* $\langle \mathcal{O}, i \rangle$ is the association of an ODE $\mathcal{O}$ and of an index i $(1 \leq i \leq n)$.

Definition 17. (Box Consistency of an ODE System) Let FL be a filter of ODE. An interval projection ODE $\langle \mathcal{O}, i \rangle$ is *box-consistent* at t_j, D_j wrt $\langle t_0, \ldots, t_{j-1}, t_{j+1}, \ldots, t_k \rangle$ and $\langle D_0, \ldots, D_{j-1}, D_{j+1}, \ldots, D_k \rangle$ if

$$I_i = \Box\{\; p_i \;\in\; I_i \mid FL(\mathbf{t}_k, \langle D_0, \ldots, D_{j-1}, DP_j^i, D_{j+1}, \ldots, D_k \rangle)\}$$

where $D_j = \langle I_1, \ldots, I_n \rangle$
$DP_j^i = \langle I_1, \ldots, I_{i-1}, \overline{p_i}, I_{i+1}, \ldots, I_n \rangle$

An ODE system $\mathcal{O}$ is *box-consistent* if its projections are box-consistent.

A specification of the PRUNE procedure for our generic SOLVE algorithm, based on box consistency, is given in Figure 7. In the above definition, box consistency can be achieved for the different enclosures D_j. In the context of ODE with initial value, one can show that in our generic SOLVE algorithm, it is sufficient to achieve the box consistency

function $\text{PRUNE}(\mathcal{O}, \langle t_0, \ldots, t_k \rangle, \langle D_0, \ldots, D_k \rangle, \langle B_0, \ldots, B_{k-1} \rangle)$
Pre: $\mathcal{O}$ is an ODE system, $t_i \in \mathcal{F}$, $D_i \in \mathcal{D}$
 $\mathcal{O}$ has a unique solution s over $[t_0, t_1]$ with $u(t_0) \in D_0$
 B_i is a bounding box of s in $[t_i, t_{i+1}]$ wrt D_i
 $s(t_0, D_0, t_i) \subseteq D_i$
Post: $s(t_0, D_0, t_k) \subseteq \text{PRUNE} \subseteq D_k$
 $\mathcal{O}$ is box-consistent at t_k, D_k wrt $\langle t_0, \ldots, t_{k-1} \rangle$ and $\langle D_0, \ldots, D_{k-1} \rangle$

Figure 7. Specification of the PRUNE component based on box consistency

of the current enclosure. For other classes of problems (such as problems where enclosures are initially given at different points (Cruz and Barahona, 1999)), pruning realized at the current enclosure must be propagated through all the enclosures to achieve the box consistency. Stronger consistency notions, such as box(k)-consistency could also be defined easily. Notice that different filters can also be combined. Finally, it is also important to mention that the filtering operator can be used in many different ways, even if only the last enclosure is considered for pruning. For instance, once a box $D \subseteq D_k$ is selected, it is possible to prune the boxes $D_0, \ldots, D_{k-1}$ using, say, the forward process run backwards as already suggested in (Deville et al., 1998). This makes it possible to obtain tighter enclosures, thus obtaining a more effective filtering algorithm for D.

As usual, box consistency can be defined in a more procedural form that can be used in practical consistency algorithms.

Proposition 3. Let FL be a filter of ODE, $D_j = \langle I_1, \ldots, I_n \rangle$, and $I_i = [l_i, r_i]$. An interval projection ODE $\langle \mathcal{O}, i \rangle$ is *box-consistent* at t_j, D_j wrt $\langle t_0, \ldots, t_{j-1}, t_{j+1}, \ldots, t_k \rangle$ and $\langle D_0, \ldots, D_{j-1}, D_{j+1}, \ldots, D_k \rangle)$ iff, when $l_i \neq r_i$,

$$\begin{array}{ll} & FL(\mathbf{t}_k, \langle D_0, \ldots, D_{j-1}, DL_j^{+i}, D_{j+1}, \ldots, D_k \rangle) \\ \wedge & FL(\mathbf{t}_k, \langle D_0, \ldots, D_{j-1}, DR_j^{-i}, D_{j+1}, \ldots, D_k \rangle) \end{array}$$

and, when $l_i = r_i$,

$$FL(\mathbf{t}_k, \langle D_0, \ldots, D_{j-1}, DL_j^{i}, D_{j+1}, \ldots, D_k \rangle)$$

where
$$\begin{aligned} DL_j^{+i} &= \langle I_1, \ldots, I_{i-1}, [l_i, l_i^+], I_{i+1}, \ldots, I_n \rangle \\ DR_j^{-i} &= \langle I_1, \ldots, I_{i-1}, [r_i^-, r_i], I_{i+1}, \ldots, I_n \rangle \\ DL_j^{i} &= \langle I_1, \ldots, I_{i-1}, [l_i, l_i], I_{i+1}, \ldots, I_n \rangle \end{aligned}$$

Traditional propagation algorithms can now be defined to enforce box-consistency of ODE systems.

