, , 1–22 ()

An Interval Hermite-Obreschkoff Method for Computing Rigorous Bounds on the Solution of an Initial Value Problem for an Ordinary Differential Equation*

N. S. NEDIALKOV AND K. R. JACKSON {ned,krj}@cs.toronto.edu
Department of Computer Science, University of Toronto, Toronto, Ontario, Canada, M5S 3G4.

Abstract. To date, the only effective approach for computing guaranteed bounds on the solution of an initial value problem (IVP) for an ordinary differential equation (ODE) has been interval methods based on Taylor series. This paper derives a new approach, an interval Hermite-Obreschkoff (IHO) method, for computing such enclosures. Compared to interval Taylor series (ITS) methods, for the same stepsize and order, our IHO scheme has a smaller truncation error, better stability, and requires fewer Taylor coefficients and high-order Jacobians.

The stability properties of the ITS and IHO methods are investigated. We show as an important by-product of this analysis that the stability of an interval method is determined not only by the stability function of the underlying formula, as in a standard method for an IVP for an ODE, but also by the associated formula for the truncation error.

Keywords: initial value problems, ordinary differential equations, interval arithmetic, Taylor series methods, Hermite-Obreschkoff methods, stability.

1. Introduction

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

$$\begin{aligned} y'(t) &= f(y) &\quad (1)\\ y(t_0) &\in [y_0], &\quad (2) \end{aligned}$$

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, $f : \mathcal{D} \rightarrow \mathbb{R}^n$, 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 equations (ODEs) can be converted into an autonomous system. Moreover, the methods discussed here can be extended easily to nonautonomous systems.

* 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.

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

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

Our goal is to compute 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,

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

Usually, validated methods for IVPs for ODEs are one-step methods, where the $(j+1)$st step consists of two phases, which we refer to as Algorithm I and Algorithm II [25]:

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

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

Significant developments in the area of validated solutions of IVPs for ODEs are the interval methods of Moore [22], [23], [24], Krückeberg [18], Eijgenraam [9], and Lohner [1], [20], [21]. All these methods are based on Taylor series. One reason for the popularity of the Taylor series approach is the simple form of the error term. In addition, the Taylor series coefficients can be readily generated by automatic differentiation [24, pp. 107–130], 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.

In this paper, we develop an interval Hermite-Obreschkoff (IHO) method for computing tight enclosures of the solution. Validated methods based on the Hermite-Obreschkoff formula [15], [27], [28] have not been derived or considered before. Although Taylor series methods can be viewed as a special case of the more general Hermite-Obreschkoff methods, the method we propose is conceptually different from the interval Taylor series (ITS) methods. We show that compared with ITS methods, for the same stepsize and order, our IHO scheme has a smaller truncation error, better stability, and may be less expensive for problems for which the computation of $f(y)$ involves many terms.

This paper also shows that the stability of the ITS and IHO methods depends not only on the stability function of the underlying formula, as in standard numerical methods for IVPs for ODEs, but also on the associated formula for the truncation error. In standard numerical methods, Hermite-Obreschkoff methods are known to be suitable for stiff systems [11], [12], [34], [35], but in the interval case, they still have a restriction on the stepsize. To develop an interval method for stiff problems, we need not only a formula with a good stability function, but also a better associated formula for the truncation error.

An outline of this paper follows. Section 2 explains briefly how ITS methods for computing tight enclosures of the solution work and reviews Lohner's method. Section 3 derives the IHO method. First, in §3.1, we show how the point Hermite-Obreschkoff method can be obtained. Then in §3.2, we derive the interval version and show how to represent the enclosure in a manner that reduces the wrapping effect in propagating the error. Section 4 compares the IHO and ITS methods in the constant coefficient case and gives estimates for the amount of work per step required by these methods in the general case. This section also shows that the IHO method is more stable than an ITS method. Section 5 contains numerical results comparing the two methods.

Notation

An interval or interval vector "a" is denoted by $[a]$; an interval matrix "A" is denoted by $[A]$. We use small letters for intervals and interval vectors and capital letters for interval matrices. For an interval $[a] = [\underline{a}, \bar{a}]$, we define

width: $w([a]) = \bar{a} - \underline{a}$;

midpoint: $m([a]) = (\bar{a} + \underline{a})/2$; and

magnitude: $|[a]| = \max\{|\bar{a}|, |\underline{a}|\}$.

