, , 1–17 ()

An Effective High-Order Interval Method for Validating Existence and Uniqueness of the Solution of an IVP for an ODE*

NEDIALKO S. NEDIALKOV ** nedialk@mcmaster.ca
Department of Computing and Software, McMaster University, Hamilton, Ontario, L8S 4L7, Canada.

KENNETH R. JACKSON krj@cs.toronto.edu
Department of Computer Science, University of Toronto, Toronto, Ontario, M5S 3G4, Canada.

JOHN D. PRYCE pryce@rmcs.cranfield.ac.uk
Computer Information Systems Engineering Department, Cranfield University, RMCS Shrivenham, Swindon SN6 8LA, UK.

Abstract. Validated methods for initial value problems for ordinary differential equations produce bounds that are guaranteed to contain the true solution of a problem. When computing such bounds, these methods verify that a unique solution to the problem exists in the interval of integration and compute a priori bounds for the solution in this interval. A major difficulty in this verification phase is how to take as large a stepsize as possible, subject to some tolerance requirement. We propose a high-order enclosure method for proving existence and uniqueness of the solution and computing a priori bounds.

1. Introduction

We consider the set of autonomous initial value problems (IVPs)

$$y'(t) = f(y) \tag{1}$$
$$y(t_0) \in [y_0], \tag{2}$$

where $t \in [t_0, T]$ for some $T > t_0$. Here t_0 and $T \in \mathbb{R}$, $f \in C^{k-1}(\mathcal{D})$, $k \geq 2$, $\mathcal{D} \subseteq \mathbb{R}^n$ is open, $f : \mathcal{D} \to \mathbb{R}^n$, and $[y_0] \subseteq \mathcal{D}$. The condition (2) 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. For expositional convenience, we consider only autonomous systems. This is not a restriction of consequence since a nonautonomous system of ordinary differential

* This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, the Information Technology Research Centre of Ontario, and Communications and Information Technology Ontario.
** Most of this work was done when N. S. Nedialkov was at the Department of Computer Science of the University of Toronto.

equations (ODEs) can be converted into an autonomous system. Moreover, the methods discussed here can be extended easily to nonautonomous systems.

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 (1) with an initial condition y_j at t_j by $y(t; t_j, y_j)$. For an interval vector $[y_j]$, we denote by $y(t; t_j, [y_j])$ the set of solutions

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

We are interested in computing interval vectors $[y_j]$, $j = 1, \ldots, m$, that are guaranteed to contain the solution of (1–2) at $t_1, t_2, \ldots, t_m$. That is,

$$y(t_j; t_0, [y_0]) \subseteq [y_j]\,, \quad \text{for } j = 1, \ldots, m$$

[9]. In the methods that we consider, the $(j+1)$st step consists of two phases, which we refer to as Algorithm I and Algorithm II [4, 6, 11]:

Algorithm I: validate existence and uniqueness of the solution in $[t_j, t_{j+1}]$ and compute an enclosure $[\tilde{y}_j]$ such that

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

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

$$y(t_{j+1}; t_0, [y_0]) \subseteq [y_{j+1}]\,.$$

(These methods do not include Berz and Makino's scheme [1], which uses high-order Taylor series expansions in both time and the initial conditions, in contrast to the the methods considered in this paper, which employ high-order Taylor series with respect to time only. Another significant difference is that Berz and Makino's scheme validates existence and uniqueness and also computes tight bounds on the solution at each grid point t_j in one phase, while the methods considered here separate these tasks into Algorithms I and II, as noted above.)

This paper addresses Algorithm I: given an enclosure $[y_j]$ of the solution of (1–2) at t_j, find h_j and $[\tilde{y}_j]$ such that the IVP

$$y'(t) = f(y), \quad y(t_j) = y_j \tag{3}$$

has a unique solution

$$y(t; t_j, y_j) \in [\tilde{y}_j] \quad \text{for all } t \in [t_j, t_j + h_j] \quad \text{and all } y_j \in [y_j].$$

(A detailed treatment of validated methods for IVPs for ODEs can be found in [9, 10].)

In a validated ODE solver, we normally predict a stepsize from some tolerance requirement [6]. Then, we attempt to verify existence and uniqueness of the solution with this stepsize in Algorithm I. The implementation of Algorithm II is usually based on a high-order method, such as interval Taylor series [4, 6, 11], or the recently proposed interval Hermite-Obreschkoff method [6, 7]. In both cases, the predicted stepsize depends on the order of the method in Algorithm II. However, Algorithm I may decrease the input stepsize. This happens frequently in the AWA [4] and ADIODES [11] solvers. Often, the stepsize restriction from Algorithm I is much more severe than the tolerance or stability limitation imposed by Algorithm II [6]. (In a forthcoming paper [8], we investigate the stability of interval methods in more detail.) We refer to the standard method employed in Algorithm I as a first-order enclosure (FOE) method and describe it briefly later.

In [6], the first author implemented a high-order Taylor series method, based on a theorem by Corliss and Rihm [2], to validate existence and uniqueness of the solution in Algorithm I. He showed empirically that this method enables larger stepsizes than the FOE method. In this paper, we propose a similar high-order enclosure (HOE) method, based on the same theorem. Like the original HOE method, the new version allows larger stepsizes than the FOE method. However, our current implementation is simpler than the one suggested in [6]. In addition, the HOE method does not require significantly more work than the FOE method.

The rest of this paper is organized as follows. After we introduce some notation in the next section, we present the basis for the FOE method in section 3. In section 4, we derive our HOE method, describe a stepsize selection scheme that can be used with it, and give an algorithmic description of this method. In section 5, we present numerical results which show that a validated solver with the HOE method can take fewer steps than with the FOE method.

2. Notation

An interval or an interval vector is denoted by $[a]$. An interval $[a] \equiv [\underline{a}, \bar{a}]$ is symmetric if $\underline{a} = -\bar{a}$. An interval vector is symmetric if each of its components is symmetric.