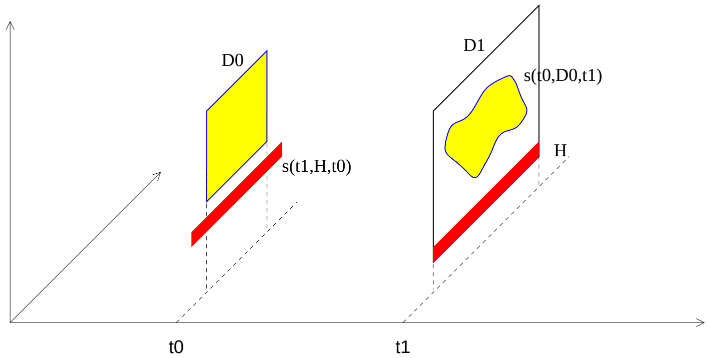

Figure 8. Pruning based on backward computation

6.4. Filters Based on Backward Computation

The fundamental intuition in this filter is illustrated in Figure 8. We know that all the solutions at t_0 are in D_0. If, in D_1, there is some box H such that $S(t_1, H, t_0) \cap D_0 = \emptyset$, then we know that the box H is *not* part of the solution at t_1. In other words, it is possible to use the step methods *backwards* as a filter to determine whether pieces of the box D_1 can be pruned away.

Proposition 4. (Backward Filter) Let S be a one-step interval solution of an ODE system $\mathcal{O}$.

$$FL(t_0, t_1, D_0, D_1) \quad \equiv \quad D_0 \cap S(t_1, D_1, t_0) \neq \emptyset$$

is a filtering operator.

We thus have different filters for the different one-step interval methods. Notice that using the same interval method in the Step component and in the filter does not preclude some pruning.

6.5. Filters Based on Implicit Methods

We already showed how traditional implicit numerical methods can be turned into implicit interval methods. These interval methods amount to finding all solutions D of an interval constraint of the form

$$D \approx SC(t_0, D_0, t_1, D) + E(t_0, D_0, t_1).$$

Instead of solving this constraint, it can also be used as a filter.

Proposition 5. (Implicit Filter) Let $s(t_0, u_0, t_1) = u_1$, where u_1 is the solution of the equation $u_1 = sc(t_0, u_0, t_1, u_1) + e(t_0, u_0, t_1)$, be an implicit one-step solution of an ODE system $\mathcal{O}$. Let SC be an interval extension of sc and E be an interval extension of e.

$$FL(t_0, t_1, D_0, D_1) \quad \equiv \quad D_1 \cap (SC(t_0, D_0, t_1, D_1) + E(t_0, D_0, t_1)) \neq \emptyset$$

is a filtering operator.

6.6. *Filters Based on Polynomial Interpolation*

A second approach that we developed in (Janssen et al., 1999) aims at using the equation $u' = f(t, u)$ as a filter. This equation cannot be used directly since u and u' are unknown functions. Assuming that we have at our disposal the multistep solution ms, the equation $u' = f(t, u)$ can be rewritten into

$$\frac{\partial ms}{\partial t}(\mathbf{t}_k, \mathbf{u}_k, t) = f(t, ms(\mathbf{t}_k, \mathbf{u}_k, t)).$$

At first sight, of course, this equation may not appear useful since ms is still an unknown function. However, it is possible to obtain interval extensions of ms and $\frac{\partial ms}{\partial t}$ by using, say, polynomial interpolations together with their error terms. If MS and DMS are such interval extensions, then we obtain an interval equation

$$DMS(\mathbf{t}_k, \mathbf{D}_k, t) = F(\bar{t}, MS(\mathbf{t}_k, \mathbf{D}_k, t))$$

that can be used as a filtering operator

$$FL(\mathbf{t}_k, \mathbf{D}_k)$$

A complete description of these filters as well as experimental results can be found in (Janssen et al., 1999).

7. Experimental Results

This section compares some standard interval techniques with piecewise interval solutions, and the use of filtering operators. The goal is to show that consistency techniques in the STEP component and in the PRUNE component can bring substantial gain in precision. The results were computed with NUMERICA with a precision of `1e-8`, using optimal bounding boxes.