We define width, midpoint, and magnitude componentwise for interval vectors and matrices.

We introduce the sequence of functions

$$f^{[0]}(y) = y, \tag{3}$$

$$f^{[i]}(y) = \frac{1}{i}\left(\frac{\partial f^{[i-1]}}{\partial y} f\right)(y), \qquad \text{for } i \geq 1. \tag{4}$$

For the autonomous IVP $y'(t) = f(y)$, $\quad y(t_j) = y_j$, $f^{[i]}(y_j) = y^{(i)}(t_j)/i!$. That is, $f^{[i]}(y_j)$ denotes the ith Taylor coefficient of $y(t; t_j, y_j)$

2. Taylor Series Methods

Consider the Taylor series expansion

$$y_{j+1} = y_j + \sum_{i=1}^{k-1} h_j^i f^{[i]}(y_j) + h_j^k f^{[k]}(y; t_j, t_{j+1}), \tag{5}$$

where $y_j \in [y_j]$ and $f^{[k]}(y; t_j, t_{j+1})$ denotes $f^{[k]}$ with its lth component evaluated at $y(\xi_{jl})$, for some $\xi_{jl} \in [t_j, t_{j+1}]$. If (5) is evaluated in interval arithmetic with y_j

replaced by $[y_j]$ and $f^{[k]}(y;t_j,t_{j+1})$ replaced by $f^{[k]}([\tilde{y}_j])$, we obtain

$$[y_{j+1}] \;=\; [y_j] + \sum_{i=1}^{k-1} h_j^i f^{[i]}([y_j]) + h_j^k f^{[k]}([\tilde{y}_j]). \tag{6}$$

With (6), we can compute enclosures of the solution, but the width of $[y_j]$ always increases with j, even if the true solution contracts.

By applying the mean-value theorem to $f^{[i]}$ at some $\hat{y}_j \in [y_j]$, we have

$$f^{[i]}(y_j) = f^{[i]}(\hat{y}_j) + J\left(f^{[i]}; y_j, \hat{y}_j\right)(y_j - \hat{y}_j), \tag{7}$$

where $J\left(f^{[i]}; y_j, \hat{y}_j\right)$ is the Jacobian of $f^{[i]}$ with its lth 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 (5) and (7),

$$\begin{aligned} y_{j+1} = \hat{y}_j \;&+\; \sum_{i=1}^{k-1} h_j^i f^{[i]}(\hat{y}_j) + h_j^k f^{[k]}(y;t_j,t_{j+1}) \\ &+\; \left\{ I + \sum_{i=1}^{k-1} h_j^i J\left(f^{[i]}; y_j, \hat{y}_j\right) \right\} (y_j - \hat{y}_j). \end{aligned} \tag{8}$$

This formula is the basis of the interval Taylor series methods of Moore [22], [23], [24], Eijgenraam [9], Lohner [1], [20], [21], and Rihm [30] (see also [25]). If we evaluate (8) in interval arithmetic, we can often obtain enclosures with smaller widths than in (6). A serious problem that may occur in this evaluation is the wrapping effect [24]. Here, we briefly describe Lohner's method for reducing it.

2.1. Lohner's Method

Let

$$\begin{aligned} z_{j+1} &= h_j^k f^{[k]}(y;t_j,t_{j+1}) \in h_j^k f^{[k]}([\tilde{y}_j]) \equiv [z_{j+1}]\,, && (9) \\ s_{j+1} &= m\left([z_{j+1}]\right), && (10) \\ \hat{y}_{j+1} &= \hat{y}_j + \sum_{i=1}^{k-1} h_j^i f^{[i]}(\hat{y}_j) + s_{j+1}, \quad \text{and} && (11) \\ S_j &= I + \sum_{i=1}^{k-1} h_j^i J\left(f^{[i]}; y_j, \hat{y}_j\right) \in I + \sum_{i=1}^{k-1} h_j^i J\left(f^{[i]}; [y_j]\right) \equiv [S_j]\,. && (12) \end{aligned}$$

