VALIDATED SOLUTIONS OF INITIAL VALUE PROBLEMS FOR ORDINARY DIFFERENTIAL EQUATIONS

N. S. NEDIALKOV[†§], K. R. JACKSON[†§], AND G. F. CORLISS[‡]

Abstract. Compared to standard numerical methods for initial value problems (IVPs) for ordinary differential equations (ODEs), validated methods for IVPs for ODEs have two important advantages: if they return a solution to a problem, then (1) the problem is guaranteed to have a unique solution, and (2) an enclosure of the true solution is produced. The authors survey Taylor series methods for validated solutions of IVPs for ODEs, describe several such methods in a common framework, and identify areas for future research.

Key words. initial value problems, ordinary differential equations, interval arithmetic, Taylor series methods.

AMS subject classifications. 65L05, 65G10, 65L60.

1. Introduction.

We consider validated numerical methods for the solution of the autonomous initial-value problems (IVPs)

$$y'(t) = f(y), \qquad y(t_0) = y_0, \tag{1.1}$$

where $t \in [t_0, T]$ for some $T > t_0$. Here $t_0, T \in \mathbb{R}$, $f \in C^{k-1}(\mathcal{D})$, $\mathcal{D} \subseteq \mathbb{R}^n$ is an open set, $f : \mathcal{D} \to \mathbb{R}^n$, and $y_0 \in \mathcal{D}$. For expositional convenience, we consider only autonomous systems. This is not a restriction of consequence since a time-dependent system of ODEs can be converted into an autonomous system. Alternatively, the numerical methods discussed here can be extended easily to non-autonomous systems.

Our goal is to compute, at given points $\{t_j\}$ in $(t_0, T]$, an interval vector $[y_j]$ that is guaranteed to contain the true solution $y(t_j)$ of (1.1).

Standard numerical methods for IVPs for ordinary differential equations (ODEs) attempt to compute an approximate solution that satisfies a user-specified tolerance. These methods are usually robust and reliable for most applications, but it is easy to find examples for which they return very inaccurate results. On the other hand, if a validated method (also called an interval method) for IVPs for ODEs returns successfully, it not only produces a guaranteed error bound for the true solution, but also verifies that a unique solution to the problem exists.

There are situations when guaranteed error bounds are desired or needed. For example, a solution with a guaranteed error bound could be used to prove a theorem [39]. Also, some calculations may be critical to the safety or reliability of a system. It may therefore be necessary or desirable to ensure that a computed solution meets a user-specified accuracy requirement.

One reason why validated solutions to IVPs for ODEs have not been popular in the past is that their computation typically requires considerably more time than does the computation of standard methods. However, now that "chips are cheap", it seems natural to shift the burden of determining the reliability of a numerical solution from

[†]Department of Computer Science, University of Toronto, Toronto, Ontario, CANADA M5S 3G4. (ned@cs.toronto.edu, krj@cs.toronto.edu)

[‡]Department of Mathematics, Statistics, and Computer Science, Marquette University, P.O. Box 1881, Milwaukee, WI 53201-1881, USA. (georgec@mscs.mu.edu)

[§]This work was supported in part by the Natural Sciences and Engineering Research Council of Canada and the Information Technology Research Centre of Ontario.

the user to the computer — at least for many standard, computationally inexpensive problems.

In addition, there are situations where interval methods for IVPs for ODEs may not be computationally more expensive than standard methods. For example, many ODEs arising in practical applications contain parameters. Often these parameters cannot be measured exactly, but are known to lie in certain intervals, for example, as in economic models or in robot control problems. In these situations, a user might want to compute solutions for ranges of parameters. For such problems, standard numerical methods are often expensive, but interval methods can easily "capture" all the solutions at essentially no extra cost.

The purpose of this paper is to review the most significant developments in the area of validated solutions of IVPs for ODEs. We focus on the interval methods of Moore [28, 29, 30], Krückeberg [22], Eijgenraam [14], and Lohner [1, 25, 26], all of which are based on Taylor series. One reason for the popularity of this approach is the simple form of the error term for Taylor series methods. In addition, the Taylor series coefficients can be readily generated by automatic differentiation, the order of the method can be changed easily by adding or deleting Taylor series terms, and the stepsize can be changed without recomputing Taylor series coefficients.

Surveys of Taylor series and other interval methods can be found in [3, 7, 31, 35, 37, 40, 41]. These papers give a "high-level" description of existing methods. The main contribution of our paper is that it provides a common framework for the Taylor series methods referenced above, allowing us to compare these schemes with one another in more detail and providing a basis for future work in this area. We also identify the difficulties that arise in validated solutions of IVP for ODEs.

A brief outline of the structure of this paper follows. In §2, we give the background that we need later. We introduce interval-arithmetic operations on real intervals, interval vectors, and interval matrices. We also define interval-valued functions, interval integration, and discuss a method for efficient generation of Taylor series coefficients. In §3, we state in more detail the IVP that is the subject of this paper. In §4, we present an overview of Taylor series methods for validated solutions of IVPs for ODEs. In §5, we examine three techniques for computing an a priori enclosure of the solution and selecting a stepsize: namely, the constant, Taylor series, and polynomial enclosure methods. In §6, we explain the so-called "wrapping effect", consider it in linear systems, and discuss a class of problems where it does not appear. In §7, we describe five algorithms for computing a tight enclosure and reducing the wrapping effect: the algorithms of Moore, Krückeberg, Eijgenraam, Lohner, and an improvement of Lohner's algorithm due to Rihm [38]. In §8, we consider Eijgenraam's stepsize and order control strategies. In §9, we briefly describe a method based on controlling the defect of the computed solution.

2. Preliminaries.

2.1. Interval Arithmetic. The set of intervals on the real line $\mathbb{R}$ is defined by

$$\mathbb{IR} = \{[a] = [\underline{a}, \bar{a}] \mid \underline{a}, \bar{a} \in \mathbb{R},\ \underline{a} \le \bar{a}\}\,.$$

If $\underline{a} = \bar{a}$ then $[a]$ is a *thin* interval; if $\underline{a} \ge 0$ then $[a]$ is *nonnegative* ($[a] \ge 0$); and if $\underline{a} = -\bar{a}$ then $[a]$ is *symmetric*. Two intervals $[a]$ and $[b]$ are equal if $\underline{a} = \underline{b}$ and $\bar{a} = \bar{b}$.

Let $[a], [b] \in \mathbb{IR}$ and $* \in \{+, -, \times, /\}$. The interval-arithmetic operations are defined [30, pp. 8–9] by

$$[a] * [b] = \{x * y \mid x \in [a]\,,\ y \in [b]\}\,, \qquad 0 \notin [b] \text{ when } * = /,$$

which can be written in the equivalent form (we omit "×" in the notation) :

$$\begin{aligned}
[a] + [b] &= \left[\underline{a} + \underline{b}, \bar{a} + \bar{b}\right] \\
[a] - [b] &= \left[\underline{a} - \bar{b}, \bar{a} - \underline{b}\right] \\
[a]\,[b] &= \left[\min\left\{\underline{a}\underline{b}, \underline{a}\bar{b}, \bar{a}\underline{b}, \bar{a}\bar{b}\right\},\ \max\left\{\underline{a}\underline{b}, \underline{a}\bar{b}, \bar{a}\underline{b}, \bar{a}\bar{b}\right\}\right] \\
[a]\,/\,[b] &= [\underline{a}, \bar{a}]\left[1/\bar{b}, 1/\underline{b}\right], \qquad 0 \notin [b].
\end{aligned}$$

With the symbols $\subseteq$ and $\subset$ we denote the usual set inclusion relations. We have an inclusion of intervals

$$[a] \subseteq [b] \iff \underline{a} \geq \underline{b} \text{ and } \bar{a} \leq \bar{b},$$

and a proper inclusion

$$[a] \subset [b] \iff \underline{a} > \underline{b} \text{ and } \bar{a} < \bar{b}.$$

We also define the following quantities for intervals [30]:

- width $\quad w([a]) = \bar{a} - \underline{a}$;
- midpoint $m([a]) = (\bar{a} + \underline{a})/2$;
- magnitude $|[a]| = \max\{|\bar{a}|, |\underline{a}|\}$.

The interval-arithmetic operations are inclusion monotone. That is, for real intervals $[a]$, $[a_1]$, $[b]$, and $[b_1]$ such that $[a] \subseteq [a_1]$ and $[b] \subseteq [b_1]$, we have

$$[a] * [b] \subseteq [a_1] * [b_1], \qquad * \in \{+, -, \times, /\}.$$

The distributive law does not hold in general [2, pp. 3–5]. That is, for any three intervals $[a]$, $[b]$, and $[c]$,

$$[a]\,([b] + [c]) \subseteq [a]\,[b] + [a]\,[c] \qquad \text{(subdistributivity)},$$

with an equality

$$[a]\,([b] + [c]) = [a]\,[b] + [a]\,[c]$$

if, for example, $[b]\,[c] \geq 0$, or $[a]$ is a thin interval, or $[b]$ and $[c]$ are symmetric. Interval addition and multiplication are associative.

For $[a], [b] \in \mathbb{IR}$, [2, pp. 14–17],

$$|[a] + [b]| \leq |[a]| + |[b]|, \tag{2.1.1}$$

$$|[a]\,[b]| = |[a]|\,|[b]|, \tag{2.1.2}$$

$$w([a] \pm [b]) = w([a]) + w([b]), \tag{2.1.3}$$

$$w([a]\,[b]) \geq \max\{|[a]|\,w([b]),\ w([a])\,|[b]|\}, \text{ and} \tag{2.1.4}$$

$$w([a]\,[b]) \leq |[a]|\,w([b]) + w([a])\,|[b]|. \tag{2.1.5}$$

If $[a]$ is symmetric, then

$$w([a]\,[b]) = |[b]|\,w([a]). \tag{2.1.6}$$

From (2.1.4) and (2.1.6), if $[a]$ is a symmetric interval, then

$$w([a]\,[b]) \leq w([a']\,[b]),$$

for any $[a']$ with $w([a']) = w([a])$.

2.2. Interval Vectors and Matrices. By an *interval vector* we mean a vector with interval components. By an *interval matrix* we mean a matrix with interval components. We denote the set of n-dimensional real interval vectors by $\mathbb{IR}^n$, and the set of $n \times m$ real interval matrices by $\mathbb{IR}^{n\times m}$. The arithmetic operations involving interval vectors and matrices are defined by using the same formulas as in the scalar case, except that scalars are replaced by intervals. For example, if $[A] \in \mathbb{IR}^{n\times n}$ has components $[a_{ij}]$, and $[b] \in \mathbb{IR}^n$ has components $[b_k]$, then the components of $[c] = [A]\,[b]$ are given by

$$[c_i] = \sum_{k=1}^{n} [a_{ik}]\,[b_k]\,.$$

An inclusion for interval matrices (and vectors) is defined component-wise by

$$[A] \subseteq [B] \iff [a_{ij}] \subseteq [b_{ij}] \qquad \text{(for all } i, j),$$

and a proper inclusion by

$$[A] \subset [B] \iff [a_{ij}] \subset [b_{ij}] \qquad \text{(for all } i, j).$$

The *maximum norm* of an interval vector $[a] \in \mathbb{IR}^n$ is given by

$$\|[a]\| = \max_{1\le i\le n} \{|[a_i]|\}\,,$$

and of a matrix $[A]$ by

$$\|[A]\| = \max_{1\le i\le n} \sum_{j=1}^{n} |[a_{ij}]|\,.$$

We also use the symbol $\|\cdot\|$ to denote the maximum norm of scalar vectors, scalar matrices, and functions.

Let $\mathcal{A}, \mathcal{B} \in \mathbb{R}^n$ be compact non-empty sets. With $q\,(\mathcal{A}, \mathcal{B})$ we denote the Hausdorff distance between $\mathcal{A}$ and $\mathcal{B}$:

$$q\,(\mathcal{A}, \mathcal{B}) = \max\left\{ \max_{x\in\mathcal{A}} \min_{y\in\mathcal{B}} \|x-y\|,\ \max_{y\in\mathcal{B}} \min_{x\in\mathcal{A}} \|x-y\| \right\}. \tag{2.2.1}$$

The distance between two intervals $[a]$ and $[b]$ is

$$q\,([a],[b]) = \max\left\{ |\underline{a} - \underline{b}|, |\bar{a} - \bar{b}| \right\}, \tag{2.2.2}$$

and the distance between two interval vectors $[u]$, $[v] \in \mathbb{IR}^n$ is

$$q\,([u],[v]) = \max_{1\le i\le n} \{q\,([u_i],[v_i])\}\,. \tag{2.2.3}$$

Let $[A] \in \mathbb{IR}^{n\times m}$. We define the following quantities component-wise for interval matrices (and vectors):

- width

$$w\,([A]) = \begin{pmatrix} w\,([a_{11}]) & \dots & w\,([a_{1m}]) \\ \vdots & & \vdots \\ w\,([a_{n1}]) & \dots & w\,([a_{nm}]) \end{pmatrix};$$

- midpoint

$$m([A]) = \begin{pmatrix} m([a_{11}]) & \dots & m([a_{1m}]) \\ \vdots & & \vdots \\ m([a_{n1}]) & \dots & m([a_{nm}]) \end{pmatrix};$$

- magnitude

$$|[A]| = \begin{pmatrix} |[a_{11}]| & \dots & |[a_{1m}]| \\ \vdots & & \vdots \\ |[a_{n1}]| & \dots & |[a_{nm}]| \end{pmatrix}.$$

Addition of interval matrices is associative, but multiplication of interval matrices is not associative in general [32, pp. 80–81]. Also, the distributive law does not hold in general for interval matrices, [32, p. 79]. That is, if $[A] \in \mathbb{IR}^{n\times m}$ and $[B], [C] \in \mathbb{IR}^{m\times p}$, then

$$[A]([B]+[C]) \subseteq [A][B]+[A][C] \quad \text{(subdistributivity)}, \tag{2.2.4}$$

with an equality

$$[A]([B]+[C]) = [A][B]+[A][C],$$

if, for example, $[b_{ij}][c_{ij}] \geq 0$, or $[A]$ is a point matrix, or all components of $[B]$ and $[C]$ are symmetric intervals.

Let $[A], [B] \in \mathbb{IR}^{n\times n}$. Then [2, pp. 125–126]

$$|[A]+[B]| \leq |[A]| + |[B]|, \tag{2.2.5}$$

$$|[A][B]| \leq |[A]|\,|[B]|, \tag{2.2.6}$$

$$w([A] \pm [B]) = w([A]) + w([B]), \tag{2.2.7}$$

$$w([A][B]) \geq \max\{|[A]|\,w([B]),\ w([A])\,|[B]|\}, \text{ and} \tag{2.2.8}$$

$$w([A][B]) \leq |[A]|\,w([B]) + w([A])\,|[B]|. \tag{2.2.9}$$

Let the components of $[B]$ be symmetric intervals. Then

$$\begin{aligned} w([A][B]) &= |[A]|\,w([B]) \text{ and} \\ w([A][B]) &\leq w([A][B']) \end{aligned} \tag{2.2.10}$$

for any $[B']$ with $w([B']) = w([B])$.

Let $[c] \in \mathbb{IR}^n$ be a symmetric vector (all components of $[c]$ are symmetric intervals). Then

$$\begin{aligned} w(([A][B])[c])) &= |[A][B]|\,w([c]) \\ &\leq |[A]|\,|[B]|\,w([c]) = w([A]([B][c])). \end{aligned} \tag{2.2.11}$$

Throughout this paper, we assume *exact* real interval arithmetic, as described in this subsection. In an implementation in floating-point arithmetic, if one or both endpoints of a real interval are not representable, then they must be rounded out-ward to the closest representable floating-point numbers. Moore [30, pp. 1–13] describes how interval arithmetic can be implemented. It is often called a machine, or rounded, interval arithmetic. A discussion of its properties can be found in [23].

2.3. Interval-Valued Functions. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function on $\mathcal{D} \subseteq \mathbb{R}^n$. We consider functions whose representations contain only a finite number of constants, variables, arithmetic operations, and standard functions (sin, cos, log, exp, etc.).