For an interval $[a] = [\underline{a}, \bar{a}]$, we define its width and magnitude by

$$\begin{aligned} w([a]) &= \bar{a} - \underline{a} \quad \text{and} \\ |[a]| &= \max\{ |\underline{a}|, |\bar{a}| \}, \end{aligned}$$

respectively. We define width and magnitude componentwise for interval vectors.

We define the sequence of functions $f^{[i]}(y)$, $i = 0, 1, 2, \ldots$, by

$$f^{[0]}(y) = y, \tag{4}$$

$$f^{[i]}(y) = \frac{1}{i}\left(\frac{\partial f^{[i-1]}}{\partial y} f\right)(y), \qquad \text{for } i \geq 1. \tag{5}$$

For the autonomous IVP (3), $f^{[i]}(y_j) = y^{(i)}(t_j)/i!$. That is, $f^{[i]}(y_j)$ denotes the ith Taylor coefficient of $y(t; t_j, y_j)$ expanded around t_j.

The interior of an interval vector $[a]$, $\mathrm{int}([a])$, is the set of vectors $a \in [a]$ for which no component of a is an endpoint of the corresponding component of $[a]$.

3. The First-Order Enclosure Method

Using the Picard-Lindelöf operator and the Banach fixed-point theorem, one can show [3, 4] that, if h_j and $[\tilde{y}_j]$ satisfy

$$[\tilde{y}_j^1] \equiv [y_j] + [0, h_j]\, f\left([\tilde{y}_j]\right) \subseteq [\tilde{y}_j]\,, \tag{6}$$

then (3) has a unique solution $y(t; t_j, y_j) \in [\tilde{y}_j^1]$ for all $t \in [t_j, t_j + h_j]$ and all $y_j \in [y_j]$.

We refer to a method based on (6) as a first-order enclosure (FOE) method. It can be easily implemented, but a serious disadvantage of this method is that it often severely restricts the stepsize a high-order method in Algorithm II could otherwise use.

4. The High-Order Enclosure Method

Our HOE method is based on the following theorem of Corliss and Rihm [2, Theorem 3].

Theorem 1 *Suppose that $y_j \in \operatorname{int}([\tilde{y}_j])$ where $[\tilde{y}_j] \subseteq \mathcal{D}$. If*

$$\sum_{i=0}^{k-1} (t - t_j)^i f^{[i]}(y_j) + (t - t_j)^k f^{[k]}([\tilde{y}_j]) \subseteq [\tilde{y}_j] \tag{7}$$

for all $t \in [t_j, t_j + h_j]$, then

$$y(t; t_j, y_j) \in \sum_{i=0}^{k-1} (t - t_j)^i f^{[i]}(y_j) + (t - t_j)^k f^{[k]}([\tilde{y}_j]) \subseteq [\tilde{y}_j]$$

for all $t \in [t_j, t_j + h_j]$.

Theorem 1 ensures the existence of a solution $y(t; t_j, y_j)$ of problem (3) for all $t \in [t_j, t_j + h_j]$ and for all $y_j \in [y_j] \subseteq \operatorname{int}([\tilde{y}_j])$. Although not mentioned explicitly in [2], it is easy to see that this solution is unique. To this end, note that Corliss and Rihm [2] assume at the start of their paper that $f \in C^{k-1}(\mathcal{D})$ for some $k \geq 2$, as do we. Therefore, $f(y)$ is uniformly Lipschitz continuous for $y \in [\tilde{y}_j]$. Since $y(t; t_j, y_j) \in [\tilde{y}_j]$ for all $t \in [t_j, t_j + h_j]$, it follows immediately that $y(t; t_j, y_j)$ is the unique solution of problem (3) for $t \in [t_j, t_j + h_j]$.

Note that Theorem 1 and the argument above ensures the existence and uniqueness of the solution $y(t; t_j, y_j)$ of problem (3) for all $t \in [t_j, t_j + h_j]$ and for all $y_j \in [y_j] \subseteq \operatorname{int}([\tilde{y}_j])$. Therefore, for given $[y_j]$,

- if we find h_j and $[\tilde{y}_j]$ such that $[y_j] \subseteq \operatorname{int}([\tilde{y}_j])$, and
- (7) holds for all $y_j \in [y_j]$,

then we have verified existence and uniqueness of the solution $y(t; t_j, y_j)$ of problem (3) for all $t \in [t_j, t_j + h_j]$ and all $y_j \in [y_j]$. Moreover, $y(t; t_j, y_j) \in [\tilde{y}_j]$ for all $t \in [t_j, t_{j+1}]$ and all $y_j \in [y_j]$.

4.1. The Validation Method

Denote by $h_{j,0} > 0$ the stepsize input to Algorithm I on the $(j+1)$st step $(j \geq 0)$. Let

$$p_{k-1}(t; t_j, y_j) \;=\; \sum_{i=0}^{k-1} (t - t_j)^i f^{[i]}(y_j), \tag{8}$$

and $[p_j]$ be an interval vector enclosing the set

$$\mathcal{P}_j = \{\, p_{k-1}(t; t_j, y_j) \mid t \in [t_j, t_j + h_{j,0}], \quad y_j \in [y_j] \,\}\,. \tag{9}$$

That is, $\mathcal{P}_j \subseteq [p_j]$. Let $[u_j]$ be such that

$$[\tilde{y}_j] \;=\; [p_j] + [u_j], \quad \text{and} \tag{10}$$

$$[y_j] \;\subseteq\; \operatorname{int}([\tilde{y}_j]). \tag{11}$$

(In subsections 4.2 and 4.3, we consider how to compute $[p_j]$ and $[u_j]$.)