Also let

$$A_0 = I,\;\; \hat{y}_0 = m\left([y_0]\right),\;\; \text{and}\;\; r_0 = y_0 - \hat{y}_0 \in [r_0] = [y_0] - \hat{y}_0, \tag{13}$$

where I is the identity matrix. From (8) and (9–13), we compute

$$[y_{j+1}] = \hat{y}_j + \sum_{i=1}^{k-1} h_j^i f^{[i]}(\hat{y}_j) + \left([S_j]\,A_j\right)[r_j] + [z_{j+1}] \tag{14}$$

and propagate for the next step the error vector

$$[r_{j+1}] = \left(A_{j+1}^{-1}([S_j]\,A_j)\right)[r_j] + A_{j+1}^{-1}\left([z_{j+1}] - s_{j+1}\right). \tag{15}$$

Here $A_j \in \mathbb{R}^{n\times n}$ $(j \geq 0)$ is a nonsingular matrix. Usually, a good choice for A_{j+1} is the Q-factor from the QR-factorization of $\widehat{A}_{j+1} = m\left([S_j]A_j\right)$. A detailed explanation of the reasons for this and other choices for A_{j+1} can be found in [21] and [25].

3. Derivation of the Interval Hermite-Obreschkoff Method

3.1. The Point Method

Let

$$\begin{aligned} P_{p,q}(s) &= \frac{s^q(s-1)^p}{(p+q)!}, && (16)\\ c_i^{q,p} &= \frac{q!}{(p+q)!}\,\frac{(q+p-i)!}{(q-i)!}, \quad \text{and} && (17)\\ g_i(s) &= \frac{g^{(i)}(s)}{i!}, && (18) \end{aligned}$$

where $p \geq 0$, $q \geq 0$, $0 \leq i \leq q$, and $g(t)$ is $(p+q+1)$ times differentiable. If we integrate $\int_0^1 P_{p,q}(s)g^{(p+q+1)}(s)\,ds$ repeatedly by parts, we find

$$(-1)^{(p+q)}\int_0^1 P_{p,q}(s)g^{(p+q+1)}(s)\,ds = \sum_{i=0}^{q}(-1)^i c_i^{q,p} g_i(1) - \sum_{i=0}^{p} c_i^{p,q} g_i(0). \tag{19}$$

This derivation is sometimes attributed to Darboux [7] and Hermite [15].

If $y(t)$ is the solution to

$$y' = f(y), \qquad y(t_j) = y_j,$$

and we set $g(s) = y(t_j + sh_j)$, then

$$\begin{aligned} g^{(p+q+1)}(s) &= h_j^{p+q+1} y^{(p+q+1)}(t_j + sh_j), && (20)\\ g_i(0) &= \frac{g^{(i)}(0)}{i!} = h_j^i \frac{y^{(i)}(t_j)}{i!} = h_j^i f^{[i]}(y_j), \quad \text{and} && (21)\\ g_i(1) &= \frac{g^{(i)}(1)}{i!} = h_j^i \frac{y^{(i)}(t_j+h_j)}{i!} = h_j^i f^{[i]}(y_{j+1}), && (22) \end{aligned}$$

where $y_{j+1} = y(t_j + h_j)$, and the functions $f^{[i]}$ are defined in (3–4). From (19) and (20–22),

$$\begin{aligned} \sum_{i=0}^{q}(-1)^i c_i^{q,p} h_j^i f^{[i]}(y_{j+1}) &= \sum_{i=0}^{p} c_i^{p,q} h_j^i f^{[i]}(y_j)\\ &\quad + (-1)^q \frac{q!p!}{(p+q)!} h_j^{p+q+1} \frac{y^{(p+q+1)}(t;t_j,t_{j+1})}{(p+q+1)!}, \end{aligned} \tag{23}$$

where the lth component of $y^{(p+q+1)}(t;t_j,t_{j+1})$ is evaluated at some $\xi_{jl} \in [t_j,t_{j+1}]$.

For a given y_j, if we solve the nonlinear (in general) system of equations

$$\sum_{i=0}^{q}(-1)^i c_i^{q,p} h_j^i f^{[i]}(y_{j+1}) = \sum_{i=0}^{p} c_i^{p,q} h_j^i f^{[i]}(y_j) \tag{24}$$

for y_{j+1}, we obtain an approximation of local order $O(h_j^{p+q+1})$ to the solution $y(t;t_j,y_j)$ at t_{j+1}. The system (24) defines the point (q,p) Hermite-Obreschkoff method [11], [12], [14, p. 277], [34], [35].