We define the *range* of f over an interval vector $[a] \subseteq \mathcal{D}$ by

$$R(f;[a]) = \{f(x) \mid x \in [a]\}.$$

The *interval-arithmetic evaluation* of f on $[a]$, which we denote by $f([a])$, is obtained by replacing each occurrence of a real variable with the corresponding interval, by replacing the standard functions with enclosures of their ranges, and by performing interval-arithmetic operations instead of the real operations. In practice, $f([a])$ is not unique, because it depends on how f is evaluated in interval arithmetic. For example, expressions that are mathematically equivalent for scalars, such as $x(y+z)$ and $xy + xz$, may have different values if x, y, and z are intervals. However, since we are interested in the interval-arithmetic evaluation of f on a computer, we can assume that $f([a])$ is uniquely determined by the code list of f. No matter how $f([a])$ is evaluated, it follows from the inclusion monotone property of the the interval operations that

$$R(f;[a]) \subseteq f([a]).$$

It f satisfies a Lipschitz condition on $\mathcal{D} \subseteq \mathbb{R}^n$, then for any $[a] \subseteq \mathcal{D}$,

$$q\left(R(f;[a]), f([a])\right) \leq c_1 \|w([a])\|$$

for some constant $c_1 \geq 0$ independent of $[a]$, where $q(\cdot,\cdot)$ is defined by (2.2.2), [30, p. 24], [2].

Mean-value form. If $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable on $\mathcal{D} \subseteq \mathbb{R}^n$, then

$$f(y) \in f_M([a]) := f(b) + f'([a])([a] - b), \tag{2.3.1}$$

for any $y, b \in [a]$, and $[a] \subseteq \mathcal{D}$, [30, p. 47]. The expression $f(b) + f'([a])([a] - b)$ is called the mean-value form of f. Here, f' is any way of evaluating the gradient of f. Mathematically, f_M is not uniquely defined, but it is uniquely determined by the code list of f'. If, in addition, f' is Lipschitz on $\mathcal{D}$, then for any $[a] \subseteq \mathcal{D}$,

$$q\left(R(f;[a]), f_M([a])\right) \leq c_2 \|w([a])\|^2$$

for some constant $c_2 \geq 0$ independent of $[a]$, [32, pp. 55–56]. Therefore, the mean-value evaluation is *quadratically convergent* in the sense that the distance between $R(f;[a])$ and $f_M([a])$ tends to zero as the square of $\|w([a])\|$, as $\|w([a])\|$ tends to zero.

Similar results apply to functions from $\mathbb{R}^n$ to $\mathbb{R}^n$.

2.4. Integration. Let $f : \mathcal{D} \to \mathbb{R}^n$ be a continuous function on $\mathcal{D} \subseteq \mathbb{R}$ and $[a] \subseteq \mathcal{D}$. Then the components of $\int_{\underline{a}}^{\bar{a}} f(t)dt$ are

$$\left\{\int_{\underline{a}}^{\bar{a}} f(t)dt\right\}_i = \int_{\underline{a}}^{\bar{a}} \{f(t)\}_i \, dt,$$

where $\{\cdot\}_i$ denotes the i-th component of a vector. Obviously, [30, pp. 70–71],

$$\int_{\underline{a}}^{\bar{a}} f(t)dt \in (\bar{a} - \underline{a})f([a]) = w([a])\, f([a]). \tag{2.4.1}$$

2.5. Automatic Generation of Taylor Coefficients. Moore [30, pp. 107–130] presents a method for efficient generation of Taylor coefficients. Rall [36] describes in detail algorithms for automatic differentiation and generation of Taylor coefficients. He also considers different applications of automatic differentiation, including applications to ordinary differential equations. Two books containing papers and extensive bibliographies on automatic differentiation are [4] and [17].

Since we shall need point and interval Taylor coefficients, we briefly describe the basic idea of the method. Denote the i-th Taylor coefficient of $u(t)$ evaluated at some point t_j by

$$(u_j)_i = \frac{u^{(i)}(t_j)}{i!},$$

where $u^{(i)}(t)$ is the i-th derivative of $u(t)$. Let $(u_j)_i$ and $(v_j)_i$ be the i-th Taylor coefficients of $u(t)$ and $v(t)$ at t_j. It can be shown that

$$(u_j \pm v_j)_i = (u_j)_i \pm (v_j)_i, \tag{2.5.1}$$

$$(u_j v_j)_i = \sum_{r=0}^{i} (u_j)_r (v_j)_{i-r}, \text{ and} \tag{2.5.2}$$

$$\left(\frac{u_j}{v_j}\right)_i = \frac{1}{v_j}\left\{(u_j)_i - \sum_{r=1}^{i} (v_j)_r \left(\frac{u_j}{v_j}\right)_{i-r}\right\}. \tag{2.5.3}$$

Similar formulas can be derived for the generation of Taylor coefficients for the standard functions [30, p. 114].

Consider the autonomous differential system

$$y'(t) = f(y), \qquad y(t_j) = y_j. \tag{2.5.4}$$

We introduce the sequence of functions

$$f^{[1]} = f, \tag{2.5.5}$$

$$f^{[i]} = \frac{1}{i}\frac{\partial f^{[i-1]}}{\partial y} f, \qquad \text{for } i \geq 2. \tag{2.5.6}$$

Using (2.5.4–2.5.6), the Taylor coefficients of $y(t)$ at t_j satisfy

$$(y_j)_0 = y_j, \tag{2.5.7}$$

$$(y_j)_1 = f^{[1]}(y_j) = f(y_j), \text{ and} \tag{2.5.8}$$

$$(y_j)_i = f^{[i]}(y_j) = \frac{1}{i}\left(\frac{\partial f^{[i-1]}}{\partial y} f\right)(y_j), \qquad \text{for } i \geq 2. \tag{2.5.9}$$

By using (2.5.1–2.5.3), similar formulas for the Taylor coefficients of the standard functions, and (2.5.9), we can recursively evaluate $(y_j)_i$, for $i \geq 2$. Moore shows [30, pp. 111-112] that if the number of the arithmetic operations in the code list of f is N, then the number of arithmetic operation required for the generation of K Taylor coefficients is between $N \times K$ and $N \times K(K+1)/2$, depending on the ratio of additions, multiplications, and divisions in the code list for f.

Let $y(t_j) = y_j \in [y_j]$. If we have a procedure for computing the point Taylor coefficients of $y(t)$ and perform the computations in interval arithmetic with $[y_j]$

instead of y_j, we obtain a procedure for computing the interval Taylor coefficients of $y(t)$. If we denote the i-th interval Taylor coefficient of $y(t)$ at t_j by $[y_j]_i$, then we have

$$\begin{aligned}
[y_j]_0 &= [y_j]\,,\\
[y_j]_1 &= f^{[1]}([y_j]) = f([y_j]), \text{ and}\\
[y_j]_i &= f^{[i]}([y_j]) = \frac{1}{i}\left(\frac{\partial f^{[i-1]}}{\partial y} f\right)([y_j]), \qquad \text{for } i \geq 2.
\end{aligned}$$

3. The Initial Value Problem. We consider the set of autonomous initial value problems

$$y'(t) = f(y) \tag{3.1}$$

$$y(t_0) \in [y_0], \tag{3.2}$$

where $t \in [t_0, T]$ for some $T > t_0$. Here $t_0, T \in \mathbb{R}$, $f \in C^{k-1}(\mathcal{D})$, $\mathcal{D} \subseteq \mathbb{R}^n$ is open, $f : \mathcal{D} \to \mathbb{R}^n$, and $[y_0] \subseteq \mathcal{D}$. This generalization of (1.1) permits the initial value $y(t_0)$ to be in an interval, rather than specifying a particular value. We assume that the representation of f contains only a finite number of constants, variables, elementary operations, and standard functions. Since we assume $f \in C^{k-1}(\mathcal{D})$, we exclude functions that contain, for example, branches, abs, or min. As noted earlier, for expositional convenience, we consider only autonomous systems, but this is not a restriction of consequence. We consider a grid $t_0 < t_1 < \cdots < t_m = T$, which is not necessarily equally spaced, and denote the stepsize from t_j to t_{j+1} by h_j ($h_j = t_{j+1} - t_j$). The step from t_j to t_{j+1} is referred to as the $(j+1)$-st step. We denote the solution of (3.1) with an initial condition y_j at t_j by $y(t; t_j, y_j)$. By $y(t; t_j, [y_j])$, we denote the set of solutions

$$y(t; t_j, [y_j]) = \{ y(t; t_j, y_j) \mid y_j \in [y_j] \}\,. \tag{3.3}$$

An enclosure of the solution of (3.1–3.2) at $t_j > t_0$ is an interval vector $[y_j]$ such that

$$y(t_j; t_0, [y_0]) \subseteq [y_j]\,.$$

Our goal is to compute tight enclosures of the solution at $t_1, t_2, \ldots, t_m$.

4. A Brief Overview of Taylor Series Methods for the Validated Solution of IVPs for ODEs. In most validated methods for IVPs for ODEs, each step consists of two phases:

Algorithm I. Compute a stepsize h_j and an a priori enclosure $[\tilde{y}_j]$ of the solution such that $y(t; t_j, y_j)$ is guaranteed to exist for all $t \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$, and

$$y(t; t_j, [y_j]) \subseteq [\tilde{y}_j] \qquad \text{for all } t \in [t_j, t_{j+1}]\,.$$

Then, for $t \in [t_j, t_{j+1})$,

$$y(t; t_0, [y_0]) \subseteq [y(t)] := [y_j] + \sum_{i=1}^{k-1} [y_j]_i \,(t - t_j)^i + [\tilde{y}_j]_k (t - t_j)^k.$$

Algorithm II. Using $[\tilde{y}_j]$, compute a tighter enclosure $[y_{j+1}]$ of $y(t_{j+1}; t_j, [y_j])$.

This two-phase process is similar to the predictor-corrector procedure of some approximate algorithms.

Usually the algorithm to validate the existence of a unique solution uses the Picard-Lindelöf operator and the Banach fixed-point theorem. In Taylor series methods, the computation of a tighter enclosure is based on Taylor series plus remainder, the mean-value theorem, and various interval transformations.

In Algorithm I, it is important to take as a large a step as possible. Krückeberg [22], Eijgenraam [14], and Lohner [26] use constant a priori enclosures in their algorithms. A disadvantage of the constant a priori approach is that the stepsize is restricted to Euler steps, independently of the order of the Taylor series used in the second algorithm. Moore uses Taylor series for validation, and recently, Lohner [27] proposed a polynomial enclosure method. In both methods, the stepsize can increase as the order of the series increases. We discuss these aspects of Algorithm I in more detail in §5.

The main difficulty in Algorithm II is how to reduce the "wrapping effect." This phenomenon appears when a set, which is not representable exactly by an interval vector, has to be enclosed in an interval vector. Since wrapping can occur during each integration step, the widths of the intervals can become unacceptably large. Moore proposes a local coordinate transformation method to reduce the wrapping effect. Krückeberg represents the enclosure of the solution at each step as a linear transformation of an interval vector. He gives two examples, for which he computes good bounds. However, Jackson [20] shows that both algorithms can produce an exponential growth of the widths of the enclosures. Lohner also represents the enclosure of the solution as a linear transformation of an interval vector. He proposes a QR-factorization method, which shows the most promising behavior. Rihm [38] suggests an extended mean-value method, which is similar to Lohner's second algorithm. He shows that, under some conditions, the extended-mean value method produces better enclosures than either Eijgenraam's or Lohner's method. We discuss in more detail the wrapping effect in §6 and the algorithms of Moore, Krückeberg, Eijgenraam, Lohner, and the extended mean-value method in §7.

The most successful software for the validated solution of IVPs for ODEs is Lohner's <u>A</u>nfangs<u>w</u>ert<u>a</u>ufgabe (AWA) package [26]. It does not automatically select the stepsize or order: they have to be specified by the user and cannot be changed during execution. As Stetter [41] points out, AWA is often computationally expensive. We discuss stepsize and order control in more detail in §8.

5. Algorithm I: Computing an A Priori Enclosure of the Solution and Selecting a Stepsize. Suppose that at t_j we have a tight enclosure $[y_j]$ of $y(t_j; t_0, [y_0])$. In this section, we consider the problem: find a stepsize $h_j > 0$ and an a priori enclosure $[\tilde{y}(t)]$ such that for any $y_j \in [y_j]$

$$y'(t) = f(y), \qquad y(t_j) = y_j \tag{5.1}$$

has a unique solution $y(t) \in [\tilde{y}(t)]$ for $t \in [t_j, t_{j+1}]$. The methods which we describe for computing h_j and $[\tilde{y}(t)]$ are based on the application of the Picard-Lindelöf operator

$$(Ty)(t) = y_j + \int_{t_j}^{t} f(y(s))ds \tag{5.2}$$

to appropriate sets of functions and the Banach fixed-point theorem.

THEOREM 5.1. Banach fixed-point theorem. *Let $\Phi : Y \to Y$ be defined on a complete metric space Y with a metric $d(\cdot,\cdot)$. Let γ satisfy $0 \le \gamma < 1$, and let*

$$d(\Phi(x), \Phi(y)) \le \gamma d(x, y) \tag{5.3}$$

for all x and $y \in Y$. Then Φ has a unique fixed-point $y^* \in Y$.

The exponential norm of a function $u(t)$ from the set of continuous functions in $[t_j, t_{j+1}]$, $u \in C^0[t_j, t_{j+1}]$, is defined by

$$\|u\|_\alpha = \max_{t\in[t_j,t_{j+1}]} \left(e^{-\alpha(t-t_j)}\|u(t)\|\right),$$

where $\alpha > 0$. If f is continuously differentiable, then there exists $\alpha > 0$ such that the Picard-Lindelöf operator satisfies (5.3) with the metric being the exponential norm with this α, [14, pp. 66–67], [26, pp. 27–29]. Therefore, if Y is a complete metric space with the exponential norm and $TY \subseteq Y$, then T has a unique fixed-point in Y.

5.1. Constant Enclosure. We extend the $C^0(\mathcal{X})$ notation used above to $C^0(\mathcal{X}, \mathcal{Y})$, the set of continuous functions from $\mathcal{X} \subseteq \mathbb{R}$ to $\mathcal{Y} \subseteq \mathbb{R}^n$.

Let $[\tilde{y}_j^0]$ be a superset of the initial condition $[y_j]$: $[y_j] \subset [\tilde{y}_j^0]$.[1] Consider the set of continuous functions on $[t_j, t_{j+1}]$ with ranges in $[\tilde{y}_j^0]$,

$$Y = \left\{u \mid u \in C^0\left([t_j, t_{j+1}], [\tilde{y}_j^0]\right)\right\}, \tag{5.1.1}$$

with the exponential norm. By applying the Picard-Lindelöf operator (5.2) to a function $y^0 \in Y$ and using (2.4.1), we obtain

$$\begin{aligned}(Ty^0)(t) &= y_j + \int_{t_j}^t f(y^0(s))ds \\ &\in y_j + \int_{t_j}^t f([\tilde{y}_j^0])ds \\ &\subseteq y_j + [0, h_j]f([\tilde{y}_j^0]).\end{aligned}$$

The operator T has a unique fixed point in Y if $TY \subseteq Y$, since Y is a closed bounded subset of C^0, T is a contraction in the exponential norm (f is continuously differentiable by assumption), and $(Ty^0)(t)$ is continuous for any $y^0 \in Y$. This fixed point is a solution of (5.1) for $t \in [t_j, t_{j+1}]$. Therefore, if the inclusion

$$[\tilde{y}_j^1] := [y_j] + [0, h_j]f\left([\tilde{y}_j^0]\right) \subseteq [\tilde{y}_j^0] \tag{5.1.2}$$

holds, then $TY \subseteq Y$, and (5.1) has a unique solution $y(t; t_j, y_j)$ that satisfies

$$y(t; t_j, y_j) \in [y_j] + [0, h_j]f([\tilde{y}_j^0]) = [\tilde{y}_j^1]$$

for all $t \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$. Furthermore, since $f([\tilde{y}_j^1]) \subseteq f([\tilde{y}_j^0])$, we have

$$y(t; t_j, y_j) \in [y_j] + [0, h_j]f([\tilde{y}_j^1])$$

for all $t \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$. The stepsize, h_j, can always be taken small enough such that the enclosure in (5.1.2) holds.

The inclusion in (5.1.2) can be easily verified. However, a serious disadvantage of the method is that the stepsize is restricted to Euler steps, even when high-order methods are used to tighten the a priori enclosure in Algorithm II.

Eijgenraam [14, pp. 59–67] and Lohner [26, pp. 27–31] use a constant a priori enclosure in their algorithms.

[1] We use superscripts on vectors to indicate different vectors, not powers of vectors.

5.2. Polynomial Enclosure. Lohner [27] proposes a polynomial enclosure method in which the stepsize can be larger when the order of the Taylor series increases. He applies the Picard-Lindelöf operator to the set of continuous functions

$$Y = \Big\{ g \mid g(t) = \sum_{i=0}^{k-1} (y_j)_i \, (t-t_j)^i + g_{j,k}(t)(t-t_j)^k \\ t \in [t_j, t_{j+1}] \, , \ y_j \in [y_j] \, , \ g_{j,k} \in C^0 \left([t_j, t_{j+1}], [\tilde{y}^0_{j,k}] \right) \Big\}.$$

As in the previous subsection, to prove that (5.1) has a unique solution $y(t; t_j, y_j)$ for $t \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$, it is sufficient to verify the inclusion $TY \subseteq Y$. We shall give the motivation for the validation procedure used in [27] and then describe the procedure itself.