Lemma 1 *If $h_{j,1}$ is such that*

$$[0, h_{j,1}^k] f^{[k]}([\tilde{y}_j]) \subseteq [u_j], \tag{12}$$

and

$$h_j = \min\{\, h_{j,0}, h_{j,1} \,\}, \tag{13}$$

then there exists a unique solution $y(t; t_j, y_j)$ to the problem (3) for all $t \in [t_j, t_j + h_j]$ and all $y_j \in [y_j]$. Moreover,

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

for all $t \in [t_j, t_j + h_j]$ and all $y_j \in [y_j]$.

Proof. Using (8–13), we obtain for any $t \in [t_j, t_j + h_j]$ and any $y_j \in [y_j]$ that

$$\begin{aligned} p_{k-1}(t; t_j, y_j) \;&+\; (t - t_j)^k f^{[k]}([\tilde{y}_j]) \\ &\subseteq\; [p_j] + [0, h_j^k] f^{[k]}([\tilde{y}_j]) \\ &\subseteq\; [p_j] + [0, h_{j,1}^k] f^{[k]}([\tilde{y}_j]) \\ &\subseteq\; [p_j] + [u_j] \\ &=\; [\tilde{y}_j]\,. \end{aligned}$$

Hence, for any $y_j \in [y_j]$, (7) holds with h_j from (13) and $[\tilde{y}_j]$ from (10), and

$$y(t; t_j, y_j) \in \sum_{i=0}^{k-1} (t - t_j)^i f^{[i]}(y_j) + (t - t_j)^k f^{[k]}([\tilde{y}_j]) \subseteq [\tilde{y}_j^1] \subseteq [\tilde{y}_j]$$

for all $y_j \in [y_j]$ and all $t \in [t_j, t_j + h_j]$.

Therefore, by Theorem 1 and our comment following it, $y(t; t_j, y_j)$ is the unique solution of problem (3) for all $t \in [t_j, t_j + h_j]$ and all $y_j \in [y_j]$.

□

Our validation algorithm is based on verifying (12). As such, it can be viewed as a practical implementation of some of Stetter's ideas presented in [12].

In subsections 4.2 and 4.3, we consider how we compute $[p_j]$ and $[u_j]$, respectively. In subsection 4.4, we describe the stepsize control scheme that we have used to produce the numerical results in this paper and discuss an alternative stepsize control strategy that can be employed with the HOE method. Finally, in subsection 4.5, we give the validation algorithm.

4.2. Computing $[\boldsymbol{p_j}]$

We use an interval form of Horner's rule to compute $[p_j]$. For given $h_{j,0}$ and $[y_j]$, obviously,

$$\begin{aligned} \mathcal{P}_j &= \left\{ \sum_{i=0}^{k-1} (t-t_j)^i f^{[i]}(y_j) \mid t \in [t_j, t_j + h_{j,0}], \quad y_j \in [y_j] \right\} \\ &\subseteq [y_j] + [0, h_{j,0}] \Big(f^{[1]}([y_j]) + \cdots \qquad\qquad (14) \\ &\qquad + [0, h_{j,0}] \big(f^{[k-2]}([y_j]) + [0, h_{j,0}] f^{[k-1]}([y_j]) \big) \cdots \Big) \equiv [p_j]. \end{aligned}$$

Computing $[p_j]$ from (14) is simple. However, this approach may produce a large overestimation for the set $\mathcal{P}_j$. In Appendix A.1, we discuss an alternative approach for enclosing $\mathcal{P}_j$ that may often give tighter bounds than $[p_j]$ from (14).

4.3. Predicting $[\boldsymbol{u_j}]$

Suppose that we have completed a step, have an a priori enclosure of the solution $[\tilde{y}_{j-1}]$ in $[t_{j-1}, t_j]$, and have predicted a stepsize $h_{j,0}$ for the next step. Before we describe how to compute $[u_j]$, we note that if $[u_j]$ is symmetric with no component equal to $[0,0]$, then (11) holds. Then, ensuring that (12) is satisfied is surprisingly simple: since $[u_j]$ is symmetric, and each component is nonzero, we can always compute $h_{j,1} > 0$. In a floating-point implementation, though, the real $h_{j,1}$ may underflow to zero. In this case, or if $h_{j,1}$ is smaller than the minimum stepsize allowed, we can notify the user that this algorithm cannot verify existence and uniqueness.

We assume that the kth Taylor coefficient, $f^{[k]}$, does not change much from step to step. That is, $f^{[k]}([\tilde{y}_j]) \approx f^{[k]}([\tilde{y}_{j-1}])$. Therefore, we approximate $h^k_{j,0} f^{[k]}([\tilde{y}_j])$ by $h^k_{j,0} f^{[k]}([\tilde{y}_{j-1}])$. Note that we use $[\tilde{y}_{j-1}]$, not $[\tilde{y}^1_{j-1}] \subseteq [\tilde{y}_{j-1}]$. One reason is that $[\tilde{y}^1_{j-1}]$ is usually not much tighter than $[\tilde{y}_{j-1}]$. In particular, if $h_{j-1} = h_{j-1,1}$ is such that $[0, h^k_{j-1,1}] f^{[k]}([\tilde{y}_{j-1}]) = [u_{j-1}]$, then $[\tilde{y}^1_{j-1}] = [\tilde{y}_{j-1}]$; see (12) and subsection 4.5. Another reason is that computing $f^{[k]}([\tilde{y}_{j-1}])$ is expensive and we want to avoid recomputing $f^{[k]}$ with $[\tilde{y}^1_{j-1}]$.