Consider the ODE $u'(t) = -u(t)$ for an initial box [-1,1] at $t_0 = 0$. Figure 9 compares the results obtained by an interval Taylor method of order 4 with step size 0.5, the results obtained by the piecewise interval extension of the same method, and the exact solutions. Relative

t	Taylor *Result*	Taylor *Error*	Piecewise Taylor *Result*	Piecewise Taylor *Error*	Exact solution
0.0	[-1.00000 , 1.00000]	0%	[-1.00000 , 1.00000]	0.00%	[-1.00000 , 1.00000]
0.5	[-1.64870 , 1.64870]	171%	[-0.60703 , 0.60703]	0.08%	[-0.60653 , 0.60653]
1.0	[-2.71826 , 2.71821]	638%	[-0.36849 , 0.36849]	0.17%	[-0.36788 , 0.36788]
1.5	[-4.48150 , 4.48150]	1908%	[-0.22368 , 0.22368]	0.25%	[-0.22313 , 0.22313]
2.0	[-7.38864 , 7.38864]	5359%	[-0.13578 , 0.13578]	0.33%	[-0.13534 , 0.13534]
2.5	[-12.18163 , 12.18163]	14740%	[-0.08242 , 0.08242]	0.41%	[-0.08209 , 0.08209]
3.0	[-20.08383 , 20.08383]	40239%	[-0.05003 , 0.05003]	0.50%	[-0.04979 , 0.04979]
3.5	[-33.11217 , 33.11217]	109552%	[-0.03037 , 0.03037]	0.58%	[-0.03020 , 0.03020]
4.0	[-54.59196 , 54.59196]	297962%	[-0.01844 , 0.01844]	0.66%	[-0.01832 , 0.01832]

Figure 9. ODE $u'(t) = -u(t)$

t	Taylor MVF *Result*	Taylor MVF *Error*	Piecewise Taylor *Result*	Piecewise Taylor *Error*	Exact solution
0.0	[0.10000 , 0.40000]	0.00%	[0.10000 , 0.40000]	0.00%	[0.10000 , 0.40000]
0.5	[0.06798 , 0.37635]	29.52%	[0.09511 , 0.33344]	0.10%	[0.09524 , 0.33333]
1.0	[0.03884 , 0.36099]	65.37%	[0.09075 , 0.28583]	0.14%	[0.09091 , 0.28571]
1.5	[0.01027 , 0.35316]	110.31%	[0.08679 , 0.25010]	0.16%	[0.08696 , 0.25000]
2.0	[-0.02004 , 0.35314]	168.68%	[0.08318 , 0.22231]	0.18%	[0.08333 , 0.22222]
2.5	$[-\infty , +\infty]$		[0.07985 , 0.20007]	0.19%	[0.08000 , 0.20000]
3.0			[0.07678 , 0.18188]	0.19%	[0.07692 , 0.18182]
3.5			[0.07394 , 0.16672]	0.20%	[0.07407 , 0.16667]
4.0			[0.07131 , 0.15389]	0.21%	[0.07143 , 0.15385]
4.5			[0.06885 , 0.14290]	0.21%	[0.06897 , 0.14286]
5.0			[0.06656 , 0.13337]	0.21%	[0.06667 , 0.13333]

Figure 10. ODE $u'(t) = -u^2(t)$

errors on the size of the boxes are also given. As can be seen, the intervals of the traditional Taylor method grow quickly, although this function is actually contracting. The piecewise interval extension, on the other hand, is close to the exact solutions and is able to exploit the contraction characteristics of the function.

Consider now the ODE $u'(t) = -u^2(t)$ for an initial box $[0.1, 0.4]$ at $t_0 = 0$. Figure 10 compares the results obtained by a mean value form of a Taylor method of order 4, the results obtained by the piecewise interval extension of the Taylor method of order 4, and the exact solutions. Once again, it can be seen that the standard method leads to an explosion of the size of the intervals, while the piecewise interval extension is close to the exact results. Note that the Taylor method of order 4 also behaves badly on this ODE.

t	Piecewise Taylor	Taylor with Pruning	Ratio
0.0	[0.00000 , 0.00000]	[0.00000 , 0.00000]	1.0
0.3	[-0.30291 , 0.89395]	[-0.07389 , 0.41865]	2.4
0.6	[-2.07810 , 3.20739]	[0.23293 , 0.87991]	8.2
0.9	[-8.65903 , 10.22568]	[0.15993 , 1.19334]	18.3
1.2	[-31.47707 , 33.34114]	[0.22960 , 1.62460]	46.5
1.5	[-109.32716 , 111.32215]	[0.09149 , 1.95039]	118.7
1.8	[-374.18327 , 376.13097]	[-0.06153 , 2.33463]	313.1
2.1	[-1274.89513 , 1276.62155]	[-0.48267 , 2.66253]	811.2
2.4	[-4337.28310 , 4338.63402]	[-1.10181 , 3.05072]	2089.3
2.7	[-14749.13410 , 14749.98886]	[-2.04553 , 3.44376]	5373.9
3.0	[-50148.94757 , 50149.22981]	[-3.27441 , 3.96133]	13861.5