Let

$$u(t, \hat{u}) = \sum_{i=0}^{k-1} (y_j)_i \, (t-t_j)^i + \hat{u}(t-t_j)^k \tag{5.2.1}$$

for $t \in [t_j, t_{j+1}]$, where $y_j \in [y_j]$ and constant $\hat{u} \in [\tilde{y}^0_{j,k}]$. Since

$$\frac{d}{ds^i} \, f(u(s, \hat{u}))\big|_{t_j} = (i+1)! \, (y_j)_{i+1} \qquad \text{for } i = 1, \ldots, k-1,$$

we obtain the Taylor series expansion

(5.2.2)

$$\begin{aligned} f(u(s, \hat{u})) &= f(u(t_j, \hat{u})) + \sum_{i=1}^{k-1} \frac{(s-t_j)^i}{i!} \frac{d}{ds^i} \, f(u(s, \hat{u}))\big|_{t_j} + \left(f\left(u(s, \hat{u}); t_j, s\right)\right)_k (s-t_j)^k \\ &= \sum_{i=0}^{k-1} (i+1) \, (y_j)_{i+1} \, (s-t_j)^i + \left(f\left(u(s, \hat{u}); t_j, s\right)\right)_k (s-t_j)^k, \end{aligned}$$

where $\left(f\left(u(s, \hat{u}); t_j, s\right)\right)_k$ is the k-th Taylor coefficient of $f(u(s, \hat{u}))$ with its l-th component evaluated at some $\xi_{jl} \in [t_j, s]$. If $[f_{j,k}]$ satisfies

$$\left(f\left(u(s, \hat{u}); t_j, s\right)\right)_k \in [f_{j,k}] \tag{5.2.3}$$

for $s \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$ and $\hat{u} \in [\tilde{y}^0_{j,k}]$, then

$$f(u(s, \hat{u})) \in \sum_{i=0}^{k-1} (i+1) \, (y_j)_{i+1} \, (s-t_j)^i + [f_{j,k}] \, (s-t_j)^k.$$

An arbitrary member v of Y has the form

$$v(s) = \sum_{i=0}^{k-1} (y_j)_i \, (s-t_j)^i + v_{j,k}(s)(s-t_j)^k \tag{5.2.4}$$

for $s \in [t_j, t_{j+1}]$, where $v_{j,k}(s) \in C^0([t_j, t_{j+1}], [\tilde{y}^0_{j,k}])$. Consider the l-th component of $f(v(s))$. There exist vectors $u_{l*}, u_l^* \in [\tilde{y}^0_{j,k}]$ such that

$$f_l \left(v(s) \right) \in \left[f_l \left(u(s, u_{l*}) \right), f_l \left(u(s, u_l^*) \right) \right].$$

From (5.2.2) and (5.2.3),

$$[f(u(s,u_{l*})), f(u(s,u_l^*))] \subseteq \sum_{i=0}^{k-1}(i+1)(y_j)_{i+1}(s-t_j)^i + [f_{j,k}](s-t_j)^k.$$

Therefore,

$$f(v(s)) \in \sum_{i=0}^{k-1}(i+1)(y_j)_{i+1}(s-t_j)^i + [f_{j,k}](s-t_j)^k. \tag{5.2.5}$$

The Picard-Lindelöf operator maps v into a continuous function: $w = Tv$. From (5.2.4) and (5.2.5), we derive

$$\begin{aligned} w(t) &:= (Tv)(t) \\ &= y_j + \int_{t_j}^{t} f(v(s))ds \\ &\in y_j + \sum_{i=0}^{k-1}(y_j)_{i+1}(t-t_j)^{i+1} + \frac{[f_{j,k}]}{k+1}(t-t_j)^{k+1} \\ &= \sum_{i=0}^{k-1}(y_j)_i(t-t_j)^i + \left((y_j)_k + \frac{[f_{j,k}]}{k+1}(t-t_j)\right)(t-t_j)^k \\ &\subseteq \sum_{i=0}^{k-1}(y_j)_i(t-t_j)^i + \left([y_j]_k + \frac{[f_{j,k}]}{k+1}[0,h_j]\right)(t-t_j)^k. \end{aligned} \tag{5.2.6}$$

From (5.2.6), if

$$[\tilde{y}_{j,k}^1] := [y_j]_k + \frac{[f_{j,k}]}{k+1}[0,h_j] \subseteq [\tilde{y}_{j,k}^0],$$

then

$$w(t) \in \sum_{i=0}^{k-1}(y_j)_i(t-t_j)^i + [\tilde{y}_{j,k}^1](t-t_j)^k \tag{5.2.7}$$

$$\subseteq \sum_{i=0}^{k-1}(y_j)_i(t-t_j)^i + [\tilde{y}_{j,k}^0](t-t_j)^k. \tag{5.2.8}$$

for all $t \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$. Since $w(t)$ is continuous and satisfies (5.2.7–5.2.8), $w(t)$ has the form

$$w(t) = \sum_{i=0}^{k-1}(y_j)_i(t-t_j)^i + w_{j,k}(t)(t-t_j)^k,$$

where $w_{j,k}(t) \in C^0([t_j,t_{j+1}],[\tilde{y}_{j,k}^1])$. Therefore, if $[\tilde{y}_{j,k}^1] \subseteq [\tilde{y}_{j,k}^0]$, then $w(t) \in Y$. Consequently, $TY \subseteq Y$, from which it follows that there exists a unique solution $y(t;t_j,y_j)$ to (5.1) for $t \in [t_j,t_{j+1}]$ and all $y_j \in [y_j]$. Moreover,

$$y(t;t_j,y_j) \in [\tilde{y}(t)] := \sum_{i=0}^{k-1}[y_j]_i(t-t_j)^i + [\tilde{y}_{j,k}^1](t-t_j)^k.$$

The enclosure $[f_{j,k}]$ in (5.2.3) cannot be computed by generating the Taylor series for the solution of the ODE. Lohner [27] gives recursive formulas for the computation of $[\tilde{y}^1_{j,k}]$. These formulas are based on the polynomial interval machine arithmetic given by Eiermann [13].

5.3. Taylor Series Enclosure. Moore's [30, pp. 100–103] method validates the existence of a unique solution and computes an enclosure in one process. Consider the interval Taylor series

$$U\left(t, [\tilde{y}^0_j]\right) = [y_j] + \sum_{i=1}^{k-1} [y_j]_i \, (t - t_j)^i + f^{[k]}([\tilde{y}^0_j])(t - t_j)^k, \tag{5.3.1}$$

where $[y_j] \subset [\tilde{y}^0_j]$. If

$$\begin{aligned}[\tilde{y}^1_j] &:= U\left([t_j, t_{j+1}], [\tilde{y}^0_j]\right) \\ &= [y_j] + \sum_{i=1}^{k-1} [y_j]_i \, [0, h_j]^i + f^{[k]}([\tilde{y}^0_j])[0, h^k_j] \subseteq [\tilde{y}^0_j],\end{aligned} \tag{5.3.2}$$

then there exists a unique solution $y(t; t_j, y_j)$ to (5.1) for $t \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$ that satisfies $y(t; t_j, y_j) \in U\left(t, [\tilde{y}^0_j]\right)$. In addition, since

$$U\left([t_j, t_{j+1}], [\tilde{y}^1_j]\right) \subseteq U\left([t_j, t_{j+1}], [\tilde{y}^0_j]\right) = [\tilde{y}^1_j],$$

it follows that $y(t; t_j, y_j) \in U\left(t, [\tilde{y}^1_j]\right)$. The interval-valued function $U\left(t, [\tilde{y}^1_j]\right)$ is used as an enclosure of $y(t; t_j, [y_j])$ for $t \in [t_j, t_{j+1}]$.

Moore does not give a proof of his validation procedure, but it can be proved by using arguments similar to those in the previous subsection [12]. We now show how h_j and $[\tilde{y}^0_j]$ are computed such that (5.3.2) holds.

Computing the stepsize. We want $\left\| w\left(U\left(t, [\tilde{y}^0_j]\right)\right) \right\|$ to be small relative to $\left\|U\left(t, [\tilde{y}^0_j]\right)\right\|$ for $t \in [t_j, t_{j+1}]$. That is, we require that

$$\frac{\left\| w\left(U\left(t, [\tilde{y}^0_j]\right)\right) \right\|}{\|U(t, [\tilde{y}^0_j])\|} \approx \epsilon \tag{5.3.3}$$

for some small $\epsilon > 0$. To simplify the discussion, we assume that $\|w([y_j])\| = 0$. Then we obtain from (5.3.1) and (5.3.3)

$$\left\| w\left(U\left(t, [\tilde{y}^0_j]\right)\right) \right\| = \|w(f^{[k]}([\tilde{y}^0_j]))\| h^k_j \approx \epsilon \left\|U\left(t, [\tilde{y}^0_j]\right)\right\|. \tag{5.3.4}$$

Approximating $\|w(f^{[k]}([\tilde{y}^0_j]))\|$. If we assume that $\left[\tilde{y}^0_j\right]$ is the smallest interval containing $y(t; t_j, [y_j])$ for all $t \in [t_j, t_{j+1}]$, then from

$$\begin{aligned} w\left(f^{[k]}_l([\tilde{y}^0_j])\right) &\geq w\left(R\left(f^{[k]}_l; [\tilde{y}^0_j]\right)\right) \\ &= f^{[k]}_l(y(\alpha_{jl})) - f^{[k]}_l(y(\beta_{jl})) \\ &= \left| f^{[k]}_l(y(\alpha_{jl})) - f^{[k]}_l(y(\beta_{jl})) \right| \\ &\geq \left| f^{[k]}_l(y(t_j)) - f^{[k]}_l(y(t_{j+1})) \right| \\ &= \left| (k+1) f^{[k+1]}_l(y(\xi_{jl})) \right| h_j, \end{aligned}$$

where f_l is the l-th component of f, and $\xi_{jl}, \alpha_{jl}, \beta_{jl} \in [t_j, t_{j+1}]$ $(l = 1, \dots, n)$. We approximate $\left\| w\left(f^{[k]}([\tilde{y}_j^0])\right) \right\|$ by

$$\|w(f^{[k]}([\tilde{y}_j^0]))\| \approx (k+1)\|f^{[k+1]}(y_j)\|h_j. \tag{5.3.5}$$

Approximating $\|U\left(t, [\tilde{y}_j^0]\right)\|$. We obtain from (5.3.2)

$$w\left([\tilde{y}_j^0]\right) \geq w\left(U\left([t_j, t_{j+1}], [\tilde{y}_j^0]\right)\right) \geq w\left(f(y_j)\,[0, h_j]\right) = h_j\,|f(y_j)|\,. \tag{5.3.6}$$

Using (5.3.1) and (5.3.6), we approximate $\|U\left(t, [\tilde{y}_j^0]\right)\|$ on $[t_j, t_{j+1}]$ by

$$\|U\left(t, [\tilde{y}_j^0]\right)\| \approx \max\left\{\|y_j\|, h_j\|f(y_j)\|\right\}. \tag{5.3.7}$$

An approximation for the stepsize is determined from (5.3.4), (5.3.5), and (5.3.7),

$$h_j \approx \begin{cases} \left(\frac{\epsilon\|y_j\|}{(k+1)\|f^{[k+1]}(y_j)\|}\right)^{\frac{1}{k+1}} & \text{if} \quad \|y_j\| \geq h_j\|f(y_j)\| \\ \left(\frac{\epsilon\|f(y_j)\|}{(k+1)\|f^{[k+1]}(y_j)\|}\right)^{\frac{1}{k}} & \quad \text{otherwise.} \end{cases} \tag{5.3.8}$$

Moore suggests that the interval $[\tilde{y}_j^0]$ be computed by the formula

$$[\tilde{y}_j^0] = [y_j] + f([y_j])[0, h_j], \tag{5.3.9}$$

and then the inclusion $[\tilde{y}_j^1] \subseteq [\tilde{y}_j^0]$ is verified. This test will almost always fail, because usually

$$[\tilde{y}_j^1] = [\tilde{y}_j^0] + \sum_{i=2}^{k-1} [y_j]_i\,[0, h_j]^i + f^{[k]}([\tilde{y}_j^0])[0, h_j^k]$$

is not contained in $\left[\tilde{y}_j^0\right]$.

Finally, Moore's method can be summarized in the following algorithm:

Algorithm 5.3.1

Input:

$[y_j]$, ϵ.

Compute:

1 h_j from (5.3.8) and $[\tilde{y}_j^0]$ from (5.3.9);
2 $[\tilde{y}_j^1] := U\left([t_j, t_{j+1}], [\tilde{y}_j^0]\right)$;
3 **if** $[\tilde{y}_j^1] \subseteq [\tilde{y}_j^0]$ **then**
4 $\quad [\tilde{y}_j^1] := U\left([t_j, t_{j+1}], [\tilde{y}_j^1]\right)$;
5 **else**
6 $\quad [\tilde{y}_j^0] := [\tilde{y}_j^1]$;
7 $\quad [\tilde{y}_j^1] := U\left([t_j, t_{j+1}], [\tilde{y}_j^1]\right)$;
8 $\quad$ **if** $[\tilde{y}_j^1] \subseteq [\tilde{y}_j^0]$ **then**
9 $\quad\quad [\tilde{y}_j^1] := U\left([t_j, t_{j+1}], [\tilde{y}_j^1]\right)$;
10 $\quad$ **else**
11 $\quad\quad h_j := h_j/2$;
12 $\quad\quad$ **goto** 2;
13 $\quad$ **end-if**
14 **end-if**

Output:

h_j, $[\tilde{y}_j^1]$.

If the test in step 3 is not successful, the algorithm tries to verify

$$U\left([t_j, t_{j+1}], [\tilde{y}_j^1]\right) \subseteq [\tilde{y}_j^1]$$

in step 8 with $[\tilde{y}_j^1]$ computed in step 2. If the inclusion still does not hold, then the stepsize is halved.

The main advantage of the method is that the stepsize can be bigger than in the constant enclosure method. Also, the validation and computation of a final enclosure are carried out in one process.

6. The Wrapping Effect. Moore [30] noticed a difficulty arising in his method, the so-called *wrapping effect*. It can be illustrated by his example,

$$y_1' = y_2 \qquad y_2' = -y_1. \tag{6.1}$$

The solution of (6.1) with an initial condition y_0 is given by $y(t) = A(t)y_0$, where

$$A(t) = \begin{pmatrix} \cos t & \sin t \\ -\sin t & \cos t \end{pmatrix}.$$

Let $y_0 \in [y_0]$. The interval vector $[y_0] \in \mathbb{IR}^2$ can be viewed as a rectangle in the (y_1, y_2) plane. At $t_1 > t_0$, $[y_0]$ will be mapped by $A(t_1)$ into a rectangle of the same size, as shown in Fig. 6.1. If we want to enclose this rectangle in an interval vector, we have to wrap it by another rectangle with sides parallel to the y_1 and y_2 axes. This larger rectangle is rotated on the next step, and so must be enclosed in a still larger rectangle. Thus, at each step, the enclosing rectangles become larger and larger, but

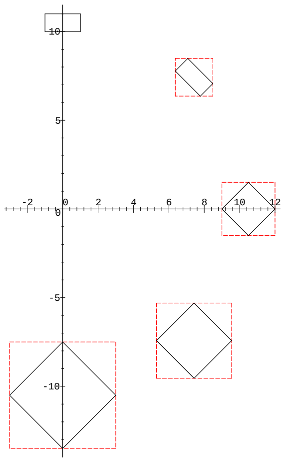

FIG. 6.1. *Wrapping of a rectangle specified by the interval vector* $([-1,1],[10,11])^T$. *The rotated rectangle is wrapped at* $t = \frac{\pi}{4}n$, *where* $n = 1, \ldots, 4$.

the set $\{A(t)y_0 \mid y_0 \in [y_0],\ t > t_0\}$ remains a rectangle of the same size. Moore [30, p. 134] showed that at $t = 2\pi$, the interval inclusion is inflated by a factor of $e^{2\pi} \approx 535$, as the stepsize tends to zero.

Jackson [20] gives a definition of wrapping.

DEFINITION 6.1. *Let* $T \in \mathbb{R}^{n\times n}$, $[x] \in \mathbb{IR}^n$, *and* $c \in \mathbb{R}^n$. *Then the wrapping of the parallelepiped*

$$P = \{Tx + c \mid x \in [x]\}$$

is the tightest interval vector containing P.