Since $[u_j]$ must enclose the set

$$\left\{ (t-t_j)^k f^{[k]}(y(t;t_j,y_j)) \mid t \in [t_j, t_j + h_{j,0}], \quad y_j \in [y_j] \right\}, \tag{15}$$

each component of $[u_j]$ must contain the minimum and maximum of the corresponding component of $(t-t_j)^k f^{[k]}(y(t;t_j,y_j))$ for all $t \in [t_j, t_j + h_{j,0}]$ and any $y_j \in [y_j]$. Thus, we set $\beta_{j-1} = |f^{[k]}([\tilde{y}_{j-1}])|$ and use $h_{j,0}^k[-\beta_{j-1}, \beta_{j-1}]$ as the basis for computing $[u_j]$. To increase the likelihood that we enclose (15), we inflate $h_{j,0}^k[-\beta_{j-1}, \beta_{j-1}]$ by an arbitrary factor of 2 and initialize

$$[u_j] = 2h_{j,0}^k \, [-\beta_{j-1}, \beta_{j-1}].$$

However, if some component of β_{j-1} is zero, then the corresponding component of $[u_j]$ is initially $[0,0]$. In this case, we reset this component of $[u_j]$ to $[-\epsilon, \epsilon]$ for some small positive parameter ϵ. The tolerance, for example, is a reasonable choice for ϵ.

Clearly, $[u_j]$ may be an overestimation of (15). However, this is not a serious disadvantage, because, if we reduce $h_{j,0}$ by a factor $\alpha < 1$, then we reduce $\|w\,([u_j])\,\|$ by a factor α^k.

Since on the first step we do not have $[\tilde{y}_0]$ available, we compute $[u_0]$ as follows. Let

$$[v_0] = [p_0] + 2h_{0,0}^k \, [-1,1] \, |f^{[k]}([y_0])|$$

and $\beta_{-1} = |f^{[k]}([v_0])|$. We initialize

$$[u_0] = 2h_{0,0}^k \, [-\beta_{-1}, \beta_{-1}]$$

and modify any zero components of $[u_0]$ as described above.

4.4. The HOE Method with Stepsize Control

When comparing the FOE and HOE methods in section 5, we use an interval Taylor series method [6] in Algorithm II and a stepsize controller as described below.

Let

$$err_j = h_{j-1}^k \|w(f^{[k]}([\tilde{y}_{j-1}]))\| \quad (j \geq 1),$$

where $\|\cdot\|$ is the max norm, and let Tol be the tolerance specified by the user. Here, this tolerance is a bound on the local excess, err_j, per unit step (LEPUS) [6]. That is, we satisfy on each successful step

$$err_j \leq h_{j-1} Tol.$$

On the first attempt on the first step, we compute

$$h_{0,0} = \left(\frac{Tol}{\|(k+1) f^{[k+1]}([y_0])\|} \right)^{1/k} \tag{16}$$

(see [9]). Subsequently, we set

$$h_{j,0} = h_{j-1} \left(\frac{h_{j-1} Tol}{err_j} \right)^{1/(k-1)} \tag{17}$$

after a rejected step, and

$$h_{j,0} = 0.9\, h_{j-1} \left(\frac{0.5\, h_{j-1} Tol}{err_j} \right)^{1/(k-1)} \tag{18}$$

after a successful step. For more details, see [6].

We have also considered the following alternative stepsize control strategy. We have from (12–13)

$$h_j^k f^{[k]}([\tilde{y}_j]) \subseteq [0, h_{j,1}^k] f^{[k]}([\tilde{y}_j]) \subseteq [u_j],$$

which implies

$$h_j^k \|w(f^{[k]}([\tilde{y}_j]))\| \le \|w([u_j])\|. \tag{19}$$

We can set each component of $[u_j]$ to be $h_{j,0}[-Tol/2, Tol/2]$. Then (19) becomes

$$h_j^k \|w(f^{[k]}([\tilde{y}_j]))\| \le h_{j,0}\, Tol. \tag{20}$$

If $h_j = h_{j,0} \le h_{j,1}$, (20) ensures that the method satisfies the LEPUS criterion [6], but it may happen that $h_j = h_{j,1} < h_{j,0}$. Then (20) becomes

$$h_{j,1}^k \|w(f^{[k]}([\tilde{y}_j]))\| \le h_{j,0}\, Tol,$$

which does not imply

$$h_{j,1}^k \|w(f^{[k]}([\tilde{y}_j]))\| \le h_{j,1}\, Tol.$$

However, if $h_{j,1} \approx h_{j,0}$, then

$$h_{j,0}^k \|w(f^{[k]}([\tilde{y}_j]))\| \lessapprox h_{j,0}\, Tol,$$

and we have an "almost" LEPUS.

With the stepsize control scheme (17–18), the HOE method usually verifies existence and uniqueness of the solution with a stepsize that is equal to or approximately equal to the input stepsize (see subsections 5.1 and 5.2). That is, either $h_j = h_{j,0}$ or $h_j < h_{j,0}$ with $h_j \approx h_{j,0}$.

Instead of predicting a stepsize from (17–18), we can select for the next step $h_{j+1,0} = h_{j,1}$. This is reasonable since, if $h_{j,1} > h_{j,0}$, we have verified existence and uniqueness with $h_{j,0}$, but we could have possibly verified existence and uniqueness with a stepsize between $h_{j,0}$ and $h_{j,1}$. That is, we assume that we might be successful on the next step with $h_{j+1,0} = h_{j,1} > h_{j,0}$. Similar considerations apply when $h_{j,1} < h_{j,0}$. We can be "more conservative" and always choose $h_{j+1,0} = (h_{j,0} + h_{j,1})/2$, for example.

We do not report numerical results with this stepsize control strategy, but we have performed experiments which show that, if we take $h_{j+1,0} = h_{j,1}$ as input to the HOE method, then either $h_{j+1} = h_{j+1,0}$ or $h_{j+1} < h_{j+1,0}$ with $h_{j+1} \approx h_{j+1,0}$.

Finally, note that, if we set each component of $[u_j]$ to $[-Tol/2, Tol/2]$, then

$$h_j^k \| w(f^{[k]}([\tilde{y}_j])) \| \leq Tol,$$

and we can have a local excess control per step. In this case, we always satisfy the tolerance requirement.

4.5. The Validation Algorithm

In Figure 1, we give an algorithmic description of the HOE method.

Although this algorithm may seem more expensive than an implementation of a FOE method (see for example [4]), we do not introduce much extra work in the former, compared to the latter, for the reasons discussed below.