If $p > 0$ and $q = 0$, then we obtain from (23) an explicit Taylor series method; if $p = 0$ and $q > 0$, then (23) becomes an implicit Taylor series formula. Hence, we can consider the Hermite-Obreschkoff methods that we obtain from (23) as a generalization of Taylor series methods.

3.2. The Interval Method

Suppose that we have computed $[y_j]$, $\hat{y}_j$, A_j, and $[r_j]$ at t_j such that

$$\begin{aligned} y(t_j;t_0,[y_0]) &\subseteq [y_j] \quad \text{and} \\ y(t_j;t_0,[y_0]) &\subseteq \{\hat{y}_j + A_j r_j \mid r_j \in [r_j]\} \equiv \mathcal{U}_j, \end{aligned}$$

where $\hat{y}_j = m([y_j])$, $A_j \in \mathbb{R}^{n\times n}$ is nonsingular, and $[r_j]$ is an interval vector. The interval vectors $[y_j]$ and $\hat{y}_j + A_j[r_j]$ are not necessarily the same. We use the representation $\mathcal{U}_j$ to reduce the wrapping effect in propagating the error and the representation $[y_j]$ in generating the Jacobians of the Taylor coefficients $f^{[i]}$. Suppose also that we have verified existence and uniqueness of the solution on $[t_j,t_{j+1}]$ and have computed an a priori enclosure $[\tilde{y}_j]$ on $[t_j,t_{j+1}]$.

The method we propose consists of two phases, which can be considered as a predictor and a corrector. The predictor computes an enclosure $[y_{j+1}^{(0)}]$ of the solution at t_{j+1}, and using this enclosure, the corrector computes a tighter enclosure $[y_{j+1}] \subseteq [y_{j+1}^{(0)}]$ at t_{j+1}. In the predictor, we compute the enclosure $[y_{j+1}^{(0)}]$ with an ITS method of order $(q+1)$; for example, we can use Lohner's method from §2. Here, we show how to construct a corrector based on (23).

Our goal is to compute a tighter enclosure $[y_{j+1}]$ of the solution than $[y_{j+1}^{(0)}]$ and a representation of the enclosure set in the form

$$\mathcal{U}_{j+1} = \{\hat{y}_{j+1} + A_{j+1} r_{j+1} \mid r_{j+1} \in [r_{j+1}]\}.$$

That is, we have to compute $[y_{j+1}]$, $\hat{y}_{j+1}$, A_{j+1}, and $[r_{j+1}]$ for the next step.

Let

$$y_j = y(t_j;t_0,y_0) \quad \text{and} \quad y_{j+1} = y(t_{j+1};t_0,y_0)$$

for some $y_0 \in [y_0]$, and $\hat{y}_{j+1}^{(0)} = m([y_{j+1}^{(0)}])$. Since

$$y_{j+1},\ \hat{y}_{j+1}^{(0)} \in [y_{j+1}^{(0)}] \quad \text{and} \quad y_j,\ \hat{y}_j \in [y_j],$$

we can apply the mean-value theorem to the two sums in (23) to obtain

$$\left(\sum_{i=0}^{q}(-1)^i c_i^{q,p} h_j^i J\left(f^{[i]}; y_{j+1}, \hat{y}_{j+1}^{(0)}\right)\right)(y_{j+1} - \hat{y}_{j+1}^{(0)}) =$$

$$\sum_{i=0}^{p} c_i^{p,q} h_j^i f^{[i]}(\hat{y}_j) - \sum_{i=0}^{q}(-1)^i c_i^{q,p} h_j^i f^{[i]}(\hat{y}_{j+1}^{(0)})$$

$$+\left(\sum_{i=0}^{p} c_i^{p,q} h_j^i J\left(f^{[i]}; y_j, \hat{y}_j\right)\right)(y_j - \hat{y}_j) + \epsilon_{j+1}, \qquad (25)$$

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

$$\epsilon_{j+1} = (-1)^q \frac{q!p!}{(p+q)!} h_j^{p+q+1} \frac{y^{(p+q+1)}(t; t_j, t_{j+1})}{(p+q+1)!}. \qquad (26)$$