It can be easily seen that the wrapping of the set $\{Tx + c \mid x \in [x]\}$ is given by $T\,[x]{+}c$, where $\{T[x]\}_i = \sum_{k=1}^{n} T_{ik}\,[x_k]$.

6.1. Systems that Do Not Suffer from the Wrapping Effect. We begin with a definition.

DEFINITION 6.2. [42] *Let* $f : \mathbb{R}^n \to \mathbb{R}^n$, $f = (f_1, f_2, \ldots, f_n)^T$ *and* $f_i = f_i(y_1, y_2, \ldots y_n)$, $i = 1, \ldots, n$. *Then* f *is quasi-isotone if* f_i *is non-decreasing with respect to all* y_j *with* $j \neq i$. *That is,*

$$f_i(u) \leq f_i(v), \qquad \text{for} \quad u \leq v, \qquad u_i = v_i.$$

If the right side of a system of ODEs is quasi-isotone, then the following theorem holds.

THEOREM 6.1.1. [34] *If the function* f *in (3.1) is quasi-isotone, then*

$$y(t; t_0, \underline{y}_0) \leq y(t; t_0, y_0) \leq y(t; t_0, \bar{y}_0)$$

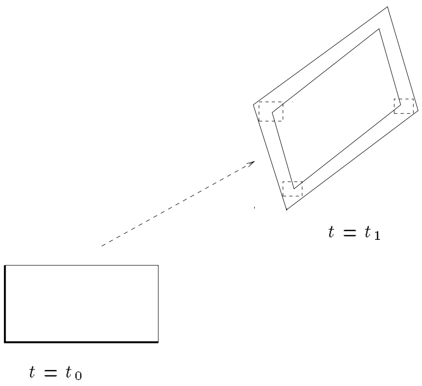

FIG. 6.2.1. *The initial set in $\mathbb{R}^2$ is mapped into a parallelepiped. Then a new parallelepiped is determined such that it contains the inclusions of the point solutions.*

for all $y_0 \in [y_0]$.
If the system is quasi-isotone, and we wrap the set of solutions at each point $t_1, t_2, \ldots t_j$, we obtain the same overestimate as if we wrap the set of solutions only at t_j. In Moore's example, the width of the interval vector at a point t_j increases as we take more, but smaller, steps to reach t_j.

6.2. Wrapping Effect in Linear Systems. Independently of each other, Nickel [33] and Lohner [24] proposed a method for handling the wrapping effect in linear systems. Consider the linear system

$$u' = G(t)u(t) + g(t) \qquad u(t_0) = u_0 \in [u_0]\,,$$

the solution of which is

$$u(t) = \Psi(t)u_0 + \Psi(t)\int_{t_0}^{t} \Psi^{-1}(s)g(s)ds, \tag{6.2.1}$$

where $\Psi(t)$ is the solution of $\Psi'(t) = G(t)\Psi(t)$, $\Psi(t_0) = I$. Then (6.2.1) is an affine transformation of the initial conditions. Since a parallelepiped in $\mathbb{R}^n$ can be specified by $(n+1)$ points, the idea is to follow the solutions with initial conditions at these points. However, at each step, we have to compute enclosures of the point solutions only. This implies that at each step we need to determine a parallelepiped that contains these enclosures, as shown in Fig. 6.2.1.

7. Algorithm II: Computing a Tight Enclosure. Suppose that at the $(j+1)$-st step we have a constant enclosure $[\tilde{y}_j]$ such that

$$y(t; t_j, [y_j]) \subseteq [\tilde{y}_j]\,, \qquad \text{for all } t \in [t_j, t_{j+1}].$$

Using $[\tilde{y}_j]$, Algorithm II computes a tighter enclosure $[y_{j+1}] \subseteq [\tilde{y}_j]$, for which

$$y(t_{j+1}; t_j, [y_j]) \subseteq [y_{j+1}]\,.$$

The main difficulty that arises in the second algorithm is controlling the wrapping effect.

Moore's method (see §5.3) computes $[y_{j+1}]$ by

$$\begin{aligned}[y_{j+1}] &= [y_j] + \sum_{i=1}^{k-1} [y_j]_i \, h_j^i + f^{[k]}([\tilde{y}_j^1])h_j^k \\ &= [y_j] + \sum_{i=1}^{k-1} f^{[i]}([y_j])h_j^i + f^{[k]}([\tilde{y}_j^1])h_j^k,\end{aligned} \tag{7.1}$$

where $[\tilde{y}_j^1]$ is given by (5.3.2). A disadvantage of using formula (7.1) is that the widths of $[y_j]$ always increase with j, even if the true solutions contract. This follows from property (2.1.3) applied to (7.1),

$$w([y_{j+1}]) = w([y_j]) + \sum_{i=1}^{k-1} w\left(f^{[i]}([y_j])\right) h_j^i + w\left(f^{[k]}([\tilde{y}_j^1])\right) h_j^k \geq w([y_j]),$$

where an equality is possible only in the trivial case in which the terms after $[y_j]$ have zero widths. Moore does not apply his method to compute the enclosures for (3.1–3.2), but uses it to compute the enclosures for another problem, for which the solution changes slowly, and then recovers the solution by a local coordinate transformation. Formula (7.1) is a straightforward evaluation of

$$y_{j+1} = y_j + \sum_{i=1}^{k-1} f^{[i]}(y_j)h_j^i + f^{[k]}(y; t_j, t_{j+1})h_j^k, \tag{7.2}$$

in interval arithmetic with y_j replaced by $[y_j]$, and $f^{[k]}(y; t_j, t_{j+1})$ replaced by $f^{[k]}([\tilde{y}_j^1])$. Here, $f^{[k]}(y; t_j, t_{j+1})$ denotes $f^{[k]}$ with its l-th component evaluated at $y(\xi_{jl})$, for some $\xi_{jl} \in [t_j, t_{j+1}]$. If we use the mean-value evaluation (2.3.1) for computing the enclosures of $R\left(f^{[i]}; [y_j]\right)$ $(i = 1, \ldots, k-1)$ instead of the direct evaluation $f^{[i]}([y_j])$, we can often obtain enclosures with smaller widths than in (7.1) [37]. By applying the mean-value theorem to $f^{[i]}$ at some $\hat{y}_j \in [y_j]$, we have

$$f^{[i]}(y_j) = f^{[i]}(\hat{y}_j) + J\left(f^{[i]}; y_j, \hat{y}_j\right)(y_j - \hat{y}_j), \tag{7.3}$$

where $J\left(f^{[i]}; y_j, \hat{y}_j\right)$ is the Jacobian of $f^{[i]}$ with its l-th row evaluated at $y_j + \theta_{il}(\hat{y}_j - y_j)$ for some $\theta_{il} \in [0, 1]$ $(l = 1, \ldots, n)$. Then from (7.2) and (7.3),

$$\begin{aligned}y_{j+1} = \hat{y}_j &+ \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}(y; t_j, t_{j+1})h_j^k \\ &+ \left\{ I + \sum_{i=1}^{k-1} J\left(f^{[i]}; y_j, \hat{y}_j\right) h_j^i \right\} (y_j - \hat{y}_j).\end{aligned} \tag{7.4}$$

Eijgenraam's, Lohner's, and the extended mean-value methods do not look similar in their original descriptions, but they are all realizations of different algorithms for the computation of (7.4) in interval-arithmetic.

We describe Moore's local coordinate transformation in §7.1. In §7.2, we show a straightforward method for evaluating (7.4) in interval arithmetic. In §7.3 through §7.6, we describe the methods of Krückeberg, Eijgenraam, Lohner, and Rihm, respectively.

7.1. Moore's Local Coordinate Transformation. We present the main points of Moore's method for reducing the wrapping effect. More details about the method can be found in [29] or [30, pp. 135–140].

Let $y(t; t_0, y_0)$ and $y(t; t_0, v_0)$ be exact solutions of (3.1) with initial conditions $y_0 \in [y_0]$ and $v_0 \in [y_0]$ at t_0. We expand $y(t; t_0, v_0)$ around y_0,

$$y(t; t_0, v_0) = y(t; t_0, y_0) + C(t, y_0)(v_0 - y_0) + O(\|v_0 - y_0\|^2), \tag{7.1.1}$$

where the matrix $C(t, y_0)$ has entries

$$C_{ij}(t, y_0) = \left. \frac{\partial y_i(t; t_0, u)}{\partial u_j} \right|_{u=y_0}.$$

Moore calls $C(t, y_0)$ the *connection* matrix for the solution curve $y(t; t_0, y_0)$. Let

$$\tilde{y}(t; t_0, v_0) = y(t; t_0, y_0) + C(t, y_0)(v_0 - y_0). \tag{7.1.2}$$

Since $\tilde{y}(t; t_0, v_0)$ is a good approximation to $y(t; t_0, v_0)$ when $\|v_0 - y_0\|$ is small, Moore defines $z(t, y_0, v_0)$ by

$$y(t; t_0, v_0) = y(t; t_0, y_0) + C(t, y_0) z(t, y_0, v_0), \tag{7.1.3}$$

based on (7.1.1) and (7.1.2). The function z is well-defined since $C(t, y_0)$ is nonsingular.

Let $J(t; y(t; t_0, y_0))$ be the Jacobian of f evaluated at $y(t; t_0, y_0)$. It can be shown that $C(t, y_0)$ satisfies the variational equation

$$\frac{\partial C(t, y_0)}{\partial t} = J(t; y(t; t_0, y_0)) C(t, y_0) \tag{7.1.4}$$

$$C(t_0, y_0) = I, \tag{7.1.5}$$

where I is the identity matrix. Since y_0 is fixed, we also use the notation $C(t) \equiv C(t, y_0)$ and $J(t) \equiv J(t; y(t; t_0, y_0))$. By differentiating (7.1.3) with respect to t and using (7.1.4), we obtain

$$y'(t; t_0, v_0) = y'(t; t_0, y_0) + J(t) C(t) z + C(t) z', \tag{7.1.6}$$

where $z \equiv z(t, y_0, v_0)$. From (7.1.3–7.1.6), we obtain an initial value problem for z,

$$\begin{aligned} z' &= (C(t))^{-1} \{ y'(t; t_0, v_0) - y'(t; t_0, y_0) - J(t) C(t) z \} \\ &= (C(t))^{-1} \{ f(y(t; t_0, v_0)) - f(y(t; t_0, y_0)) - J(t) C(t) z \} \\ &= (C(t))^{-1} \{ f(y(t; t_0, y_0) + C(t) z) - f(y(t; t_0, y_0)) - J(t) C(t) z \} \end{aligned} \tag{7.1.7}$$

$$z(t_0) = v_0 - y_0 \in [y_0] - y_0. \tag{7.1.8}$$

Moore applies his Taylor series method (see §5.3) to (7.1.7–7.1.8) and then recovers the solution in y coordinates by (7.1.3). The reason for using the system (7.1.7–7.1.8) is that, if $w([y_0] - y_0) = w([y_0]) = \epsilon$ is small, then hopefully, the Taylor series method will produce enclosures with width $w([y_0]) + O(\epsilon^2)$. It can be shown, from (7.1.7) and (7.1.5) that

$$z'(t_0, y_0, v_0) \in [u_0],$$

where $w([u_0]) = O(\epsilon^2)$, and, by the continuity of the right side of (7.1.7), $w([u_0])$ remains of order ϵ^2 for a significant length of time if ϵ is sufficiently small. Therefore, we can expect that the solution z changes slowly. That is,

$$\begin{aligned} z(t, y_0, v_0) &= z(t_0, y_0, v_0) + O(\epsilon^2) \\ &= v_0 - y_0 + O(\epsilon^2) \\ &\in [y_0] - y_0 + O(\epsilon^2). \end{aligned}$$

Moore uses a more tractable version of the transformation (7.1.3). Let

$$y(t; t_0, v_0) = \hat{y}(t; t_0, y_0) + \widehat{C}(t, y_0) z(t, y_0, v_0), \tag{7.1.9}$$

where $\hat{y}(t; t_0, y_0)$ is an approximation to $y(t; t_0, y_0)$, and $\widehat{C}(t, y_0)$ is an approximation to $C(t, y_0)$. Note that we do not need to use exact values for $y(t; t_0, y_0)$ and $C(t, y_0)$ since we need only to set up a local coordinate system. Then (7.1.7) becomes

$$z' = (\widehat{C}(t))^{-1} \left\{ f\left(\hat{y}(t; t_0, y_0) + \widehat{C}(t) z \right) - \hat{y}'(t; t_0, y_0) - \widehat{J}(t) \widehat{C}(t) z \right\} \tag{7.1.10}$$

$$z(t_0) = v_0 - y_0 \in [y_0] - y_0, \tag{7.1.11}$$

where $\hat{y}'(t; t_0, y_0) = \frac{\partial \hat{y}(t; t_0, y_0)}{\partial t}$. Instead of the system (7.1.7–7.1.8), the system (7.1.10–7.1.11) is supplied as an input to Algorithm 5.3.1, and then an enclosure of the solution in y coordinates is computed by (7.1.9). A method for the recursive generation of the Taylor coefficients for $z(t)$ is described in [29] and [30, pp. 139].

The solution $y(t; t_0, y_0)$ and its derivative are approximated at $t = t_{j+1}$ by

$$\hat{y}(t_{j+1}; t_0, y_0) := \hat{y}_{j+1} = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j) h_j^i \quad \text{and} \tag{7.1.12}$$

$$\hat{y}'(t_{j+1}; t_0, y_0) := \hat{y}'_{j+1} = \sum_{i=1}^{k-1} i f^{[i]}(\hat{y}_j) h_j^{i-1}, \tag{7.1.13}$$

where $\hat{y}_0 = y_0$. The Jacobian is approximated at t_j by $J(t_j; \hat{y}_j)$, which we also denote by $J(f; \hat{y}_j)$. The matrix $C(t_{j+1}, y_0)$ is approximated by applying the Euler method to (7.1.4–7.1.5),

$$\widehat{C}(t_{j+1}, y_0) = \{I + J(f; y_j) h_j\} \widehat{C}(t_j, y_0). \tag{7.1.14}$$

Let $[z_{j+1}]$ be computed on the $(j+1)$-st step by applying Algorithm 5.3.1 to (7.1.10–7.1.11). Let $A_j = \widehat{C}(t_j, y_0)$ and $S_j = I + J(f; y_j) h_j$. By using (7.1.12) and (7.1.14), we can rewrite (7.1.9) in the form

$$\begin{aligned} y(t_{j+1}; t_0, v_0) &\in \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j) h_j^i + \{I + J(f; y_j) h_j\} A_j [z_{j+1}] \\ &= \hat{y}_{j+1} + S_j A_j [z_{j+1}] = \hat{y}_{j+1} + A_{j+1} [z_{j+1}]. \end{aligned} \tag{7.1.15}$$

7.2. Direct Method. A straightforward method for computing a tight enclosure $[y_{j+1}]$ at t_{j+1} is based on the evaluation of (7.4) in interval arithmetic. From (7.4), since

$$\begin{aligned} & f^{[k]}(y; t_j, t_{j+1}) \in f^{[k]}([\tilde{y}_j]), \\ & J\left(f^{[i]}; y_j, \hat{y}_j\right) \in J\left(f^{[i]}; [y_j]\right) \qquad (i = 1, \ldots, k-1), \text{ and} \\ & y_j, \hat{y}_j \in [y_j], \end{aligned}$$

we have

$$
\begin{aligned}
[y_{j+1}] = \hat{y}_j + &\sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}([\tilde{y}_j])h_j^k \\
&+ \left\{ I + \sum_{i=1}^{k-1} J\left(f^{[i]};[y_j]\right) h_j^i \right\} ([y_j] - \hat{y}_j).
\end{aligned} \tag{7.2.1}
$$

Here, $[\tilde{y}_j]$ is an a priori enclosure of $y(t;t_j,[y_j])$ for all $t \in [t_j, t_{j+1}]$, $[y_j]$ is a tight enclosure of the solution at t_j, and $J\left(f^{[i]};[y_j]\right)$ is the Jacobian of $f^{[i]}$ evaluated at $[y_j]$. We choose $\hat{y}_0$ to be the midpoint of the initial interval $[y_0]$. Then, we choose

$$
\hat{y}_{j+1} = m\left(\hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}([\tilde{y}_j])h_j^k\right). \tag{7.2.2}
$$

That is, $\hat{y}_{j+1}$ is the midpoint of the enclosure of the point solution at t_{j+1} starting from $\hat{y}_j$. For convenience, we introduce the notation $(j \geq 0)$

$$
[v_{j+1}] = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}([\tilde{y}_j])h_j^k \tag{7.2.3}
$$

$$
[S_j] = I + \sum_{i=1}^{k-1} J\left(f^{[i]};[y_j]\right) h_j^i. \tag{7.2.4}
$$

Using (7.2.3–7.2.4), (7.2.1) can be written in the form

$$
[y_{j+1}] = [v_{j+1}] + [S_j]\,([y_j] - \hat{y}_j). \tag{7.2.5}
$$

In (7.2.5), $[v_{j+1}]$ encloses the local error from the current step, and $[S_j]\,([y_j] - \hat{y}_j)$ propagates the error from the previous step.