Figure 11. ODE $u'(t) = -10(u(t) - \sin(t)) + \cos(t)$

Our final example is the ODE $u'(t) = -10(u(t) - \sin(t)) + \cos(t)$, which is a stiff problem. Figure 11 compares the piecewise interval extension of the Taylor method (order 4), and the result obtained by an interval Taylor method (order 4) with a PRUNE step, using box consistency, with a filter based on polynomial interpolation. It shows an explosion of the piecewise Taylor method, although it is the best forward method possible. The pruning step, although applied on a classical interval Taylor forward step, substantially reduces the explosion in this case. This clearly shows that the pruning step is orthogonal to the forward step (since it improves the best possible forward step). Other experiments are presented in (Deville et al., 1998; Janssen et al., 1999).

8. Conclusion

This paper studied the application of interval analysis and consistency techniques to ordinary differential equations. Its main contribution is to take a fresh look at the solving of ODE systems using interval analysis and to show that consistency techniques may play a prominent role in solving these systems in the future. It presented a generic framework consisting of a forward phase (typical in interval analysis) and a backward phase to prune the enclosures produced by the forward phase (one of the main contributions of this paper). The paper also shows how consistency techniques can help both of these phases to reduce the accuracy problem. In particular, it presented various approaches to prune the enclosures that seem to produce significant improvement in accuracy in practice.

This paper is a revised and extended version of (Deville et al., 1998) where the idea of applying consistency techniques for the generation and for the reduction of enclosures was first described. Since the publication of its conference version, several results (Janssen et al., 1999; Nedialkov, 1999) have appeared, which are natural instantiations of the framework proposed herein. Reference (Janssen et al., 1999) proposes filtering operators based on enclosures of interpolation polynomials to reduce the enclosures, while reference (Nedialkov, 1999) proposes a filtering operator based on a Hermite theorem. These results seem to indicate that the combination of interval analysis and consistency techniques is indeed an interesting avenue for further research. Extensive experimental evaluation of these ideas is the next natural step to validate this belief and will be our main research topic in the near future.

Acknowledgment

Many thanks to Philippe Delsarte for fruitful discussions. We would also thank reviewers for their helpful and constructive comments. This research is partially supported by the *Actions de recherche concertées (ARC/95/00-187)* of the Direction générale de la Recherche Scientifique – Communauté Française de Belgique, the Belgian *Fonds National de la Recherche Scientifique*, and by an NSF NYI award.

References

Aberth, O.: 1988, *Precise Numerical Analysis.* Dubuque, Iowa: William Brown.

Berz, M., Bischof, C., Corliss, G. and Griewank, A. (eds.): 1996, *Computational Differentiation: Techniques, Applications, and Tools.* Philadelphia, Penn.: SIAM.

Bohlender, G.: 1996, 'Literature on Enclosure Methods and Related Topics'. Technical Report `www.uni-karlsruhe.de/ Gred.Bohlender`, Institut fr Angewandte Mathematik, Universitt Karlsruhe.

Corliss, G.: 1995, *Theory of Numerics in Ordinary and Partial Diffential Equations (W.A. Light, M. Machetta (Eds)*, Vol. Vol IV, Chapt. Guaranteed Error Bounds for Ordinary Differential Equations, pp. 1–75. Oxford University Press.

Corliss, G. F.: 1988, 'Applications of Differentiation Arithmetic'. In: R. E. Moore (ed.): *Reliability in Computing.* London: Academic Press, pp. 127–148.

Cruz, J. and Barahona, P.: 1999, 'An Interval Constraint Approach to Handle Parametric Ordinary Differential Equations for Decision Support'. In: J. Jaffar (ed.): *Principles and Practice of Constraint Programming (CP99).* pp. 478–479.

Davey, D. and Stewart, N.: 1976, 'Guaranteed Error Boubds for the Initial Value problem Using Polytope Arithmetic'. *BIT* **16**, 257–268.

Deville, Y., Janssen, M. and Van Hentenryck, P.: 1998, 'Consistency Techniques in Ordinary Differential Equations'. In: M. Maher and J.-F. Puget (eds.): *Principles and Practice of Constraint Programming (CP98)*, LNCS 1520, Springer Verlag, pp. 162–176.