The most expensive part of the HOE method is the computation of the interval Taylor coefficients $f^{[k]}([\tilde{y}_j])$ and $f^{[i]}([y_j])$, for $i = 1, \ldots, k-1$. However, if we use an interval Taylor series method with k terms to compute tight bounds for the solution in Algorithm II, we have to compute the term $h_j^k f^{[k]}([\tilde{y}_j])$ to bound the truncation error and generate $f^{[i]}([y_j])$ for $i = 1, \ldots, k-1$, when computing the Taylor coefficients for the solution to the associated variational equation [4, 6], which can also be viewed as the Jacobians of $f^{[i]}(y)$ evaluated at $[y_j]$. Thus, we can use $h_j^k f^{[k]}([\tilde{y}_j])$ and $f^{[i]}([y_j])$ for $i = 1, \ldots, k-1$ in the HOE method in Algorithm I at almost no additional cost.

Our HOE method is also cheaper and algorithmically simpler than the Taylor series enclosure method described in [6]. The main difference between the two methods is that, if we cannot verify existence and uniqueness with the input stepsize $h_{j,0}$, in the HOE method we compute $h_{j,1}$ such that (12) holds, while in [6], we keep reducing the input stepsize and repeating the validation procedure until we verify existence and uniqueness of the solution or reach a prescribed minimum value for the stepsize.

In addition, in [6], we use the Jacobian of $f^{[k]}(y)$ evaluated at $[y_j]$ to predict an a priori enclosure for the solution in $[t_j, t_j + h_{j,0}]$. However, in a Taylor series method with k terms in Algorithm II, we do not reuse this Jacobian.

5. Numerical Results

We compare the FOE and HOE methods on a constant coefficient problem, Van Der Pol's equation, and the two-body problem. For each of the problems, we report the number of steps and the global excess at the endpoint. We measure this excess by the max norm of the width of the enclosure, $\| w\left([y_m]\right) \|$.

We do not report numerical results comparing the HOE method with the Taylor series enclosure method from [6]. However, we have performed experiments which show that the HOE method generally allows fewer stepsizes for nearly the same, or slightly smaller, global excess than the Taylor series enclosure method described in [6].

The implementation of the FOE method is as in the AWA [4] package. In Algorithm II, we use an interval Taylor series method with Lohner's QR-factorization

INPUT:

n, k, $h_{j,0}$, $h_{\min}$, $[y_j]$, ϵ

$f^{[k]}([\tilde{y}_{j-1}])$ when $j \geq 1$.

COMPUTE:

$[p_j] := [y_j] + [0, h_{j,0}]\Big(f^{[1]}([y_j]) + \cdots + [0, h_{j,0}]\big(f^{[k-2]}([y_j]) + [0, h_{j,0}] f^{[k-1]}([y_j])\big)\cdots\Big)$

if $j = 0$ **then** % *first step*

$[u] := 2h_{0,0}^k \, [-|f^{[k]}([y_0])|, |f^{[k]}([y_0])|]$

$\beta := \big|f^{[k]}([p_j] + [u])\big|$

else

$\beta := \big|f^{[k]}([\tilde{y}_{j-1}])\big|$

end-if

for $i = 1, \ldots, n$ **do**

if $\beta_i = 0$ **then** $\beta_i := \epsilon$ **end-if**

end-for

$[u] := 2h_{j,0}^k \, [-\beta, \beta]$

$[\tilde{y}_j] := [p_j] + [u]$

$[z] := h_{j,0}^k f^{[k]}([\tilde{y}_j])$

$[v] := [0, 1][z]$

$h_j := h_{j,0}$

if $[v] \not\subseteq [u]$ **then**

find α such that $[0, \alpha^k][z] \subseteq [u]$

$h_j := \alpha h_j$

if $h_j < h_{\min}$ **then**

print "The stepsize is too small: cannot verify existence and uniqueness"

exit

end-if

OUTPUT:

h_j, $[\tilde{y}_j]$, $h_j^k f^{[k]}([\tilde{y}_j])$.

Figure 1. The HOE Method.

technique for reducing the wrapping effect [4]. We use the same (constant) order for the Taylor series in Algorithm II and in the HOE method and select a stepsize using (16–18). The numerical results are produced with the VNODE (Validated Numerical ODE) package [6].

5.1. Constant Coefficient Problem

We compare the FOE and HOE methods for the system

$$y' = By = \begin{pmatrix} 1 & -2 \\ 3 & -4 \end{pmatrix} y \tag{21}$$

[4] with

$$y(0) = (1, -1)^T, \quad \text{for} \quad t \in [0, 100]. \tag{22}$$

The eigenvalues of B are -1 and -2, and the solution components are

$$y_1(t) = 5e^{-t} - 4e^{-2t} \quad \text{and} \quad y_2(t) = 5e^{-t} - 6e^{-2t}.$$

We integrated (21–22) with orders $k = 10, 20, 30$, and 40 and $Tol = 10^{-5}$. We show in Table 1 the excess and the number of steps obtained with the FOE and HOE methods for these orders. Obviously, VNODE with the HOE method took significantly fewer steps than with the FOE method. Note also that VNODE with the FOE method computed much tighter bounds than we asked for because the stepsize was limited by the FOE method, as explained briefly below.

Table 1. FOE and HOE on (21–22) with orders $k = 10, 20, 30$, and 40 and $Tol = 10^{-5}$.