By a *direct method* we mean using (7.2.1), or (7.2.5), to compute a tight enclosure of the solution. This method is summarized in Algorithm 7.2.1. (Note that from (7.2.2), (7.2.3), and (7.2.5), $\hat{y}_{j+1} = m\,([v_{j+1}]) = m\,([y_{j+1}])$.)

Algorithm 7.2.1

Input:

$[\tilde{y}_j]$, h_j, $[y_j]$, $\hat{y}_j = m\,([y_j])$.

Compute:

$[v_{j+1}] = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}([\tilde{y}_j])h_j^k$;

$[S_j] = I + \sum_{i=1}^{k-1} J\left(f^{[i]};[y_j]\right) h_j^i$;

$[y_{j+1}] = [v_{j+1}] + [S_j]\,([y_j] - \hat{y}_j)$;

$\hat{y}_{j+1} = m\,([v_{j+1}])$ $(= m\,([y_{j+1}]))$.

Output:

$[y_{j+1}]$, $\hat{y}_{j+1}$.

If we use the direct method to compute the enclosures $[y_j]$, we might obtain unacceptably large interval vectors. This can be seen from the following considerations [37].

Using (7.2.5), we compute

$$\begin{aligned}
[y_1] &= [v_1] + [S_0]\,([y_0] - \hat{y}_0), \\
[y_2] &= [v_2] + [S_1]\,([y_1] - \hat{y}_1) \\
&= [v_2] + [S_1]\,(([v_1] - \hat{y}_1) + [S_0]\,([y_0] - \hat{y}_0)) \\
&= [v_2] + [S_1]\,([v_1] - \hat{y}_1) + [S_1]\,([S_0]\,([y_0] - \hat{y}_0)), \\
&\vdots \\
[y_{j+1}] &= [v_{j+1}] + [S_j]\,([y_j] - \hat{y}_j) \\
&= [v_{j+1}] + [S_j]\,([v_j] - \hat{y}_j) \\
&+ [S_j]\,([S_{j-1}]\,([v_{j-1}] - \hat{y}_{j-1})) \\
&+ \cdots \\
&+ [S_j]\,([S_{j-1}] \cdots ([S_1]\,([S_0]\,([v_0] - \hat{y}_0))) \cdots),
\end{aligned} \tag{7.2.6}$$

where $[v_0] = [y_0]$. Note that the interval vectors $[v_l] - \hat{y}_l$ $(l = 0, \ldots, j)$ are symmetric, and denote them by $[\delta_l] = [v_l] - \hat{y}_l$. Let us consider one of the summands in (7.2.6), for example, the last one,

$$[S_j]\,([S_{j-1}] \cdots ([S_1]\,([S_0]\,[\delta_0])) \cdots). \tag{7.2.7}$$

To simplify our discussion, we assume that the matrices in (7.2.7) are point matrices and denote them by $S_j, S_{j-1}, \ldots, S_0$. What we want to compute is the tightest interval vector that contains the set

$$\{S_j(S_{j-1} \cdots (S_1(S_0\delta_0)) \cdots) \mid \delta_0 \in [\delta_0]\}.$$

But this set is the same as

$$\{(S_jS_{j-1} \cdots S_1S_0)\delta_0 \mid \delta_0 \in [\delta_0]\},$$

which is wrapped by the interval vector

$$(S_jS_{j-1} \cdots S_1S_0)\,[\delta_0] \tag{7.2.8}$$

(see §6). In practice, though, we compute

$$S_j(S_{j-1} \cdots (S_1(S_0\,[\delta_0])) \cdots), \tag{7.2.9}$$

and we can have wrapping at each step. That is, we first compute $S_0\,[\delta_0]$, resulting in one wrapping, then we compute $S_1(S_0\,[\delta_0])$, resulting in another wrapping, and so on. We can also see the result of the wrapping effect if we express the widths of the interval vectors in (7.2.8) and (7.2.9):

$$\begin{aligned}
w\,((S_jS_{j-1} \cdots S_1S_0)\,[\delta_0]) &= |S_jS_{j-1} \cdots S_1S_0|\,w\,([\delta_0]) \\
&\le |S_j|\,|S_{j-1}| \cdots |S_1|\,|S_0|\,w\,([\delta_0]) \\
&= w\,(S_j(S_{j-1} \cdots (S_1(S_0\,[\delta_0])) \cdots)).
\end{aligned}$$

Frequently, $w\,((S_jS_{j-1} \cdots S_1S_0)\,[\delta_0]) \ll w\,(S_j(S_{j-1} \cdots (S_1(S_0\,[\delta_0])) \cdots))$ for j large, and the direct method often produces enclosures with increasing widths. However, it is suitable for systems with quasi-isotone right-hand sides, because, if f is quasi-isotone and the stepsize is small enough, the matrices $[S_j]$ are nonnegative.

The main goal of the methods that we describe in the next four subsections is to avoid the direct evaluations of products of the form

$$[S_j]\,([S_{j-1}]\cdots([S_l]\,[\delta_l])\cdots).$$

By choosing the vectors $\hat{y}_l = m([v_l])$, we provide symmetric intervals $[v_l] - \hat{y}_l$, and by (2.2.10), we should have smaller overestimations in the evaluations of the enclosures than if we were to use non-symmetric interval vectors.

Computing $[S_j]$. Lohner shows how the matrices $[S_j]$ can be computed (see for example [1]). We describe his method in a slightly different notation.

Consider the variational equation

$$\Psi' = J\,(f;y)\,\Psi, \qquad \Psi(t_j) = I. \tag{7.2.10}$$

It can be shown that

$$\frac{\Psi^{(i)}(t)}{i!} = J\left(f^{[i]};y\right)\Psi(t), \tag{7.2.11}$$

where $f^{[i]}$ is defined in (2.5.5–2.5.6), and $J\left(f^{[i]};y\right)$ is the Jacobian of $f^{[i]}$. Then, from the Taylor series expansion of $\Psi(t)$ and (7.2.10–7.2.11), we have

$$\Psi(t_{j+1}) = I + \sum_{i=1}^{k-1} J\left(f^{[i]};y\,(t_j)\right)h_j^i + (\text{Remainder Term}). \tag{7.2.12}$$

Since, in (7.2.12),

$$I + \sum_{i=1}^{k-1} J\left(f^{[i]};y\,(t_j)\right)h_j^i \in I + \sum_{i=1}^{k-1} J\left(f^{[i]};[y_j]\right)h_j^i = [S_j]\,, \tag{7.2.13}$$

the interval matrices $[S_j]$ can be computed by computing the interval Taylor series (7.2.13) for the variational equation (7.2.10).

7.3. Krückeberg's Three Process Method. Krückeberg [22] proposes a three-process method. We can consider process 1 as Algorithm I and processes 2 and 3 as Algorithm II.

Process 1. Compute a stepsize h_j and a polynomial with interval vector coefficients $[P(t)] = \sum_{i=0}^{n}[p_i]\,(t-t_j)^i$ such that

$$y(t;t_j,[y_j]) \subseteq [P(t)]\,, \qquad \text{for } t \in [t_j, t_{j+1}]\,.$$

The result of this process will be used in processes 2 and 3.

Process 2. Let $\hat{y}_j \in [v_j]$, where $[v_0] = [y_0]$. Compute an enclosure $[v_{j+1}]$ such that

$$y(t_{j+1};t_j,\hat{y}_j) \in [v_{j+1}]\,. \tag{7.3.1}$$

Krückeberg does not specify exactly how $\hat{y}_j$ is chosen, but we shall assume that $\hat{y}_j = m([v_j])$.

Process 3. Compute an enclosure $[u_{j+1}]$ such that

$$y(t_{j+1};t_j,[y_j]) - y(t_{j+1};t_j,\hat{y}_j) \subseteq [u_{j+1}]\,. \tag{7.3.2}$$

We obtain from (7.3.1) and (7.3.2) that

$$y(t_{j+1}; t_j, [y_j]) \subseteq [v_{j+1}] + [u_{j+1}]\,. \tag{7.3.3}$$

For the realization of process 1, Krückeberg points out that it is sufficient to have an enclosure for $n = 0$, which gives a constant a priori enclosure, and that only the cases $n = 1, 2$ might be of practical interest. We assume that we have computed a constant a priori enclosure $[\tilde{y}_j]$ in process 1.

Process 2 is realized by

$$[v_{j+1}] = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}([\tilde{y}_j])h_j^k.$$

We describe process 3 in more detail since its implementation is crucial for the quality of the enclosure in (7.3.3). Let $y(t; t_j, y_j)$ and $y(t; t_j, \hat{y}_j)$ be exact solutions to (3.1) starting from initial values $y_j, \hat{y}_j \in [y_j]$, respectively, at t_j. Consider $e(t) = y(t; t_j, y_j) - y(t; t_j, \hat{y}_j)$. To find $[u_{j+1}]$ in (7.3.2), we have to find an enclosure of $e(t_{j+1})$ when y_j varies in $[y_j]$. By differentiating $e(t)$, we obtain

$$e'(t) = J\left(f; y(t; t_j, y_j), y(t; t_j, \hat{y}_j)\right) e(t) \in J\left(f; [\tilde{y}_j]\right) e(t), \tag{7.3.4}$$

where $J\left(f; y(t; t_j, y_j), y(t; t_j, \hat{y}_j)\right)$ is the Jacobian of f with its l-th row ($l = 1, \dots, n$) evaluated at $y_{\xi_l}(t)$, which is a value somewhere on the line joining $y(t; t_j, y_j)$ and $y(t; t_j, \hat{y}_j)$. Since $e(t_j) = y_j - \hat{y}_j$, we have from (7.3.4)

$$\begin{aligned} e(t_{j+1}) &\in e^{h_j J(f; [\tilde{y}_j])}([y_j] - \hat{y}_j) \\ &= \left(\sum_{i=0}^{\infty} \frac{\left\{h_j J\left(f; [\tilde{y}_j]\right)\right\}^i}{i!} \right) ([y_j] - \hat{y}_j) \\ &=: [S_j]([y_j] - \hat{y}_j). \end{aligned} \tag{7.3.5}$$

Krückeberg does not show how one can compute the matrices $[S_j]$.

It follows from (7.3.3) and (7.3.5) that

$$y(t_{j+1}; t_j, [y_j]) \subseteq [v_{j+1}] + [S_j]\left([y_j] - \hat{y}_j\right). \tag{7.3.6}$$

From the considerations in the previous subsection, if we use

$$[y_{j+1}] = [v_{j+1}] + [S_j]\left([y_j] - \hat{y}_j\right) \tag{7.3.7}$$

to compute the enclosures at each step, we could obtain interval vectors with unacceptable widths, caused by the wrapping effect at each step.

Let $\widehat{S}_j = m\left([S_j]\right)$. From (7.3.7) and (2.2.4),

$$y(t_{j+1}; t_j, [y_j]) \subseteq [v_{j+1}] + ([S_j] - \widehat{S}_j)([y_j] - \hat{y}_j) + \widehat{S}_j([y_j] - \hat{y}_j). \tag{7.3.8}$$

Here, we take $\widehat{S}_j = m\left([S_j]\right)$, while in the original description of the method the only assumption is that $\widehat{S}_j \in [S_j]$. Krückeberg also represents the enclosure of the solution in the form $\hat{y}_j + A_j\,[x_j]$, such that

$$[y_j] \subseteq \hat{y}_j + A_j\,[x_j] \qquad (j \geq 1), \tag{7.3.9}$$

where $A_j \in \mathbb{R}^{n\times n}$, A_j is regular, and $[x_j] \in \mathbb{IR}^n$ is symmetric. For $j = 0$,

$$A_0 = I \text{ and } [x_0] = [y_0] - \hat{y}_0,$$

where I is the identity matrix. Then, from (7.3.8) and (7.3.9),

$$y(t_{j+1}; t_j, [y_j]) \subseteq [v_{j+1}] + ([S_j] - \widehat{S}_j)([y_j] - \hat{y}_j) + \widehat{S}_j(A_j\,[x_j]). \tag{7.3.10}$$

Consider the term $\widehat{S}_j(A_j\,[x_j])$ in (7.3.10). We are interested in computing the tightest interval vector containing the set

$$\left\{\widehat{S}_j A_j x_j \mid x_j \in [x_j]\right\} = \left\{(\widehat{S}_j A_j) x_j \mid x_j \in [x_j]\right\}.$$

This vector is given by $(\widehat{S}_j A_j)\,[x_j]$, (see §6) whereas in (7.3.10) we compute $\widehat{S}_j(A_j\,[x_j])$. In the former case, the wrapping effect can appear at most once, while in the latter it can appear at most twice. Since both $[S_j] - \widehat{S}_j$ and $[x_j]$ are symmetric and

$$([S_j] - \widehat{S}_j)([y_j] - \hat{y}_j) \subseteq ([S_j] - \widehat{S}_j)(A_j\,[x_j]) = (([S_j] - \widehat{S}_j)A_j)\,[x_j],$$

we do not replace $([S_j] - \widehat{S}_j)([y_j] - \hat{y}_j)$ in (7.3.10) by $(([S_j] - \widehat{S}_j)A_j)\,[x_j]$.

Let

$$A_{j+1} = \widehat{S}_j A_j \text{ and} \tag{7.3.11}$$

$$[b_{j+1}] = [v_{j+1}] - \hat{y}_{j+1} + ([S_j] - \widehat{S}_j)([y_j] - \hat{y}_j). \tag{7.3.12}$$

Using (7.3.10–7.3.12), we compute $[y_{j+1}]$ from

$$\begin{aligned} y(t_{j+1}; t_j, [y_j]) &\subseteq \hat{y}_{j+1} + A_{j+1}\,[x_j] + [b_{j+1}] \\ &=: [y_{j+1}]. \end{aligned} \tag{7.3.13}$$

As a preparation for the next step, we have to compute $[x_{j+1}]$. We transform the right side (7.3.13) as follows,

$$\begin{aligned} y(t_{j+1}; t_j, [y_j]) &\subseteq \hat{y}_{j+1} + A_{j+1}\,[x_j] + [b_{j+1}] \\ &= \hat{y}_{j+1} + A_{j+1}\,[x_j] + (A_{j+1}A_{j+1}^{-1})\,[b_{j+1}] \\ &\subseteq \hat{y}_{j+1} + A_{j+1}\,[x_j] + A_{j+1}(A_{j+1}^{-1}\,[b_{j+1}]) \\ &= \hat{y}_{j+1} + A_{j+1}\left\{[x_j] + A_{j+1}^{-1}\,[b_{j+1}]\right\} \\ &=: \hat{y}_{j+1} + A_{j+1}\,[x_{j+1}], \end{aligned} \tag{7.3.14}$$

where $[x_{j+1}] = [x_j] + A_{j+1}^{-1}\,[b_{j+1}]$.

Finally, Krückeberg's method can be summarized as the following algorithm.

Algorithm 7.3.1

Initialize:

$[y_0]$, $\hat{y}_0 = m([y_0])$, $A_0 = I$, $[x_0] = [y_0] - \hat{y}_0$.

Input:

$[\tilde{y}_j]$, h_j, $\hat{y}_j$, $[y_j]$, A_j, $[x_j]$.

Compute:

$[v_{j+1}] = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}([\tilde{y}_j])h_j^k$;

$[S_j] = \sum_{i=0}^{\infty} \frac{\{h_j J(f;[\tilde{y}_j])\}^i}{i!}$;

$\widehat{S}_j = m([S_j])$;

$\hat{y}_{j+1} = m([v_{j+1}])$;

$[b_{j+1}] = [v_{j+1}] - \hat{y}_{j+1} + ([S_j] - \widehat{S}_j)([y_j] - \hat{y}_j)$;

$A_{j+1} = \widehat{S}_j A_j$;

$[y_{j+1}] = \hat{y}_{j+1} + A_{j+1}[x_j] + [b_{j+1}]$;

$[x_{j+1}] = [x_j] + A_{j+1}^{-1}[b_{j+1}]$.

Output:

$\hat{y}_{j+1}$, $[y_{j+1}]$, A_{j+1}, $[x_{j+1}]$.

A disadvantage of the method is that it uses the constant a priori enclosure to compute $[S_j]$: this enclosure is usually larger than the enclosure in the methods to follow. If A_j becomes singular, for some $j \geq 1$, then the method breaks down.