Hartman, P.: 1964, *Ordinary Differential Equations.* Wiley, New York.
Henrici, P.: 1962, *Discrete Variable Methods in Ordinary Differential Equations.* John Wiley & Sons, New York.
Hickey, T.: July 1999, 'Analytic Constraint Solving and Interval Arithmetic'. Technical Report Cs-99-203, Michtom Schooll of Computer Science, Brandeis University.
Janssen, M., Deville, Y. and Van Hentenryck, P.: 1999, 'Multistep filtering operators for ordinary differential equations'. In: J. Jaffar (ed.): *Principles and Practice of Constraint Programming (CP99).* LNCS 1713, Springer Verlag, pp. 246–260.
Lohner, R. J.: 1987, 'Enclosing the solutions of ordinary initial and boundary value problems'. In: E. W. Kaucher, U. W. Kulisch, and C. Ullrich (eds.): *Computer Arithmetic: Scientific Computation and Programming Languages.* Stuttgart: Wiley-Teubner Series in Computer Science, pp. 255–286.
Moore, R.: 1966, *Interval Analysis.* Englewood Cliffs, NJ: Prentice-Hall.
Moore, R.: 1979, *Methods and Applications of Interval Analysis.* SIAM Publ.
Nedialkov, N. S.: 1999, 'Computing rigorous bounds on the solution of an initial value problem for an ordinary differential equation'. Ph.D. thesis, University of Toronto.
Rall, L. B.: 1980, 'Applications of Software for Automatic Differentiation in Numerical Computation'. In: G. Alefeld and R. D. Grigorieff (eds.): *Fundamentals of Numerical Computation (Computer Oriented Numerical Analysis)*, Computing Supplement No. 2. Berlin: Springer-Verlag, pp. 141–156.
Rall, L. B.: 1981, *Automatic Differentiation: Techniques and Applications*, LNCS 120, Springer-Verlag.
Rihm, R.: 1999, 'Implicit Methods for Enclosing Solutions of ODEs'. *Journal of Universal Coomputer Science* **4(2)**, 202–209.
Stauning, O.: 1996, 'Enclosing Solutions of Ordinary Differential Equations'. Technical Report Tech. Report IMM-REP-1996-18, Technical University Of Denmark.
Stewart, N.: 1971, 'A Heuristic to Reduce the Wrapping Effect in the Numerical Solution of ODE'. *BIT* **11**, 328–337.
Van Hentenryck, P.: 1998a, 'A Constraint Satisfaction Approach to a Circuit Design Problem'. *Journal of Global Optimization* **13**, 75–93.
Van Hentenryck, P.: 1998b, 'A Gentle Introduction to Numerica'. *Artificial Intelligence* **103(1-2)**, 209–235.
Van Hentenryck, P., Laurent, M. and Deville, Y.: 1997, *Numerica, A Modeling Language for Global Optimization.* MIT Press.

Appendix

A.1. Proofs of the Results

We prove here Theorem 3, a general version of Proposition 1. We begin with two lemmas. The first one is classical; the proof of the second one is straightforward.

Lemma 1. Let $A \subseteq \mathbb{R}^n$ and $g : A \to \mathbb{R}^m$ be continuous on A. If A is a compact (resp. connected) set, then $g(A)$ is a compact (resp. connected) set.

Lemma 2. A is a closed set iff $Fr(A) \subseteq A$.

The following proposition guarantees continuity of the solution s of ODE $u' = f(t,u)$ under the (weak) condition that f be a continuous function (Hartman, 1964).

Proposition 6. (Continuity of the solution s of an ODE) Let f be continuous on an open (t,u)-set $E \subseteq \mathbb{R} \times \mathbb{R}^n$ with the property that for every $(t_0, u_0) \in E$, the initial value problem $\{u' = f(t,u), u(t_0) = u_0\}$ has a unique solution $u(t) \equiv s(t_0, u_0, t)$. Let $T(t_0, u_0)$ be the maximal interval of existence of $u(t) = s(t_0, u_0, t)$. Then s is continuous on $\{(t_0, u_0, t) | (t_0, u_0) \in E, t \in T(t_0, u_0)\}$.

The following proposition states sufficient conditions of a function such that the image of the frontier of a compact set is the frontier of the image of this set. Two proofs will be given. The first one, based on elementary analysis, is longer but is self-contained. The second proof, provided by a reviewer, is elegant and much shorter; it reduces the property into an isomorphism on topologies.

Proposition 7. Let $U \subseteq \mathbb{R}^n$ be an open set. Let $D \subseteq U$ be a compact set. If the function $g : U \to \mathbb{R}^m$ is continuous on U, and if the inverse function g^{-1} exists and is continuous on an open set $V \supseteq g(D)$, then $g(Fr(D)) = Fr(g(D))$.

Proof. (Version 1)
By Lemma 1, $g(D)$ is a closed set. Thus, by Lemma 2, $Fr(g(D)) \subseteq g(D)$. As a consequence, we have to show that :

(a) $x \in Int(D) \Rightarrow y = g(x) \in Int(g(D))$;
(b) $y \in Int(g(D)) \Rightarrow x = g^{-1}(y) \in Int(D)$.