Order	Excess		Steps	
	FOE	HOE	FOE	HOE
10	6.1×10^{-12}	6.1×10^{-6}	825	206
20	6.2×10^{-25}	2.3×10^{-6}	797	96
30	4.0×10^{-40}	9.1×10^{-7}	737	64
40	4.5×10^{-55}	1.9×10^{-7}	725	48

With the FOE method, we try to validate existence and uniqueness of the solution with h_j and $[\tilde{y}_j]$ such that

$$[y_j] + [0, h_j]B\,[\tilde{y}_j] \subseteq [\tilde{y}_j]\,. \tag{23}$$

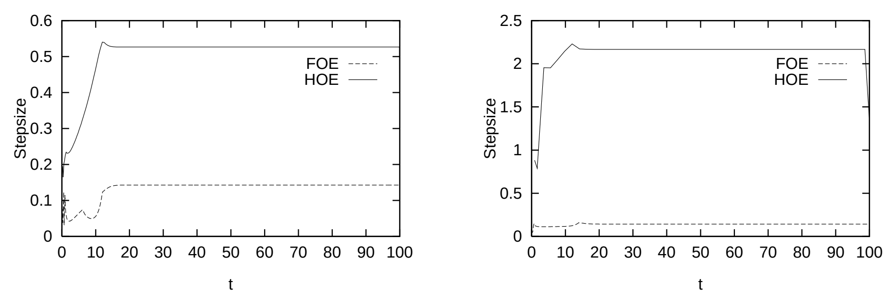
As the true solution approaches zero, the endpoints of $[y_j]$ in (23) approach zero too. Then, verifying (23) is almost like verifying

$$[0, h_j]B\,[\tilde{y}_j] \subseteq [\tilde{y}_j]\,. \tag{24}$$

Our implementation of the FOE method uses approximately $([-\epsilon,\epsilon],[-\epsilon,\epsilon])^T$, where $\epsilon > 0$ denotes the tolerance, as an initial guess for $[\tilde{y}_j]$. Then (24) is equivalent to

$$[0,h_j]B([-\epsilon,\epsilon],[-\epsilon,\epsilon])^T \subseteq ([-\epsilon,\epsilon],[-\epsilon,\epsilon])^T. \tag{25}$$

The largest h_j that satisfies (25) is $1/7$.



(a) $k = 10$

(b) $k = 40$

Figure 2. The FOE and HOE methods on (21–22).

Note that, in this example, the stepsize is limited to $1/7$, independently of ϵ. This implies that no matter what tolerance or order is used in Algorithm II, the largest possible stepsize, with this choice for $[\tilde{y}_j]$, is $1/7$. An important point to make here is that this stepsize restriction is not imposed by stability. That is, the eigenvalues of B do not play a role in restricting the stepsize.

As Stetter [12] points out, this stepsize restriction can be relaxed significantly for linear problems, but his technique does not apply directly to nonlinear problems, whereas our more general approach does. Since we are interested in comparing the FOE and HOE methods for a general Validated ODE solver, we do not pursue the special case of linear problems here.

In Figures 2(a) and 2(b), we plot the stepsizes used by VNODE with orders $k = 10$ and $k = 40$ versus t. In both cases, the stepsizes with the FOE method settled at $0.142578 \approx 1/7$. (The reduction in the stepsize on the last step with the HOE method seen in Figure 2(b) is a result of our program reducing the stepsize to hit the endpoint exactly.)

For this problem, as t increases, the norm of the width of the computed enclosures becomes smaller than the tolerance, and the predicted stepsize is determined by the stability of the interval Taylor series method in Algorithm II. (Some stability results can be found in [6].) For the orders used, the predicted stepsize is usually bigger than $1/7$, and the FOE reduces it, while the HOE method normally verifies existence and uniqueness with the input stepsize, $h_{j,0}$.

5.2. *Van der Pol's equation*

We integrated Van der Pol's equation, written as a first-order system,

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= 2(1-y_1^2)y_2 - y_1, \end{aligned} \tag{26}$$

with

$$y(0) = (2,0)^T, \quad \text{for} \quad t \in [0,20]. \tag{27}$$

We used orders $k = 10, 20, 30$, and 40 and tolerances $10^{-5}, 10^{-7}, \ldots, 10^{-13}$. We show in Tables 2–5 the excess and the number of steps obtained with the FOE and HOE methods for these orders and tolerances. In Figure 3, we plot the base-10 logarithm of the number of steps versus the base-10 logarithm of the tolerance.

Table 2. FOE and HOE on Van der Pol's equation; $k = 10$.

Tol	Excess		Steps	
	FOE	HOE	FOE	HOE
1.0×10^{-5}	1.9×10^{-5}	1.5×10^{-5}	260	178
1.0×10^{-7}	1.9×10^{-7}	1.6×10^{-7}	332	272
1.0×10^{-9}	2.1×10^{-9}	1.8×10^{-9}	417	420
1.0×10^{-11}	2.1×10^{-11}	1.9×10^{-11}	649	656
1.0×10^{-13}	8.0×10^{-13}	8.0×10^{-13}	1020	1028

Table 3. FOE and HOE on Van der Pol's equation; $k = 20$.

Tol	Excess		Steps	
	FOE	HOE	FOE	HOE
1.0×10^{-5}	1.1×10^{-5}	4.0×10^{-6}	213	93
1.0×10^{-7}	1.1×10^{-7}	4.1×10^{-8}	218	112
1.0×10^{-9}	1.2×10^{-9}	4.2×10^{-10}	225	135
1.0×10^{-11}	1.4×10^{-11}	4.5×10^{-12}	251	164
1.0×10^{-13}	3.4×10^{-13}	2.0×10^{-13}	286	201

From Table 2, we see that for $k = 10$ and $Tol = 10^{-9}, 10^{-11}$, and 10^{-13}, VNODE took approximately the same number of steps with the FOE and HOE methods. The reason is that for lower orders and stringent tolerances, the stepsize suggested by the stepsize controller is small enough that the FOE method usually verifies existence and uniqueness with the suggested stepsize. For these tolerances and order, VNODE took a few more steps with the HOE method than with the FOE method. This is probably because of the overestimation of the set $\mathcal{P}_j$ in (9) by the interval vector $[p_j]$ computed as in (14).

Table 4. FOE and HOE on Van der Pol's equation; $k = 30$.