The interval vector $[y_{j+1}]$ can be computed by

$$\hat{y}_{j+1} + A_{j+1}[x_j] + [b_{j+1}] \quad \text{or} \quad \hat{y}_{j+1} + A_{j+1}[x_{j+1}].$$

Because of the second inclusion in (7.3.14), we have chosen the former evaluation. However, Krückeberg does not point out how $[y_{j+1}]$ is to be computed.

7.4. Eijgenraam's Method. Eijgenraam [14] represents the enclosure $[y_j]$ of $y(t_j; t_0, [y_0])$ as a product of a non-singular matrix $A_j \in \mathbb{R}^{n\times n}$ and an interval vector $[x_j] \in \mathbb{IR}^n$,

$$[y_j] = A_j[x_j]. \tag{7.4.1}$$

For $j = 0$, $A_0 = I$, and $[y_0] = [x_0]$. Let

$$\hat{y}_j = m([y_j]), \tag{7.4.2}$$

$$S_j = I + \sum_{i=1}^{k-1} J\left(f^{[i]}; \hat{y}_j\right) h_j^i, \tag{7.4.3}$$

$$A_{j+1} = S_j A_j, \text{ and} \tag{7.4.4}$$

$$[c_j] = [v_{j+1}] - S_j\hat{y}_j + ([S_j] - S_j)([y_j] - \hat{y}_j), \tag{7.4.5}$$

where $[v_{j+1}]$ and $[S_j]$ are defined by (7.2.3) and (7.2.4). By adding and subtracting $S_j(y_j - \hat{y}_j)$ from (7.4) and using (7.4.1–7.4.5), we obtain

$$\begin{aligned} y(t_{j+1}; t_j, [y_j]) &\subseteq [v_{j+1}] + S_j([y_j] - \hat{y}_j) + ([S_j] - S_j)([y_j] - \hat{y}_j) \\ &= S_j(A_j[x_j]) + [c_j]. \end{aligned} \tag{7.4.6}$$

As in Krückeberg's method, we will obtain a better enclosure in (7.4.6) if we compute

$$(S_j A_j)\,[x_j] = A_{j+1}\,[x_j]$$

instead of $S_j(A_j\,[x_j])$. Therefore,

$$\begin{aligned} y(t_{j+1}; t_j, [y_j]) &\subseteq (S_j A_j)\,[x_j] + [c_j] \\ &= A_{j+1}\,[x_j] + [c_j] \\ &= A_{j+1}\,[x_j] \; + (A_{j+1} A_{j+1}^{-1})\,[c_j] \\ &\subseteq A_{j+1}\,[x_j] + A_{j+1}(A_{j+1}^{-1}\,[c_j]) \\ &= A_{j+1}\left([x_j] + A_{j+1}^{-1}\,[c_j]\right) \\ &= A_{j+1}\,[x_{j+1}] \\ &=: [y_{j+1}]\,, \end{aligned} \tag{7.4.7}$$

where $[x_{j+1}] = [x_j] + A_{j+1}^{-1}\,[c_j]$. Eijgenraam's Algorithm II follows from (7.4.7):

Algorithm 7.4.1

Initialize:

$[y_0]$, $A_0 = I$, $[x_0] = [y_0]$.

Input:

$[\tilde{y}_j]$, h_j, $[y_j]$, A_j, $[x_j]$.

Compute:

$\hat{y}_j = m\,([y_j])$;

$[v_{j+1}] = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j) h_j^i + f^{[k]}([\tilde{y}_j]) h_j^k$;

$S_j = I + \sum_{i=1}^{k-1} J\left(f^{[i]}; \hat{y}_j\right) h_j^i$;

$[S_j] = I + \sum_{i=1}^{k-1} J\left(f^{[i]}; [y_j]\right) h_j^i$;

$[c_j] = [v_{j+1}] - S_j \hat{y}_j + ([S_j] - S_j)([y_j] - \hat{y}_j)$;

$A_{j+1} = S_j A_j$;

$[x_{j+1}] = [x_j] + A_{j+1}^{-1}\,[c_j]$;

$[y_{j+1}] = A_{j+1}\,[x_{j+1}]$.

Output:

$[y_{j+1}]$, A_{j+1}, $[x_{j+1}]$.

Here, the matrix S_j is given by

$$S_j = I + \sum_{i=1}^{k-1} J\left(f^{[i]}; \hat{y}_j\right) h_j^i = I + J\,(f; \hat{y}_j)\, h_j + \sum_{i=2}^{k-1} J\left(f^{[i]}; \hat{y}_j\right) h_j^i$$

while in Moore's method it is (7.1.15)

$$S_j = I + J\,(f; \hat{y}_j)\, h_j\,.$$

Note also that Eijgenraam uses the tight enclosure from the previous step to compute $[S_j]$, while Krückeberg uses the a priori enclosure to compute $[S_j]$.

In forming $[y_{j+1}]$, Algorithm 7.4.1 evaluates indirectly

$$A_{j+1}(A_{j+1}^{-1}\,[c_j]),$$

which may be much larger than $[c_j]$, if A_{j+1} has a large condition number. However, what we really want is for $\| |A_{j+1}| \cdot |A_{j+1}^{-1}| \|$ to be small, because $cond\,(A_{j+1})$ can be large, and $\| |A_{j+1}| \cdot |A_{j+1}^{-1}| \|$ is still small. Since

$$\| |A| \cdot |A^{-1}| \| \le cond\,(A) \qquad \text{for a nonsingular } A \in \mathbb{R}^{n \times n},$$

a small condition number of A_{j+1} will always guarantee that $\| |A_{j+1}| \cdot |A_{j+1}^{-1}| \|$ is small.

7.5. Lohner's Method. We derive Lohner's method from (7.4) in a different way than in [1, 25, 26]. We show how $[y_1]$ and $[y_2]$ are computed, and then we give the algorithm for any $[y_j]$.

Let

$$z_{j+1} = f^{[k]}(y; t_j, t_{j+1}) h_j^k \in f^{[k]}([\tilde{y}_j]) h_j^k = [z_{j+1}], \tag{7.5.1}$$

$$s_{j+1} = m\left([z_{j+1}]\right), \tag{7.5.2}$$

$$\hat{y}_0 = m\left([y_0]\right),$$

$$\hat{y}_{j+1} = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j) h_j^i + s_{j+1}, \text{ and} \tag{7.5.3}$$

$$\begin{aligned} S_j &= I + \sum_{i=1}^{k-1} J\left(f^{[i]}; y_j, \hat{y}_j\right) h_j^i \\ &\in I + \sum_{i=1}^{k-1} J\left(f^{[i]}; [y_j]\right) h_j^i = [S_j]. \end{aligned} \tag{7.5.4}$$

Write also

$$A_0 = I \text{ and } r_0 = y_0 - \hat{y}_0 \in [r_0] = [y_0] - \hat{y}_0, \tag{7.5.5}$$

where I is the identity matrix.

Let $y_0 \in [y_0]$. Using the notation (7.5.1–7.5.5), we obtain

$$\begin{aligned} y_1 &= \hat{y}_1 + S_0(y_0 - \hat{y}_0) + z_1 - s_1 \\ &\in \hat{y}_1 + ([S_0] A_0)[r_0] + [z_1] - s_1 \\ &=: [y_1^1], \end{aligned} \tag{7.5.6}$$

$$\begin{aligned} y_1 &= \hat{y}_1 + S_0(y_0 - \hat{y}_0) + z_1 - s_1 \\ &= \hat{y}_1 + A_1 \left\{ \left(A_1^{-1} S_0 A_0\right) r_0 + A_1^{-1}(z_1 - s_1) \right\} \\ &= \hat{y}_1 + A_1 r_1 \\ &\in \hat{y}_1 + A_1 [r_1] \\ &=: [y_1^2], \end{aligned} \tag{7.5.7}$$

where $A_1 \in \mathbb{R}^{n \times n}$ is regular, and

$$\begin{aligned} r_1 &= \left(A_1^{-1} S_0 A_0\right) r_0 + A_1^{-1}(z_1 - s_1) \\ &\in \left(A_1^{-1}([S_0]\, A_0)\right)[r_0] + A_1^{-1}([z_1] - s_1) \\ &=: [r_1]. \end{aligned}$$

From (7.5.6) and (7.5.7), y_1 is contained in the intersection of $[y_1^1]$ and $[y_1^2]$, and we set

$$[y_1] := [y_1^1] \cap [y_1^2]. \tag{7.5.8}$$

We explain later how the matrices A_i $(i \geq 1)$ can be chosen.

Let $y_1 \in [y_1]$. Similarly,

$$\begin{aligned} y_2 &= \hat{y}_2 + S_1(y_1 - \hat{y}_1) + z_2 - s_2 \\ &= \hat{y}_2 + (S_1 A_1) r_1 + z_2 - s_2 \\ &\in \hat{y}_2 + ([S_1] A_1)\,[r_1] + [z_2] - s_2 \\ &=: [y_2^1], \end{aligned} \tag{7.5.9}$$

$$\begin{aligned} y_2 &= \hat{y}_2 + S_1(y_1 - \hat{y}_1) + z_2 - s_2 \\ &= \hat{y}_2 + A_2 \left\{ \left(A_2^{-1} S_1 A_1\right) r_1 + A_2^{-1}(z_2 - s_2) \right\} \\ &= \hat{y}_2 + A_2 r_2 \\ &\in \hat{y}_2 + A_2\,[r_2] \\ &=: [y_2^2], \end{aligned} \tag{7.5.10}$$

where $A_2 \in \mathbb{R}^{n \times n}$ is regular, and

$$\begin{aligned} r_2 \;&=\; \left(A_2^{-1} S_1 A_1\right) r_1 + A_2^{-1}(z_2 - s_2) \\ &\in\; \left(A_2^{-1}([S_1]\,A_1)\right)[r_1] + A_2^{-1}([z_2] - s_2) \\ &=: [r_2]\,. \end{aligned}$$

As in (7.5.8), we set $[y_2] := [y_2^1] \cap [y_2^2]$. Continuing in this fashion, we obtain Lohner's Algorithm II:

Algorithm 7.5.1

Initialize:

$[y_0]$, $\hat{y}_0 = m\,([y_0])$, $[r_0] = [y_0] - \hat{y}_0$, $A_0 = I$.

Input:

$[\tilde{y}_j]$, h_j, $\hat{y}_j$, $[y_j]$, A_j, $[r_j]$.

Compute:

$[z_{j+1}] = f^{[k]}([\tilde{y}_j])h_j^k$;

$s_{j+1} = m\,([z_{j+1}])$;

$\hat{y}_{j+1} = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + s_{j+1}$;

$[S_j] = I + \sum_{i=1}^{k-1} J\left(f^{[i]};[y_j]\right) h_j^i$;

Choose A_{j+1} as discussed below;

$[r_{j+1}] = \left(A_{j+1}^{-1}([S_j]\,A_j)\right)[r_j] + A_{j+1}^{-1}\,([z_{j+1}] - s_{j+1})$;

$[y_{j+1}^1] = \hat{y}_{j+1} + ([S_j]\,A_j)\,[r_j] + [z_{j+1}] - s_{j+1}$;

$[y_{j+1}^2] = \hat{y}_{j+1} + A_{j+1}\,[r_{j+1}]$;

$[y_{j+1}] = [y_{j+1}^1] \cap [y_{j+1}^2]$.

Output:

$\hat{y}_{j+1}, [y_{j+1}]$, A_{j+1}, $[r_{j+1}]$.

7.5.1. Parallelepiped Method. If $A_{j+1} = m\left([S_j]\, A_j\right)$, then we have the parallelepiped method. If $w\left([S_j]\right)$ is small and A_j is well conditioned, then

$$A_{j+1}^{-1}\, [S_j]\, A_j = I + [E_j]\,,$$

where $[E_j]$ is small. As a result, there is no large overestimation of $[r_j]$ in the evaluation of $(A_{j+1}^{-1}\, [S_j]\, A_j)[r_j]$. However, the choice of A_{j+1} does not guarantee the regularity of the matrix. Moreover, $cond\,(A_{j+1})$ can be large, and then large overestimations may occur.

7.5.2. QR-factorization Method. In this subsection, we describe Lohner's QR-factorization method, explain how it works, and illustrate it with a simple example.

Let $\tilde{A}_{j+1} \in [S_j]\, A_j$, and let $\widehat{A}_{j+1} = \tilde{A}_{j+1} P_{j+1}$, where P_{j+1} is a permutation matrix. We explain later in this subsection how P_{j+1} is chosen. We perform the QR-factorization $\widehat{A}_{j+1} = Q_{j+1} R_{j+1}$, where Q_{j+1} is an orthogonal matrix and R_{j+1} is an upper triangular matrix. If A_{j+1} is chosen to be Q_{j+1}, in Algorithm 7.5.1, we have the QR-factorization method for computing a tight enclosure of the solution.

We now give an intuitive explanation of how this method works. At each step, we want to compute a tight enclosure of the set

$$\left\{ \left(A_{j+1}^{-1} S_j A_j\right) r_j + A_{j+1}^{-1} \left(z_{j+1} - s_{j+1}\right) \;\middle|\; S_j \in [S_j]\,,\ r_j \in [r_j]\,,\ z_{j+1} \in [z_{j+1}] \right\}. \tag{7.5.11}$$

Consider first the set

$$\left\{ A_{j+1}^{-1} \left(z_{j+1} - s_{j+1}\right) \;\middle|\; z_{j+1} \in [z_{j+1}] \right\}. \tag{7.5.12}$$

If $\|A_{j+1}^{-1}\|$ is not much larger than 1, then

$$w\left(A_{j+1}^{-1}\left([z_{j+1}] - s_{j+1}\right)\right) = |A_{j+1}^{-1}|\, w\left([z_{j+1}]\right) \le \|A_{j+1}^{-1}\|\, w\left([z_{j+1}]\right)$$

will not be much larger than $w\left([z_{j+1}]\right)$. In this method, $A_{j+1}^{-1} = Q_{j+1}^{-1} = Q_{j+1}^{T}$ is orthogonal, so $\|A_{j+1}^{-1}\| \le \sqrt{n}$. In addition, $w\left([z_{j+1}]\right)$ can be made small by increasing the order of the Taylor series. Therefore, the set (7.5.12) can be enclosed in the interval vector

$$A_{j+1}^{-1}\left([z_{j+1}] - s_{j+1}\right),$$

whose width can be kept small.

Consider now the set

$$\left\{ \left(A_{j+1}^{-1} S_j A_j\right) r_j \;\middle|\; S_j \in [S_j]\,,\ r_j \in [r_j] \right\} \tag{7.5.13}$$

in (7.5.11). If $\tilde{A}_{j+1} \in \{S_j A_j \mid S_j \in [S_j]\} \subseteq [S_j]\, A_j$ and $w\left([S_j]\right)$ is small, then

$$\{(S_j A_j) r_j \mid S_j \in [S_j]\,,\ r_j \in [r_j]\} \approx \left\{ \tilde{A}_{j+1} r_j \mid r_j \in [r_j] \right\}. \tag{7.5.14}$$

From (7.5.13) and (7.5.14), we have

$$\begin{aligned} \{(A_{j+1}^{-1} S_j A_j) r_j \mid S_j \in [S_j]\,,\ r_j \in [r_j]\} &= \{(Q_{j+1}^{-1}(S_j A_j)) r_j \mid S_j \in [S_j]\,,\ r_j \in [r_j]\} \\ &\approx \left\{ (Q_{j+1}^{-1} \tilde{A}_{j+1}) r_j \mid r_j \in [r_j] \right\} \\ &\subseteq (Q_{j+1}^{-1} \tilde{A}_{j+1})[r_j]. \end{aligned} \tag{7.5.15}$$

Note that $\tilde{A}_{j+1}[r_j]$ is the wrapping of the set

$$\left\{\tilde{A}_{j+1}r_j \mid r_j \in [r_j]\right\}, \tag{7.5.16}$$

while $(Q_{j+1}^{-1}\tilde{A}_{j+1})[r_j]$ is the wrapping of the set

$$\left\{(Q_{j+1}^{-1}\tilde{A}_{j+1})r_j \mid r_j \in [r_j]\right\} = \left\{Q_{j+1}^{-1}(\tilde{A}_{j+1}r_j) \mid r_j \in [r_j]\right\}, \tag{7.5.17}$$

which is $\{r_j \in [r_j]\}$ mapped by $\tilde{A}_{j+1}$ and then the result mapped by Q_{j+1}^{-1}.