(a) Assume that $x \in Int(D)$ and $y = g(x)$. Let us choose $\varepsilon > 0$ such that

$$B(x, \varepsilon) \subseteq D. \tag{1}$$

There exists such an ε since, by hypothesis, $x \in Int(D)$. As V is an open set and $y \in g(D) \subseteq V$, we can find $\delta_1 > 0$ satisfying :

$$B(y, \delta_1) \subseteq V.$$

Function g^{-1} is continuous on V, so there exists δ_2 such that $0 < \delta_2 < \delta_1$ and verifying

$$g^{-1}(B(y,\delta_2)) \subseteq B(x,\varepsilon).$$

By (1), we obtain

$$g^{-1}(B(y,\delta_2)) \subseteq D.$$

Thus, we have

$$B(y,\delta_2) \subseteq g(D),$$

which means that $y \in Int(g(D))$.

(b) Similar to (a), using the continuity of function g.

□

Proof. (Version 2) By the hypotheses, g induces a homeomorphism between $g^{-1}(V)$ and V (i.e. a 1-1, bicontinuous map), thus it induces an isomorphism on the topologies (i.e. a 1-1 correspondence between open sets). Since the frontier is defined entirely in topological terms, $g(Fr(D)) = Fr(g(D))$.

□

The main theorem is basically an application of Proposition 7 on the solution of an ODE system. We will use the following notation. Let $g : \mathbb{R}\times\mathbb{R} \to \mathbb{R} : (x,y) \mapsto g(x,y)$ and let $a \in \mathbb{R}$. Then, $g(a,\bullet)$ denotes the one-variable function $g(a,\bullet) : \mathbb{R} \to \mathbb{R} : y \mapsto g(a,y)$. A similar definition holds for $g(\bullet,a)$. The generalization to $\mathbb{R}^n \to \mathbb{R}^m$ (partial) functions is straightforward.

Theorem 3. (Topological property of the solution s of an ODE)
Let f be continuous on an open (t,u)-set $E \subseteq \mathbb{R} \times \mathbb{R}^n$ with the property that for every $(t_0,u_0) \in E$, the initial value problem $\{u' = f(t,u), u(t_0) = u_0\}$ has a unique solution $u(t) \equiv s(t_0,u_0,t)$. Let T be an open interval and $t_0, t_1 \in T$. Let $U \subseteq \mathbb{R}^n$ be an open set such that $\{(t_0,u_0)|u_0 \in U\} \subseteq E$. Let $D \subseteq U$ be a compact and connected set. Let V be an open set such that $s(t_0,D,t_1) \subseteq V$ and $\{(t_1,u_1)|u_1 \in V\} \subseteq E$. If $s(t_0,u_0,\bullet)$ is defined on T for each $u_0 \in U$ and if $s(t_1,u_1,\bullet)$ is defined on T for each $u_1 \in V$, then

1. $s(t_0,D,t_1)$ is a compact and connected set;
2. $s(t_0,Fr(D),t_1) = Fr(s(t_0,D,t_1))$.

Proof.
1. Lemma 1.
2. By Proposition 6, function $s(t_0,\bullet,t_1)$ is continuous on U and function

function $\text{STEP}(\mathcal{O}, \langle t_0, \ldots, t_{k-1}\rangle, \langle D_0, \ldots, D_{k-1}\rangle, t_k, \langle B_0, \ldots, B_{k-1}\rangle)$
Pre: $\mathcal{O}$ is an ODE system, $t_i \in \mathcal{F}$, $D_i \in \mathcal{D}$
B_i is a bounding box of s in $[t_i, t_{i+1}]$ wrt D_i
$\mathcal{O}$ has a unique multistep solution ms over $[t_0, t_k]$ with $u(t_0) \in D_0$
Post: $ms(\langle t_0, \ldots, t_{k-1}\rangle, \langle D_0, \ldots, D_{k-1}\rangle, t_k) \subseteq \text{STEP}$
$\text{STEP} \in \mathcal{D}$

Figure 12. Specification of the STEP component for multistep methods (of order k)

$s(t_1, \bullet, t_0)$ is continuous on V. We can then apply Proposition 7, by instantiating $g \leftarrow s(t_0, \bullet, t_1)$ and $g^{-1} \leftarrow s(t_1, \bullet, t_0)$.

□

A.2. The Step Component: Multistep Methods

In a multistep method, an enclosure D_j is obtained from enclosures $D_{j-k}, \ldots, D_{j-1}$, and the associated bounding boxes. The number k of enclosures is called the order of the multistep method. The STEP function, specified in Figure 12, computes an interval extension of the multistep solution ms. Such an interval extension will be called a multistep interval solution.