Using (25), we show how to compute a better enclosure than $[y_{j+1}^{(0)}]$ at t_{j+1}. To this end, we denote by $J(f^{[i]}; [y_{j+1}^{(0)}])$ the Jacobian of $f^{[i]}$ evaluated over $[y_{j+1}^{(0)}]$ and by $J\left(f^{[i]}; [y_j]\right)$ the Jacobian of $f^{[i]}$ evaluated over $[y_j]$. Let

$$\begin{aligned}
S_{j+1,-} &= \sum_{i=0}^{q}(-1)^i c_i^{q,p} h_j^i J\left(f^{[i]}; y_{j+1}, \hat{y}_{j+1}^{(0)}\right) \\
&\in \sum_{i=0}^{q}(-1)^i c_i^{q,p} h_j^i J\left(f^{[i]}; [y_{j+1}^{(0)}]\right) \equiv [S_{j+1,-}], && (27) \\
\widehat{S}_{j+1,-} &= m\left([S_{j+1,-}]\right), && (28) \\
S_{j,+} &= \sum_{i=0}^{p} c_i^{p,q} h_j^i J\left(f^{[i]}; y_j, \hat{y}_j\right) \\
&\in \sum_{i=0}^{p} c_i^{p,q} h_j^i J\left(f^{[i]}; [y_j]\right) \equiv [S_{j,+}], && (29) \\
[B_j] &= \left(\widehat{S}_{j+1,-}^{-1}[S_{j,+}]\right) A_j, && (30) \\
[C_j] &= I - \widehat{S}_{j+1,-}^{-1}[S_{j+1,-}], && (31) \\
[v_j] &= [y_{j+1}^{(0)}] - \hat{y}_{j+1}^{(0)}, && (32) \\
\epsilon_{j+1} &\in (-1)^q \frac{q!p!}{(p+q)!} h_j^{p+q+1} f^{[p+q+1]}([\tilde{y}_j]) \equiv [\epsilon_{j+1}], && (33) \\
g_{j+1} &= \sum_{i=0}^{p} c_i^{p,q} h_j^i f^{[i]}(\hat{y}_j) - \sum_{i=0}^{q}(-1)^i c_i^{q,p} h_j^i f^{[i]}(\hat{y}_{j+1}^{(0)}), \quad \text{and} && (34) \\
\delta_{j+1} &= g_{j+1} + \epsilon_{j+1} \in g_{j+1} + [\epsilon_{j+1}] \equiv [\delta_{j+1}]. && (35)
\end{aligned}$$

We can derive from (25–35) that

$$
\begin{aligned}
y_{j+1} - \hat{y}_{j+1}^{(0)} \; &\in \; \left((\widehat{S}_{j+1,-}^{-1}[S_{j,+}])A_j \right)[r_j] + \widehat{S}_{j+1,-}^{-1}[\delta_{j+1}] \\
&\qquad\qquad + (I - \widehat{S}_{j+1,-}^{-1}[S_{j+1,-}])([y_{j+1}^{(0)}] - \hat{y}_{j+1}^{(0)}) \\
&= \; [B_j][r_j] + [C_j][v_j] + \widehat{S}_{j+1,-}^{-1}[\delta_{j+1}].
\end{aligned}
$$

Note that for small h_j, we can compute the inverse of $\widehat{S}_{j+1,-}$. Since

$$
y_{j+1} \; = y(t_{j+1}; t_0, y_0) \in \hat{y}_{j+1}^{(0)} + [B_j][r_j] + [C_j][v_j] + \widehat{S}_{j+1,-}^{-1}[\delta_{j+1}],
$$

for an arbitrary $y_0 \in [y_0]$, we compute an interval vector that is a tight enclosure of the solution at t_{j+1} by

$$
[y_{j+1}] \; = \left(\hat{y}_{j+1}^{(0)} + [B_j][r_j] + [C_j][v_j] + \widehat{S}_{j+1,-}^{-1}[\delta_{j+1}] \right) \cap [y_{j+1}^{(0)}], \tag{36}
$$

where "$\cap$" denotes intersection of interval vectors. For the next step, we propagate $\hat{y}_{j+1} = m([y_{j+1}])$, A_{j+1}, which is the Q-factor from the QR-factorization of $m([B_j])$, and

$$
\begin{aligned}
[r_{j+1}] = (A_{j+1}^{-1}[B_j])[r_j] + (A_{j+1}^{-1}[C_j])[v_j] \\
+ (A_{j+1}^{-1}\widehat{S}_{j+1,-}^{-1})[\delta_{j+1}] + A_{j+1}^{-1}(\hat{y}_{j+1}^{(0)} - \hat{y}_{j+1}).
\end{aligned} \tag{37}
$$

Remarks

1. Once we obtain $[y_{j+1}]$, we can set $[y_{j+1}^{(0)}] = [y_{j+1}]$ and compute another enclosure, hopefully better than $[y_{j+1}]$, by repeating the same procedure. Thus, we can improve this enclosure iteratively. The experiments that we have performed show, however, that this iteration does not improve the results significantly, but increases the cost.