The vector corresponding to the first column of Q_{j+1} is parallel to the vector corresponding to the first column of $\widehat{A}_{j+1}$. The matrix Q_{j+1} induces an orthogonal coordinate system, where the axis corresponding to the first column of Q_{j+1} is parallel to those edges of the parallelepiped (7.5.16) that are parallel to the first column of $\widehat{A}_{j+1}$. Intuitively, we can expect a better enclosure in the coordinate system induced by Q_{j+1} than in the original coordinate system. Furthermore, if the first column of Q_{j+1} is parallel to the longest edge of the parallelepiped in (7.5.16), we can expect a better result than if this column were parallel to a shorter edge. This is the reason for rearranging the columns of $\tilde{A}_{j+1}$ by the permutation matrix P_{j+1}. Lohner suggests that P_{j+1} be chosen such that the first column of $\widehat{A}_{j+1}$ corresponds to the longest edge of (7.5.16), the second column to the second longest and so on.

If $\|\cdot\|_2$ is the Euclidean norm of a vector, $\tilde{A}_{j+1_i}$ is the i-th column of $\tilde{A}_{j+1}$ and $[r_j]_i$ is the i-th component of $[r_j]$, then the lengths of the edges of (7.5.16) are given by

$$l_i = \|\tilde{A}_{j+1_i}\|_2 \cdot w\left([r_j]_i\right), \qquad \text{for } i = 1, \ldots, n.$$

Let $l = (l_1, l_2, \ldots, l_n)^T$. The matrix P_{j+1} is such that the components of $l^T P_{j+1}$ are in non-increasing order (from 1 to n). As a result, the vector corresponding to the first column of $\widehat{A}_{j+1} = \tilde{A}_{j+1}P_{j+1}$ will be parallel to the longest edge of (7.5.16), and the first column of Q_{j+1} will be parallel to that edge as well.

Example. Let

$$A = \begin{pmatrix} 1 & 1 \\ 2 & 1 \end{pmatrix} \quad \text{and} \quad [r] = \begin{pmatrix} [1,2] \\ [1,4] \end{pmatrix}.$$

The QR-factorization of A is

$$A = \left\{-\frac{1}{\sqrt{5}}\begin{pmatrix} 1 & 2 \\ 2 & -1 \end{pmatrix}\right\}\left\{-\frac{1}{\sqrt{5}}\begin{pmatrix} 5 & 3 \\ 0 & 1 \end{pmatrix}\right\} =: QR \tag{7.5.18}$$

Consider the set

$$\{Ar \mid r \in [r]\}. \tag{7.5.19}$$

The parallelepiped specified by $[r]$ (see Fig. 7.5.1a) is mapped by A into the parallelepiped shown in Fig. 7.5.1b. The filled part in Fig. 7.5.1b is the overestimation of (7.5.19) by $A[r]$. However, if the set in (7.5.19) is wrapped in the coordinate system induced by Q, we obtain a better enclosure of this set (see Fig. 7.5.1c).

Consider now the set

$$\left\{Q^{-1}Ar \mid r \in [r]\right\}. \tag{7.5.20}$$

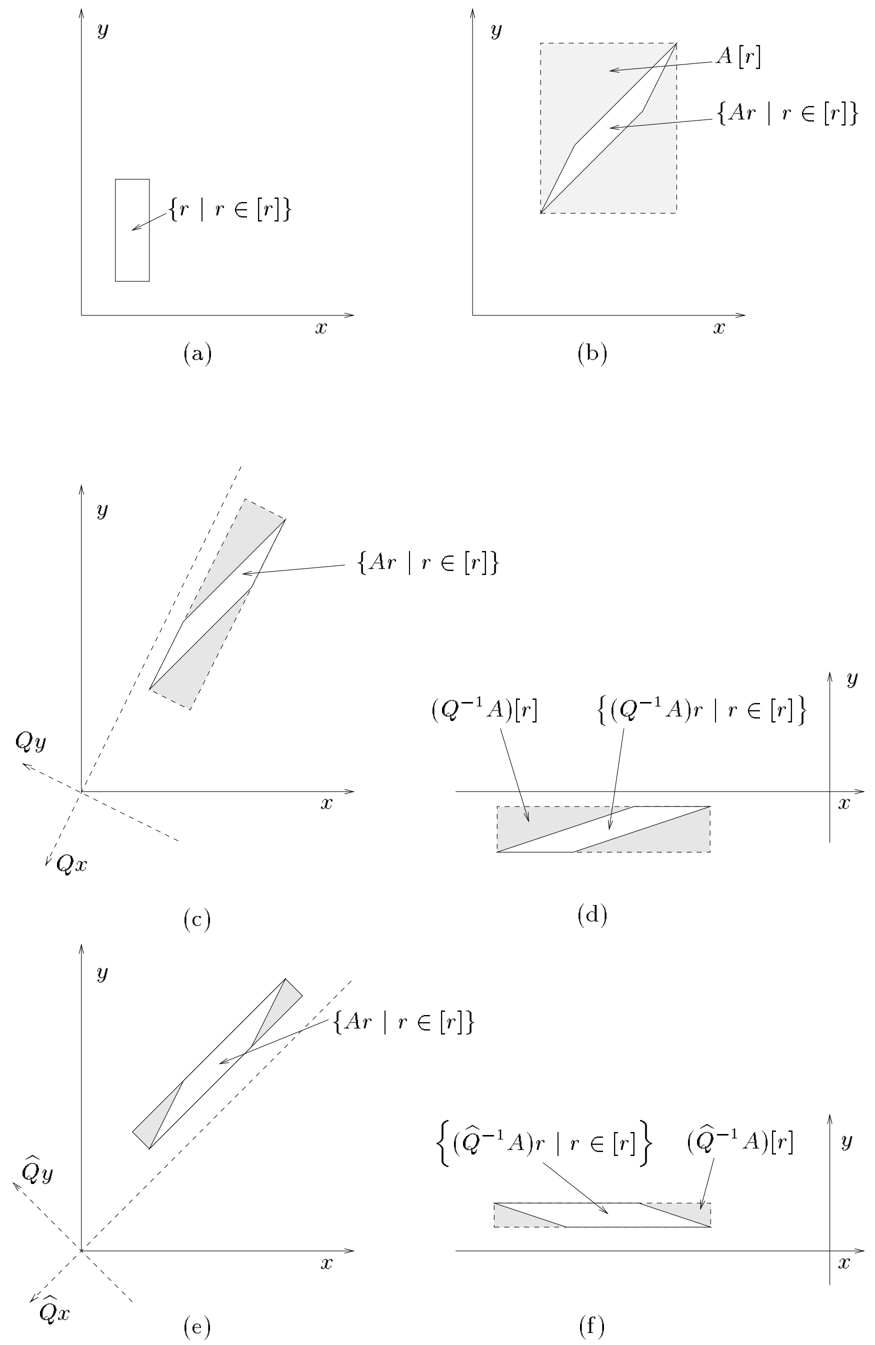

Fig. 7.5.1. *(a) The set* $\{r \mid r \in [r]\}$. *(b)* $\{Ar \mid r \in [r]\}$ *enclosed by* $A\,[r]$. *(c)* $\{Ar \mid r \in [r]\}$ *enclosed in the coordinate system induced by* Q. *(d)* $\left\{(Q^{-1}A)r \mid r \in [r]\right\}$ *enclosed by* $(Q^{-1}A)\,[r]$. *(e)* $\{Ar \mid r \in [r]\}$ *enclosed in the coordinate system induced by* $\widehat{Q}$. *(f)* $\left\{(\widehat{Q}^{-1}A)r \mid r \in [r]\right\}$ *enclosed by* $(\widehat{Q}^{-1}A)\,[r]$.

The matrix Q^{-1} maps (7.5.19) into a parallelepiped with its shorter edge parallel to the original x axis. As a result, the wrapping of (7.5.20) is $(Q^{-1}A)[r]$ (see Fig. 7.5.1d).

Now, interchange the columns of A, denote the new matrix by $\widehat{A}$, and compute the QR-factorization

$$(7.5.21) \qquad \widehat{A} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} = \left\{ -\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \right\} \left\{ -\frac{1}{\sqrt{2}} \begin{pmatrix} 2 & 3 \\ 0 & -1 \end{pmatrix} \right\} =: \widehat{Q}\widehat{R}.$$

If we wrap the set (7.5.19) in the coordinate system induced by $\widehat{Q}$ (see Fig. 7.5.1e), we obtain a better enclosure than in the coordinate system induced by Q. In Fig. 7.5.1f, the parallelepiped $\{Ar \mid r \in [r]\}$ is rotated by $\widehat{Q}^{-1}$. The longest edge of the rotated parallelepiped is parallel to the x axis, and the overestimation in $(\widehat{Q}^{-1}A)[r]$ is smaller than in $(Q^{-1}A)[r]$ and $A[r]$.

To summarize, let $A \in \mathbb{R}^{n\times n}$, $[r] \in \mathbb{IR}^n$, and $A = QR$, where Q is an orthogonal matrix and R is an upper triangular matrix. Normally, if we wrap the parallelepiped $\{Ar \mid r \in [r]\}$ in the coordinate system induced by Q, we obtain a better enclosure than in the original coordinate system. Moreover, if we rearrange the columns of A, as described in this subsection, before computing Q, we usually obtain a better enclosure than without rearranging those columns.

7.6. The Extended Mean-Value Method. We describe Rihm's [38] extended mean-value method as we described Lohner's Algorithm II in the previous subsection. We use the notation (7.5.4) and denote by

$$\begin{aligned} v_{j+1} &= \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}(y; t_j, t_{j+1})h_j^k \\ &\in \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j)h_j^i + f^{[k]}([\tilde{y}_j])h_j^k = [v_{j+1}]\,, \end{aligned}$$

where $\hat{y}_j$ is computed as in the direct method (§7.2), $f^{[k]}(y; t_j, t_{j+1})$ denotes $f^{[k]}$ with its l-th component evaluated at $y(\xi_{jl})$, for some $\xi_{jl} \in [t_j, t_{j+1}]$.

As in the direct method (§7.2), we compute

$$(7.6.1) \qquad [y_1] = [v_1] + [S_0]\left([y_0] - \hat{y}_0\right) =: [v_1] + [q_1]\,.$$

To show how we compute $[y_2]$, we represent $y_1 \in [y_1]$ in the form

$$(7.6.2) \qquad \begin{aligned} y_1 &= v_1 + S_0(y_0 - \hat{y}_0) \\ &= v_1 + A_1\left\{\left(A_1^{-1}S_0\right)(y_0 - \hat{y}_0)\right\} =: v_1 + A_1 p_1, \end{aligned}$$

where $A_1 \in \mathbb{R}^{n\times n}$ is regular and

$$(7.6.3) \qquad p_1 = \left(A_1^{-1}S_0\right)(y_0 - \hat{y}_0) \in \left(A_1^{-1}[S_0]\right)\left([y_0] - \hat{y}_0\right) =: [p_1]\,.$$

The product $[S_0]\left([y_0]-\hat{y}_0\right)$ in (7.6.1) is the source of the wrapping. If, for example, we choose $A_1 \in [S_0]$, then $A_1^{-1}[S_0]$ contains the identity matrix, and the transformation $\left(A_1^{-1}[S_0]\right)\left([y_0] - \hat{y}_0\right)$ does not cause a large overestimation if $w([S_0])$ is small and A_1 is well conditioned.

From (7.4), (7.6.2), and (7.6.3),

$$\begin{aligned} y_2 &= v_2 + S_1(y_1 - \hat{y}_1) \\ &= v_2 + (S_1 A_1)\, p_1 + S_1(v_1 - \hat{y}_1) \\ &\in [v_2] + ([S_1]\, A_1)\, [p_1] + [S_1]\, ([v_1] - \hat{y}_1) \\ &=: [v_2] + [q_2] =: [y_2]\,, \end{aligned}$$

where $[q_2] = ([S_1]\, A_1)\, [p_1] + [S_1]\, ([v_1] - \hat{y}_1)$. For the computation of $[y_3]$, we represent $y_2 \in [y_2]$ as

$$\begin{aligned} y_2 &= v_2 + (S_1 A_1)\, p_1 + S_1(v_1 - \hat{y}_1) \\ &= v_2 + A_2 \left\{ \left(A_2^{-1}(S_1 A_1)\right) p_1 + \left(A_2^{-1} S_1\right) (v_1 - \hat{y}_1) \right\} =: v_2 + A_2 p_2, \end{aligned}$$

where $A_2 \in \mathbb{R}^{n \times n}$ is regular and

$$\begin{aligned} p_2 &= \left(A_2^{-1}(S_1 A_1)\right) p_1 + \left(A_2^{-1} S_1\right) (v_1 - \hat{y}_1) \\ &\in \left(A_2^{-1}([S_1]\, A_1)\right) [p_1] + \left(A_2^{-1}\, [S_1]\right) ([v_1] - \hat{y}_1) =: [p_2]\,. \end{aligned}$$

Continuing in this way, we obtain the extended mean-value algorithm:

Algorithm 7.6.1

Initialize:

$[y_0]$, $\hat{y}_0 = m\,([y_0])$, $[p_0] = 0$, $A_0 = I$.

Input:

$[\tilde{y}_j]$, h_j, $\hat{y}_j$, A_j, $[p_j]$.

Compute:

$[v_{j+1}] = \hat{y}_j + \sum_{i=1}^{k-1} f^{[i]}(\hat{y}_j) h_j^i + f^{[k]}([\tilde{y}_j]) h_j^k$;

$[S_j] = I + \sum_{i=1}^{k-1} J\left(f^{[i]}; [y_j]\right) h_j^i$;

$[q_{j+1}] = ([S_j]\, A_j)\, [p_j] + [S_j]\, ([v_j] - \hat{y}_j)$;

$[y_{j+1}] = [v_{j+1}] + [q_{j+1}]$;

Choose A_{j+1} as in Algorithm 7.5.1;

$[p_{j+1}] = \left(A_{j+1}^{-1}([S_j]\, A_j)\right) [p_j] + \left(A_{j+1}^{-1}\, [S_j]\right) ([v_j] - \hat{y}_j)$;

$\hat{y}_{j+1} = m\,([v_{j+1}])$.

Output:

$\hat{y}_{j+1}$, $[y_{j+1}]$, A_{j+1}, $[p_{j+1}]$.

Rihm [37] claims that under the same conditions (same matrices A_j, midpoints $\hat{y}_j$, error inclusions $f^{[k]}([\tilde{y}_j])$, order of the Taylor series, and initial vectors $[y_0]$) Lohner's method produces better enclosures than Eijgenraam's method, and the extended mean-value method gives better enclosures than Lohner's method.

8. Stepsize and Order Control. We describe Eijgenraam's [14, pp. 129–136] stepsize and order control mechanisms. The 1994 implementation of AWA, currently available from Lohner's Web site[2], uses constant order and Eijgenraam's stepsize control.

[2]ftp://iamk4515.mathematik.uni-karlsruhe.de/pub/awa/

8.1. Stepsize Control. Eijgenraam [14, p. 87] uses the term local error to denote the over-estimation

$$\epsilon_{j+1} = q\left([y_{j+1}], y(t_{j+1}; t_j, [y_j])\right)$$

in the $(j+1)$-th step, where $q(\cdot,\cdot)$ is defined by (2.2.1). In order to distinguish from the standard numerical methods for ODEs, we refer to ϵ_{j+1} as the *local excess* [40] of the $(j+1)$-st step. Eijgenraam obtains an upper bound for ϵ_{j+1} and shows, under the assumptions that $cond\,(A_{j+1}) \approx 1$ and $\|w\left([y_j]\right)\|$ is small enough, that

$$\epsilon_{j+1} \approx \|w(f^{[k]}([\tilde{y}_j]))\| h_j^k = \frac{\left\|w\left(f^{[k]}([\tilde{y}_j])\right)\right\|}{h_j} h_j^{k+1}. \tag{8.1.1}$$

Eijgenraam argues that the quotient $\frac{\left\|w\left(f^{[k]}([\tilde{y}_j])\right)\right\|}{h_j}$ is nearly constant from step $j-1$ to step j and so approximates this quotient by

$$\frac{\left\|w\left(f^{[k]}([\tilde{y}_{j-1}])\right)\right\|}{h_{j-1}}. \tag{8.1.2}$$

For the $(j+1)$-st step, the order k for the Taylor series and the stepsize h_j are chosen such that the local excess ϵ_{j+1} satisfies

$$\epsilon_{j+1} \approx h_j \left\{E_a + E_r \|y(t; t_0, [y_0])\|\right\} \qquad \text{for } t \in [t_j, t_{j+1}], \tag{8.1.3}$$

where E_a and E_r are absolute and relative error tolerances. The value of $\|y(t; t_0, [y_0])\|$ in (8.1.3) is approximated by $\|\,[\tilde{y}_{j-1}]\,\|$, for $j \geq 1$. Let $E_0 = E_a + E_r \|\,[y_0]\,\|$ and $E_j = E_a + E_r \|\,[\tilde{y}_{j-1}]\,\|$ for $j \geq 1$. Then k and h_j are chosen such that

$$\epsilon_{j+1} \approx E_j h_j. \tag{8.1.4}$$

From (8.1.1), (8.1.2), and (8.1.4),

$$h_j(k) = \left(\frac{E_j h_{j-1}}{\left\|w\left(f^{[k]}([\tilde{y}_{j-1}])\right)\right\|}\right)^{1/k} \qquad \text{for } j \geq 1. \tag{8.1.5}$$

For $j = 0$, $\frac{\left\|w\left(f^{[k]}([\tilde{y}_j])\right)\right\|}{h_j}$ is approximated by $\|(k+1) f^{[k+1]}([y_0])\|$, and (8.1.5) becomes

$$h_0(k) = \left(\frac{E_0}{\|(k+1) f^{[k+1]}([y_0])\|}\right)^{1/k}.$$

8.2. Order Control. Denote the order of the Taylor series employed in the $(j+1)$-st step by k_j. For the first two steps, Eijgenraam prescribes values k_0 and k_1 such that $|k_0 - k_1| = 1$. To determine k_j for $j \geq 2$, he considers two possible orders: the order k_{j-1} of the preceding step and a tentative order k_j', which he computes by the following heuristic:

- if the order was increased on the previous step, consider a further increase of the order; if the order was decreased on the previous step, then consider a further decrease of the order; that is,

$$k_j' = \begin{cases} k_{j-1} + 1 & \text{if } k_{j-1} > k_{j-2} \\ k_{j-1} - 1 & \text{if } k_{j-1} < k_{j-2} \end{cases}$$

- if the order was not changed on the previous step but an increase was considered, then consider a decrease on this step; if the order was not changed on the previous step but a decrease was considered, then consider an increase on this step; that is, if $k_{j-1} = k_{j-2}$, then

$$k'_j = \begin{cases} k_{j-1} - 1 & \text{if } k'_{j-1} > k_{j-2} \\ k_{j-1} + 1 & \text{if } k'_{j-1} < k_{j-2}, \end{cases}$$

where $k'_1 = k_1$.