Tol	Excess		Steps	
	FOE	HOE	FOE	HOE
1.0×10^{-5}	7.3×10^{-6}	1.3×10^{-6}	211	77
1.0×10^{-7}	8.6×10^{-8}	1.3×10^{-8}	212	86
1.0×10^{-9}	8.7×10^{-10}	1.3×10^{-10}	213	97
1.0×10^{-11}	8.8×10^{-12}	1.4×10^{-12}	218	109
1.0×10^{-13}	2.8×10^{-13}	1.3×10^{-13}	221	123

Table 5. FOE and HOE on Van der Pol's equation; $k = 40$.

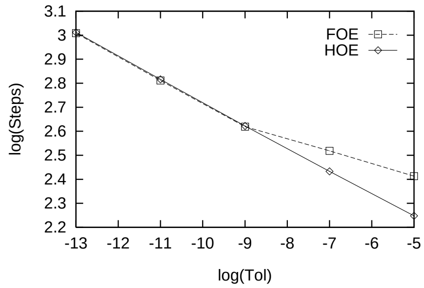
Tol	Excess		Steps	
	FOE	HOE	FOE	HOE
1.0×10^{-5}	5.7×10^{-6}	4.3×10^{-7}	218	70
1.0×10^{-7}	6.6×10^{-8}	4.2×10^{-9}	214	76
1.0×10^{-9}	6.7×10^{-10}	4.2×10^{-11}	212	83
1.0×10^{-11}	7.9×10^{-12}	5.1×10^{-13}	215	90
1.0×10^{-13}	2.6×10^{-13}	1.1×10^{-13}	215	98

For $k = 10$ and $Tol = 10^{-5}$ and 10^{-7}, and for the remaining orders and all tolerances, VNODE with the HOE method took fewer steps than with the FOE method. As the order increases, the stepsize that can be used in Algorithm II increases, but the stepsize that can be used by the FOE method in Algorithm I does not. Thus, the stepsize is limited by the FOE method in Algorithm I and the number of steps remain approximately the same for all orders $k \geq 20$ and all tolerances considered. Although the HOE method often reduces the input stepsize, the reduction factor is normally close to one. For example, for $k = 40$ and $Tol = 10^{-5}$, the average reduction, taken over $h_{j,0}/h_j > 1$, of the input stepsize for the FOE method is 8.07, while the average reduction for the HOE method is 1.16.

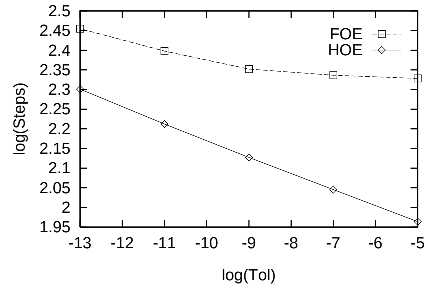
5.3. *Two Body Problem*

We integrated the two-body problem, written as a first-order system,

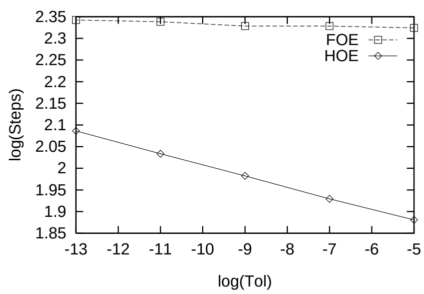
$$
\begin{aligned}
y_1' &= y_3 \\
y_2' &= y_4 \\
y_3' &= -\frac{y_1}{(y_1^2 + y_2^2)^{3/2}} \\
y_4' &= -\frac{y_2}{(y_1^2 + y_2^2)^{3/2}},
\end{aligned}
\tag{28}
$$



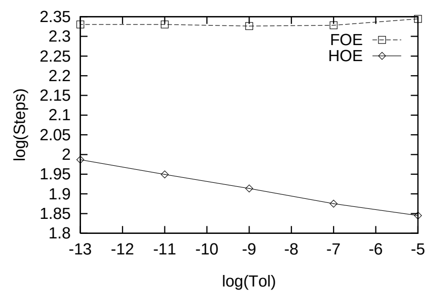
(a) $k = 10$



(b) $k = 20$



(c) $k = 30$



(d) $k = 40$

Figure 3. The FOE and HOE methods on Van der Pol's equation with $Tol = 10^{-5}, 10^{-7}, \ldots, 10^{-13}$.

with

$$y(0) \; = \; (0.5, 0, 0, \sqrt{3})^T, \quad \text{for} \qquad t \in [0, 20] \tag{29}$$

with order $k = 20$ and tolerances $10^{-7}, 10^{-9}, \ldots, 10^{-13}$. Here, the eccentricity of the orbit is 0.5, and the interval of integration corresponds to slightly more than three periods.

For this problem, VNODE with the FOE and HOE methods took almost the same number of steps, Table 6. The reason is that the stepsizes computed by the stepsize controller are small enough that the FOE and HOE methods verify existence and uniqueness with the suggested stepsize, or a stepsize that is not much smaller than that.

Acknowledgments

The authors thank Dr. George Corliss and Dr. Knut Petras for reading an early draft of this paper and making several helpful comments.

Table 6. Two-body problem; $k = 20$.

Tol	Excess		Steps	
	FOE	HOE	FOE	HOE
1.0×10^{-7}	4.5×10^{-4}	4.2×10^{-5}	169	167
1.0×10^{-9}	2.7×10^{-6}	4.2×10^{-7}	195	194
1.0×10^{-11}	2.1×10^{-8}	4.4×10^{-9}	228	229
1.0×10^{-13}	3.0×10^{-10}	1.7×10^{-10}	268	271

Appendix

A.1. A Possibly Better Method for Bounding $\mathcal{P}_j$

By computing tighter bounds for the set (9), we should obtain tighter bounds on $h_j^k f^{[k]}([\tilde{y}_j])$ and, consequently, on the truncation error. Therefore, if we use a stepsize selection scheme which controls $h_j^k \| w\left(f^{[k]}([\tilde{y}_j])\right)\|$, such as the one outlined in section 4.4, we should be able to use larger stepsizes and thereby solve problems more efficiently, provided, of course, the added cost of computing tighter bounds for the set (9) is not too high.