2. We could use the a priori enclosure $[\tilde{y}_j]$ computed in Algorithm I, instead of computing $[y_{j+1}^{(0)}]$. We briefly explain the reasons for computing $[y_{j+1}^{(0)}]$.

   (A) The a priori enclosure $[\tilde{y}_j]$ may be too wide and the corrector phase may not produce a tight enough enclosure in one iteration. As a result, the corrector, which is the expensive part, may need more than one iteration to obtain a tight enough enclosure.

   (B) Predicting a reasonably tight enclosure $[y_{j+1}^{(0)}]$ is not expensive: we need to generate the coefficients $f^{[i]}(\hat{y}_j)$ and $J\left(f^{[i]}; [y_j]\right)$ for $i = 1, \ldots, q$. We need them in the corrector, but for $i = 1, \ldots, p$. Usually, a good choice for q is $q \in \{p, p+1, p+2\}$ (see §4.1). Therefore, we do not create extra work when generating these terms in the predictor.

3. We could use the inverse of the interval matrix $[S_{j+1,-}]$ instead of $\widehat{S}_{j+1,-}^{-1}$. However, it is easier to compute the enclosure of the inverse of a point matrix than of an interval matrix. In fact, computing an enclosure of an interval matrix is NP hard in general [31].

4. If we intersect the computed enclosure as in (36), it is important to choose $\hat{y}_{j+1} \in [y_{j+1}]$. If we set $\hat{y}_{j+1} = \hat{y}_{j+1}^{(0)}$, it might happen that $\hat{y}_{j+1} = \hat{y}_{j+1}^{(0)} \notin [y_{j+1}]$, because $\hat{y}_{j+1}^{(0)}$ is the midpoint of $[y_{j+1}^{(0)}]$, which is generally a wider enclosure than $[y_{j+1}]$.

4. Interval Hermite-Obreschkoff versus Interval Taylor Series Method

For the remainder of the paper, we compare an ITS method with order k truncation error and our IHO scheme with the same order of the truncation error; that is, with p and q such that $k = p + q + 1$. In §4.1 and §4.2, we consider these methods with constant stepsize h.

4.1. The One-Dimensional Constant Coefficient Case. Instability Results

Consider the problem

$$y' = \lambda y, \qquad y(0) \in [y_0], \tag{38}$$

where $\lambda \in \mathbb{R}$ and $\lambda < 0$. Since we have not defined complex interval arithmetic, we do not consider problems with λ complex.

Definition 1. We say that an interval method with a constant stepsize is *asymptotically unstable*, if, when applied to the problem (38),

$$w([y_j]) \to \infty, \qquad \text{as } j \to \infty.$$

In this subsection, we look at the stability of the ITS and IHO methods. Note that the wrapping effect does not occur in one-dimensional problems.

4.1.1. The Interval Taylor Series Method Suppose that at $t_j > 0$, we have computed a tight enclosure $[y_j^{ITS}]$ of the solution at t_j with an ITS method, and $[\tilde{y}_j^{ITS}]$ is an a priori enclosure of the solution on $[t_j, t_{j+1}]$, for all $y_j \in [y_j^{ITS}]$, where $[y_0^{ITS}] = [y_0]$. Let

$$T_r(z) = \sum_{i=0}^{r} \frac{z^i}{i!}. \tag{39}$$

Using (39), an interval Taylor series method for computing tight enclosures of the solution to (38) can be written as

$$[y_{j+1}^{ITS}] = T_{k-1}(\lambda h)[y_j^{ITS}] + \frac{(\lambda h)^k}{k!}[\tilde{y}_j^{ITS}]; \tag{40}$$

cf. (8). Since $w([\tilde{y}_j^{ITS}]) \geq w([y_j^{ITS}])$, we derive from (39–40)

$$w([y_{j+1}^{ITS}]) \geq \left(|T_{k-1}(\lambda h)| + \frac{|\lambda h|^k}{k!} \right) w([y_j^{ITS}]).$$

Therefore, the ITS method given by (40) is asymptotically unstable for stepsizes h such that

$$|T_{k-1}(\lambda h)| + \frac{|\lambda h|^k}{k!} > 1. \tag{41}$$

This result implies that we have restrictions on the stepsize not only from the function $T_{k-1}(\lambda h)$, as in point methods for IVPs for ODEs, but also from the factor $|\lambda h|^k/k!$ in the remainder term.