Definition 18. (Multistep Interval Solution of an ODE System) Let ms be the multistep solution of an ODE system $\mathcal{O}$. A *multistep interval solution* of $\mathcal{O}$ is an interval extension S of ms, i.e.,

$$\forall \mathbf{t}_{k-1}, t_k, \mathbf{D}_{k-1} \quad ms(\mathbf{t}_{k-1}, \mathbf{D}_{k-1}, t_k) \subseteq S(\mathbf{t}_{k-1}, \mathbf{D}_{k-1}, t_k)$$

EXPLICIT MULTISTEP METHODS
In explicit multistep methods, the solution s of ODE $\mathcal{O}$ is decomposed as follows :

$$ms(\mathbf{t}_{k-1}, \mathbf{u}_{k-1}, t_k) = msc(\mathbf{t}_{k-1}, \mathbf{u}_{k-1}, t_k) + e(\mathbf{t}_{k-1}, \mathbf{u}_{k-1}, t_k) \tag{2}$$

These methods can be generalized to intervals in a way similar to one-step methods. For brevity, we only give an example of such an interval method.

Example 6. (Adams-Bashforth Interval Solution (Order 4)) Let h be $t_i - t_{i-1}$ for $1 \leq i \leq 4$. The Adams-Bashforth multistep interval solution of order 4 is the multistep interval solution

$$S_{AB}(\langle t_0, t_1, t_2, t_3\rangle, \langle D_0, D_1, D_2, D_3\rangle, t_4, B) = D_4$$

where

$$D_4 = D_3 \ +\tfrac{h}{24}(55F(t_3, D_3) - 59F(t_2, D_2) + 37F(t_1, D_1) - 9F(t_0, D_0)) \\ +\tfrac{251h^5}{720}F^{(4)}([t_0, t_4], B)$$

and B is a bounding box of s in $[t_0, t_4]$ wrt D_0. Notice that $F^{(4)}([t_0, t_4], B)$ can be approximated by $\Box(\cup_{0\leq i<4}F^{(4)}([t_i, t_{i+1}], B_i))$ where B_i is a bounding box of s in $[t_i, t_{i+1}]$ wrt D_i.

IMPLICIT MULTISTEP METHODS

Implicit multistep methods can be defined in a similar fashion. Let $u_k = ms(\mathbf{t}_{k-1}, \mathbf{u}_{k-1}, t_k)$. The value of u_k is the solution of the equation

$$u_k = msc(\mathbf{t}_{k-1}, \mathbf{u}_{k-1}, t_k, u_k) + e(\mathbf{t}_{k-1}, \mathbf{u}_{k-1}, t_k)$$

Example 7. (Adams-Moulton Implicit Interval Solution (order 3)) Let $h = t_i - t_{i-1}$ for $1 \leq i \leq 3$. The Adams-Moulton implicit multistep interval solution is the function defined as

$$S_{AM}(\langle t_0, t_1, t_2\rangle, \langle D_0, D_1, D_2\rangle, t_3, B) = D_3$$

where

$$\begin{aligned} D_3 = \Box\{D \in B \mid \quad & D \approx D_2 + \tfrac{h}{24}(9F(t_3, D) + 19F(t_2, D_2) - 5F(t_1, D_1) \\ & \qquad +F(t_0, D_0)) + \tfrac{-19h^5}{720}F^{(4)}([t_0, t_3], B) \\ & \&\ D \text{ is canonical} \\ & \&\ B \text{ is a bounding box of } s \text{ in } [t_0, t_3] \text{ wrt } D_0\}. \end{aligned}$$

A.3. Wrapping Effect

The wrapping effect is the name given to the error resulting from the enclosure of a region (which is not a box) by a box. It only occurs for multidimensional functions. In one dimension, a perfect interval extension of a continuous function g always yields the correct interval. However, a perfect interval extension of a multidimentional function g introduces overestimations in the resulting box, because the set $g(D) = \{g(d) | d \in D\}$ is not necessarily a box. This effect is especially important when the enclosure is used for finding a new region which is also enclosed by a box. The wrapping effect is thus central in interval methods for ODE. The following classical example, due to Moore (Moore, 1966) and explained in (Corliss, 1995), illustrates this problem :

$$u' = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} u \quad \text{with} \quad u_0 \in \begin{pmatrix} -0.1 & 0.1 \\ 0.9 & 1.1 \end{pmatrix}$$