If k'_j is not in a prescribed range, for example, $k'_j \notin [2, k_{\max}]$, then $k_j = k_{j-1}$. Otherwise, the value $k \in \{k_{j-1}, k'_j\}$ is chosen to maximize

$$\frac{h_j(k)}{W_k},$$

where W_k is the amount of work involved in one integration step with order k. Lohner [27] suggests two ways for obtaining W_k:

- compute the first time step with different orders and measure the elapsed time;
- count all operations in the Taylor series method of order k and add up their relative costs, which can be done automatically.

In passing, note that, if we are prepared to take a step of order k, we can easily take a step of order $k' < k$ if the lower order step gives a tighter enclosure. In addition, note that we could modify the techniques outlined above to optimize the tightness of enclosures, rather than the cost of the integration.

9. Defect-Controlled Solutions. The methods we discussed above produce an enclosure of the true solution. In this section, we describe a defect-controlled method [6] that produces an approximate solution which is the exact solution to a nearby problem. For a suitable problem and any user supplied tolerance, the method can ensure that the defect is less than the tolerance.

Let $\tilde{y}(t)$ be a continuous representation of a computed approximate numerical solution to

$$y' = f(y), \qquad y(t_0) = y_0. \tag{9.1}$$

The defect of $\tilde{y}(t)$ is defined as

$$\delta(t) = \tilde{y}'(t) - f(\tilde{y}(t)). \tag{9.2}$$

Then $\tilde{y}(t)$ satisfies exactly the nearby problem

$$\tilde{y}'(t) = f(\tilde{y}(t)) + \delta(t).$$

Defect-controlled methods, based on computing an asymptotic estimate of the defect, are considered in [15, 16, 19]. We present a defect-controlled method [6] based on computing an enclosure of some norm of the defect $\delta(t)$ of $\tilde{y}(t)$. This enclosure is used to control the stepsize of the numerical solution of (9.1). The strategy for choosing a stepsize follows from the requirement to keep the defect small. That is, in the $(j+1)$-st step, h_j is selected such that $\|\delta(t)\| \leq \epsilon$ for $t \in [t_j, t_{j+1}]$. This defect-controlled algorithm can be described in the following algorithm [6]:

Algorithm 9.1

for $j = 0, 1, \ldots,$
  compute a continuous approximate solution $\tilde{y}(t)$ in $[t_j, t_j + h_j]$;
  $\delta(t) := \tilde{y}'(t) - f(t, \tilde{y}(t))$ in $[t_j, t_j + h_j]$;
  $[E] :=$ enclosure of $\|\delta(t)\|$ in $[t_j, t_j + h_j]$;
  if $\bar{E} > \epsilon$ **then**
    reduce h_j;
  else
    if $\bar{E} \ll \epsilon$ **then**
      increase h_j;
    else
      accept h_j;
      $t_{j+1} = t_j + h_j$;
    end-if
  end-if
end-for
end.

In the standard approaches [16], the estimate of the defect is asymptotically correct as the stepsize tends to zero. With this approach, the defect is guaranteed to be uniformly below some tolerance. Therefore, $\tilde{y}(t)$ is an exact solution to a problem whose "distance" from the specified problem is guaranteed to be small.

10. Some Directions for Future Research. Compared to the standard numerical methods for IVPs for ODEs, little work has been done for the validated solution of ODEs. The most advanced (and probably the only) validated solver for ODEs is AWA [26]. As Corliss points out [9], "there is no widely available portable, production-quality software for validated solution of ODEs". Such software should allow researchers to experiment easily with interval algorithms for ODEs and should be:

- portable;
- modular;
- easy-to-modify; and
- able to solve large problems.

We list some of the problems on which further research can be done. For a more complete discussion, see [8, 9].

Validating an Enclosure. AWA uses constant a priori enclosures to validate the existence of a unique solution. As a consequence, the stepsize is restricted to Euler steps, independently of the order of the Taylor series used in the second algorithm. The polynomial and Taylor series method are more expensive computationally, but they allow larger stepsizes to be taken. There are no implementations of these two methods in widely available software. Furthermore, it is not known which of them would give better results.

Computing a Tight Enclosure. We presented some methods for computing tight enclosures. There is no best solution: for different problems, different methods will be suitable.

Stepsize and Order Control. AWA uses fixed step size and order. Recently, Lohner [27] advocated Eijgenraam's stepsize and order control strategies.

Kerbl [21] proposes a stepsize control strategy based on information about the behavior of an approximate solution. The idea is that if we know that the trajectories contract, we can take larger stepsizes, otherwise, we have to proceed with smaller stepsizes.

Initial Guess for the Stepsize. We need a good initial guess for the stepsize. One could use Eijgenraam's initial stepsize prediction. Other options are to try to adapt techniques from the standard methods for ODEs.

Quasi-monotone problems. Many problems of interest are quasi-monotone, at least locally, and one-dimensional problems are quasi-monotone. Hence, a complete suite of tools for the validated solution to ODEs should include software for quasi-monotone problems. No such software now exists.

Stiff Problems. Lohner's algorithm handles many problems of moderate stiffness. It is not strictly speaking a on-step method because it uses information on the entire interval, and it also uses information from the Jacobian of f. However, validated methods specifically suited to stiff problems have yet to be developed. Griewank [18] proposes an approximate method using implicit Taylor series which may have a validated generalization.

Differential Algebraic Equations. Chang and Corliss [5] showed how to generate a Taylor series solution for the simple pendulum differential algebraic equation (DAE). Then Corliss and Lodwick [11] showed how to use interval techniques to obtain a validated enclosure for a simple linear DAE using Lohner's AWA program. The constraints in a semi-explicit system of DAEs are used to enforce consistency of user-supplied initial conditions, tighten enclosures of state variables, and insure consistency at subsequent steps. Corliss and Lodwick [10] apply these techniques to compute a validated enclosure of the Simple Pendulum Problem. Much work remains to be done to write production-quality software implementing these ideas.

Acknowledgments. The authors would like to thank the late Prof. T. E. Hull for his careful reading of this paper and valuable suggestions.

The authors would also like to thank Dr. J. Pryce. The discussions with him during his visit to Toronto helped us understand the subject better. His comments considerably improved the presentation of the material.

REFERENCES

[1] E. ADAMS, D. CORDES, AND R. LOHNER, *Enclosure of solutions of ordinary initial value problems and applications*, in Discretization in Differential Equations and Enclosures, E. Adams, R. Ansorge, C. Großmann, and H. G. Roos, eds., Akademie-Verlag, Berlin, 1987, pp. 9–28.

[2] G. ALEFELD AND J. HERZBERGER, *Introduction to Interval Computations*, Academic Press, New York, 1983.

[3] H. BAUCH AND W. KIMMEL, *Solving ordinary initial value problems with guaranteed bounds*, Z. angew. Math. Mech., 69 (1989), pp. T110–T112.

[4] M. BERZ, C. BISCHOF, G. F. CORLISS, AND A. GRIEWANK, eds., *Computational Differentiation: Techniques, Applications, and Tools*, SIAM, Philadelphia, Penn., 1996.

[5] Y. F. CHANG AND G. F. CORLISS, *ATOMFT: Solving ODEs and DAEs using Taylor series*, Computers and Mathematics with Applications, 28 (1994), pp. 209–233.

[6] R. M. CORLESS AND G. F. CORLISS, *Rationale for guaranteed ODE defect control*, in Computer Arithmetic and Enclosure Methods, L. Atanassova and J. Herzberger, eds., North-Holland, Amsterdam, 1992, pp. 3–12.

[7] G. F. CORLISS, *Survey of interval algorithms for ordinary differential equations*, Appl. Math. Comput., 31 (1989), pp. 112–120.

[8] ———, *Guaranteed error bounds for ordinary differential equations*, in Theory of Numerics in Ordinary and Partial Differential Equations, M. Ainsworth, J. Levesley, W. Light, and M. Marletta, eds., Oxford University Press, 1995, pp. 1–76.

[9] ———, *Where is validated ODE solving going*, in Proceedings of the IMACS-GAMM International Symposium on Numerical Methods and Error Bounds, J. Herzberger, ed., Akademie Verlag, Berlin, 1996, pp. 48–57.

[10] G. F. CORLISS AND W. LODWICK, *Role of constraints in the validated solution of DAEs*, Technical Report 430, Marquette University Department of Mathematics, Statistics, and Computer Science, Milwaukee, Wisc., March 1996.

[11] G. F. CORLISS AND W. A. LODWICK, *Correct computation of solutions of differential algebraic control equations*, Z. angew. Math. Mech., special issue on Numerical Analysis, Scientific Computing, and Computer Science, (1996), pp. 37–40.

[12] G. F. CORLISS AND R. RIHM, *Validating an a priori enclosure using high-order Taylor series*, in Scientific Computing, Computer Arithmetic, and Validated Numerics, G. Alefeld and A. Frommer, eds., Akademie Verlag, Berlin, 1996, pp. 228–238.

[13] M. EIERMANN, *Adaptive Berechnung von Integraltransformationen mit Fehlerschranken*, PhD thesis, Universität Freiburg, Freiburg, 1989.

[14] P. EIJGENRAAM, *The Solution of Initial Value Problems Using Interval Arithmetic*, Mathematical Centre Tracts No. 144, Stichting Mathematisch Centrum, Amsterdam, 1981.

[15] W. H. ENRIGHT, *Analysis of error control strategies for continuous Runge-Kutta methods*, SIAM J. Numerical Analysis, 26 (1989), pp. 588–599.

[16] ———, *A new error-control for initial value solvers*, Applied Mathematics and Computation, 31 (1989), pp. 288–301.

[17] A. GRIEWANK AND G. F. CORLISS, eds., *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, SIAM, Philadelphia, Penn., 1991.

[18] A. GRIEWANK, G. F. CORLISS, P. HENNEBERGER, G. KIRLINGER, F. A. POTRA, AND H. J. STETTER, *High-order stiff ODE solvers via automatic differentiation and rational prediction*, Numer. Math., (submitted).

[19] P. M. HANSON AND W. H. ENRIGHT, *Controlling the defect in existing variable-order Adams codes for initial-value problems*, ACM Trans. Math. Soft., 9 (1983), pp. 71–97.

[20] L. JACKSON, *Interval arithmetic error–bounding algorithms*, SIAM J. Numer. Anal., 12 (1975), pp. 223–238.

[21] M. KERBL, *Stepsize strategies for inclusion algorithms for ODE's*, in Computer Arithmetic, Scientific Computation, and Mathematical Modelling, E. Kaucher, S. Markov, and G. Mayer, eds., IMACS Annals on Computing and Appl. Math. 12, J.C. Baltzer, Basel, 1991.

[22] F. KRÜCKEBERG, *Ordinary differential equations*, in Topics in Interval Analysis, E. Hansen, ed., Clarendon Press, Oxford, 1969, pp. 91–97.

[23] U. W. KULISCH AND W. L. MIRANKER, *Computer Arithmetic in Theory and Practice*, Academic Press, New York, 1981.

[24] R. J. LOHNER, *Anfangswertaufgaben im kompakten Mengen für Anfangswerte und Parameter*, diplomarbeit, Inst. f. Angew. Math., Universität Karlsruhe, 1978.

[25] R. J. LOHNER, *Enclosing the solutions of ordinary initial and boundary value problems*, in Computer Arithmetic: Scientific Computation and Programming Languages, E. W. Kaucher, U. W. Kulisch, and C. Ullrich, eds., Wiley-Teubner Series in Computer Science, Stuttgart, 1987, pp. 255–286.

[26] R. J. LOHNER, *Einschließung der Lösung gewöhnlicher Anfangs– und Randwertaufgaben und Anwendungen*, PhD thesis, Universität Karlsruhe, 1988.

[27] ———, *Step size and order control in the verified solution of IVP with ODE's*, 1995. SciCADE'95 International Conference on Scientific Computation and Differential Equations, Stanford, Calif., March 28 – April 1, 1995.

[28] R. E. MOORE, *The automatic analysis and control of error in digital computation based on the use of interval numbers*, in Error in Digital Computation, Vol. I, L. B. Rall, ed., Wiley, New York, 1965, pp. 61–130.

[29] ———, *Automatic local coordinate transformations to reduce the growth of error bounds in interval computation of solutions of ordinary differential equations*, in Error in Digital Computation, Vol. II, L. B. Rall, ed., Wiley, New York, 1965, pp. 103–140.

[30] ———, *Interval Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1966.

[31] ———, *A survey of interval methods for differential equations*, in Proceedings of the 23rd Converence on Decision and Control (Las Vegas, 1984), IEEE, 1984, pp. 1529–1535.

[32] A. NEUMAIER, *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, 1990.

[33] K. NICKEL, *Bounds for the set of solutions of functional–differential equations*, Freiburger Intervall–Berichte, 9 (1979). Also appeared as MRC Techn. Summary Report No. 1782, Univ. of Wisconsin, Madison, 1977.

[34] ———, *How to fight the wrapping effect*, in Interval Analysis 1985, K. Nickel, ed., Lecture Notes in Computer Science No. 212, Springer, Berlin, 1985, pp. 121–132.

[35] ———, *Using interval methods for the numerical solution of ODE's*, Z. angew. Math. Mech., 66 (1986), pp. 513–523.

[36] L. B. RALL, *Automatic Differentiation: Techniques and Applications*, vol. 120 of Lecture Notes in Computer Science, Springer Verlag, Berlin, 1981.

[37] R. RIHM, *Interval methods for initial value problems in ODEs*, in Topics in Validated Computations: Proceedings of the IMACS-GAMM International Workshop on Validated Computations, University of Oldenburg, J. Herzberger, ed., Elsevier Studies in Computational Mathematics, Elsevier, Amsterdam, New York, 1994.

[38] ———, *On a class of enclosure methods for initial value problems*, Computing, 53 (1994), pp. 369–377.

[39] H. SPREUER AND E. ADAMS, *On the existence and the verified determination of homoclinic and heteroclinic orbits of the origin for the Lorenz system*, Computing Suppl., 9 (1993), pp. 233–246.

[40] H. J. STETTER, *Algorithms for the inclusion of solutions of ordinary initial value problems*, in Equadiff 6: Proceedings of the International Conference on Differential Equations and Their Applications (Brno, 1985), J. Vosmanský and M. Zlámal, eds., vol. 1192 of Lecture Notes in Mathematics, Springer Verlag, Berlin, 1986, pp. 85–94.

[41] ———, *Validated solution of initial value problems for ODEs*, in Computer Arithmetic and Self-Validating Numerical Methods, C. Ullrich, ed., Academic Press, New York, 1990, pp. 171–187.

[42] W. WALTER, *Differential and Integral Inequalities*, Springer, Heidelberg, 1970.