In this section, we sketch an alternative method that should give tighter bounds for the set $\mathcal{P}_j$ in (9) than the evaluation (14). However, we have not implemented the approach described here, since we could not find a stand-alone program that computes tight bounds for the range of a polynomial with real coefficients over an interval. However, after we finished the first draft of this paper, we discovered that such an algorithm has been proposed by Makino [5]. We plan to investigate the appropriateness of her method in this context in the near future.

To outline our approach, let y_j and $\hat{y}_j \in [y_j]$. We apply the mean-value theorem to the functions $f^{[i]}$ in (8) to obtain

$$\begin{aligned} p_{k-1}(t;t_j,y_j) &= \sum_{i=0}^{k-1}(t-t_j)^i f^{[i]}(y_j) \\ &\in \sum_{i=0}^{k-1}(t-t_j)^i f^{[i]}(\hat{y}_j) + \left(\sum_{i=0}^{k-1}[0,h_{j,0}^i]\frac{\partial f^{[i]}}{\partial y}([y_j])\right)([y_j]-\hat{y}_j). \end{aligned} \tag{A.1}$$

Let $[\hat{p}_j]$ be such that

$$\sum_{i=0}^{k-1}(t-t_j)^i f^{[i]}(\hat{y}_j) \in [\hat{p}_j] \quad \text{for} \quad [t_j, t_j + h_{j,0}] \tag{A.2}$$

and denote

$$[q_j] = \left(\sum_{i=0}^{k-1}[0,h_{j,0}^i]\frac{\partial f^{[i]}}{\partial y}([y_j])\right)([y_j]-\hat{y}_j). \tag{A.3}$$

Using (A.2–A.3), (A.1) can be written as

$$p_{k-1}(t; t_j, y_j) \in [\hat{p}_j] + [q_j] \equiv [p_j]. \tag{A.4}$$

Assuming that $w([y_j])$ is small, we can consider this $[p_j]$ as a sum of an enclosure $[\hat{p}_j]$ of a Taylor polynomial with real coefficients plus a small perturbation $[q_j]$. If $[\hat{p}_j]$ is a tight enclosure for the range of $\sum_{i=0}^{k-1}(t-t_j)^i f^{[i]}(\hat{y}_j)$ for $t \in [t_j, t_j + h_{j,0}]$, then $[p_j]$ from (A.4) should often give tighter bounds for $\mathcal{P}_j$ than the $[p_j]$ from (14).

As noted above, we plan to implement this scheme soon. Here, we only make the following remark. In an interval Taylor series method (in Algorithm II) with a truncation error of order k, we need the terms $f^{[i]}(\hat{y}_j)$ and $\partial f^{[i]}([y_j])/\partial y$, for $i = 1, \ldots, k-1$, [4, 6, 11]. Thus, we can use them in (A.1) at almost no additional cost.

References

1. M. Berz and K. Makino. Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models. *Reliable Computing*, 4:361–369, 1998.
2. G. F. Corliss and R. Rihm. Validating an a priori enclosure using high-order Taylor series. In G. Alefeld and A. Frommer, editors, *Scientific Computing, Computer Arithmetic, and Validated Numerics*, pages 228–238. Akademie Verlag, Berlin, 1996.
3. P. Eijgenraam. *The Solution of Initial Value Problems Using Interval Arithmetic.* Mathematical Centre Tracts No. 144. Stichting Mathematisch Centrum, Amsterdam, 1981.
4. R. J. Lohner. *Einschließung der Lösung gewöhnlicher Anfangs– und Randwertaufgaben und Anwendungen.* PhD thesis, Universität Karlsruhe, 1988. AWA is available at ftp://iamk4515.mathematik.uni-karlsruhe.de/pub/awa/.
5. K. Makino. *Rigorous Analysis of Nonlinear Motion in Particle Accelerators.* PhD thesis, Michigan State University, 1998.
6. N. S. Nedialkov. *Computing Rigorous Bounds on the Solution of an Initial Value Problem for an Ordinary Differential Equation.* PhD thesis, Department of Computer Science, University of Toronto, Toronto, Canada, M5S 3G4, February 1999. Available at www.cs.toronto.edu/NA/reports.html.
7. N. S. Nedialkov and K. R. Jackson. An interval Hermite-Obreschkoff method for computing rigorous bounds on the solution of an initial value problem for an ordinary differential equation. In T. Csendes, editor, *Developments in Reliable Computing*, pages 289–310. Kluwer, Dordrecht, Netherlands, 1999. Also Reliable Computing 5/3 (1999), pages 289–310.
8. N. S. Nedialkov and K. R. Jackson. A new perspective on the wrapping effect in interval methods for IVPs for ODEs, 1999. In preparation.
9. N. S. Nedialkov, K. R. Jackson, and G. F. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Appl. Math. Comp.*, 105(1):21–68, August 1999.
10. R. Rihm. Interval methods for initial value problems in ODEs. In J. Herzberger, editor, *Topics in Validated Computations: Proceedings of the IMACS-GAMM International Workshop on Validated Computations, University of Oldenburg*, Elsevier Studies in Computational Mathematics, pages 173–207. Elsevier, Amsterdam, New York, 1994.
11. O. Stauning. *Automatic Validation of Numerical Solutions.* PhD thesis, Technical University of Denmark, DK-2800, Lyngby, Denmark, October 1997. This thesis and ADIODES are available at www.imm.dtu.dk/fadbad.html.
12. H. J. Stetter. Validated solution of initial value problems for ODEs. In C. Ullrich, editor, *Computer Arithmetic and Self-Validating Numerical Methods*, pages 171–187. Academic Press, New York, 1990.