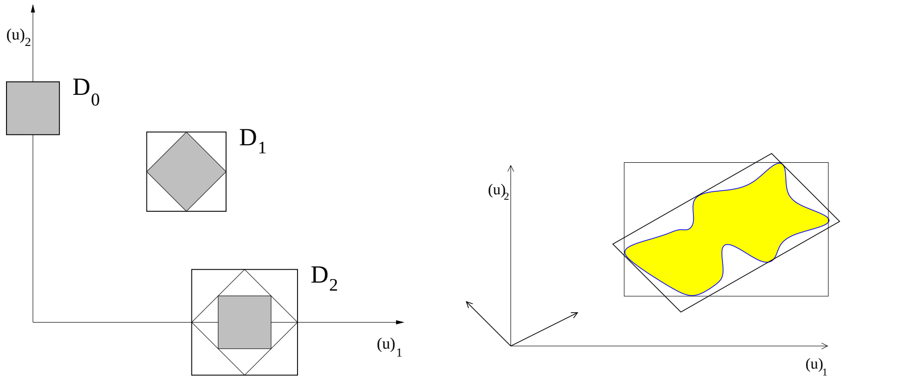

Figure 13. (a) The Wrapping Effect (b) Reducing the Overestimation by Coordinate Transformation

The trajectories of individual point-valued solutions of this ODE are circles in the $((u)_1, (u)_2)$-phase space. The set of solution values is a rotated rectangle. Figure 13.(a) shows that the resulting boxes at t_{j-1}, t_j, t_{j+1}. Moore shows that the width of the enclosures grow exponentially even if the stepwise $(t_j - t_{j-1})$ converges to zero. The wrapping effect can be reduced by changing the coordinate system at each step of the computation process. The idea is to choose a coordinate system more appropriate to the shape of $s(t_{j-1}, D_{j-1}, t_j)$, hence reducing the overestimation of the box representation of this set, as illustrated in Figure 13.(b).

An appropriate coordinate system has to be chosen at each step. Assuming that such coordinate systems are given by mean of (invertible) matrices M_j, a naive approach, based on an explicit one-step method, would consist of computing

$$\begin{array}{l} D_j := S(t_{j-1}, M_{j-1}.D'_{j-1}, t_j) \ ; \\ D'_j := M_j^{-1} D_j \end{array}$$

where D'_j and D'_{j-1} are the boxes at t_j and t_{j-1} in their local coordinate system. This approach is naive since it introduces three wrapping effects : in $M_{j-1}D'_{j-1}$ to restore the original coordinate system needed to compute S, in the computation of S, and in the computation of $M_j^{-1} D_j$ to produce the result in the new coordinate system. To remedy this limitation, more advanced techniques (see, for instance, (Lohner, 1987; Stewart, 1971; Davey and Stewart, 1976)) have been proposed but they are all bound to a specific step procedure. For instance, Lohner merges the two naive steps together using a mean value form and use associativity in the matrix products to try eliminating the wrapping effect. More precisely, the key term to be evaluated in his step method

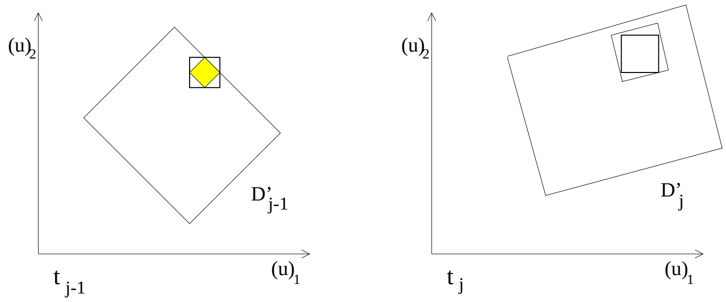

Figure 14. Coordinate transformation on ϵ-boxes

is of the form $(M_j^{-1}JM_{j-1})D'_{j-1}$ and the goal is to choose M_j^{-1} so that $M_j^{-1}JM_{j-1}$ is close to an identity matrix.

Piecewise interval solutions, however, reduce the wrapping effect in the naive method substantially, as illustrated in Figure 14. The overestimations of $M_{j-1}.D'_{j-1}$ and $M_j^{-1}D_j$ on ϵ-boxes introduce wrapping effects that are small compared to the overall size of the box and to the benefits of using piecewise interval extensions. In addition, this reduction of the wrapping effect is not tailored to a specific step method. The basic idea is thus (1) to find a linear approximation of $s(t_{j-1}, M_{j-1}.D'_{j-1}, t_j)$; (2) to compute the matrix M_j^{-1} from the linear relaxation; (3) to apply the naive method on ϵ-boxes. Step (1) can be obtained by using, for instance, a Taylor extension, while Step (2) can use Lohner's method that consists of obtaining a QR factorization of the linear relaxation. Lohner's method has the benefit of being numerically stable.