4.1.2. The Interval Hermite-Obreschkoff Method Let $y_j \in [y_j^{IHO}]$, where we assume that $[y_j^{IHO}]$ is computed with an IHO method and $[y_0^{IHO}] = [y_0]$. From (23), the true solution y_{j+1} corresponding to the point y_j satisfies

$$\begin{aligned}\left(\sum_{i=0}^{q}(-1)^i c_i^{q,p}\frac{(\lambda h)^i}{i!}\right) y_{j+1} = \left(\sum_{i=0}^{p} c_i^{p,q}\frac{(\lambda h)^i}{i!}\right) y_j \\ +(-1)^q\frac{q!p!}{(p+q)!}\frac{(\lambda h)^{p+q+1}}{(p+q+1)!}y(\xi),\end{aligned} \tag{42}$$

where $\xi \in [t_j, t_{j+1}]$. Let

$$R_{p,q}(z) = \frac{\sum_{i=0}^{p} c_i^{p,q}\frac{z^i}{i!}}{\sum_{i=0}^{q} c_i^{q,p}\frac{(-z)^i}{i!}} \quad \text{and} \quad Q_{p,q}(z) = \sum_{i=0}^{q} c_i^{q,p}\frac{(-z)^i}{i!}, \tag{43}$$

where $c_i^{q,p}$ ($c_i^{p,q}$) are defined in (17). Also let $[\tilde{y}_j^{IHO}]$ be an a priori enclosure of the solution on $[t_j, t_{j+1}]$ for any $y_j \in [y_j^{IHO}]$. From (42–43), we compute an enclosure $[y_{j+1}^{IHO}]$ by

$$[y_{j+1}^{IHO}] = R_{p,q}(\lambda h)[y_j^{IHO}] + (-1)^q\frac{\gamma_{p,q}}{Q_{p,q}(\lambda h)}\frac{|\lambda h|^k}{k!}[\tilde{y}_j^{IHO}], \tag{44}$$

where $\gamma_{p,q} = q!p!/(p+q)!$. From (44),

$$w([y_{j+1}^{IHO}]) \geq \left(|R_{p,q}(\lambda h)| + \frac{\gamma_{p,q}}{|Q_{p,q}(\lambda h)|}\frac{|\lambda h|^k}{k!}\right) w([y_j^{IHO}]).$$

Therefore, the IHO method is asymptotically unstable for h such that

$$|R_{p,q}(\lambda h)| + \frac{\gamma_{p,q}}{|Q_{p,q}(\lambda h)|}\frac{|\lambda h|^k}{k!} > 1. \tag{45}$$

In (41) and (45),

$$T_{k-1}(z) = e^z + O(z^k) \quad \text{and} \quad R_{p,q}(z) = e^z + O(z^{p+q+1}) = e^z + O(z^k)$$

are approximations to e^z of the same order. However, $R_{p,q}(z)$ is the Padé rational approximation to e^z (see for example [29]). If z is complex with $\mathrm{Re}(z) < 0$, the following results are known:

- if $p = q$, then $\left|R_{p,q}(z)\right| < 1$ [5]; and
- if $q = p+1$ or $q = p+2$, then $\left|R_{p,q}(z)\right| < 1$, and $\left|R_{p,q}(z)\right| \to 0$ as $|z| \to \infty$ [8]

(see also [19, pp. 236–237]).

Consider (41) and (45). For the ITS method, $\left|T_{k-1}(\lambda h)\right| < 1$ when λh is in the stability region of $T_{k-1}(z)$. However, for the IHO method, $\left|R_{p,q}(\lambda h)\right| < 1$ for any $h > 0$ when $q \geq p$ [33], and $\left|R_{p,q}(\lambda h)\right| \to 0$ as $\lambda h \to -\infty$ when $q > p$. Roughly speaking, the stepsize in the ITS method is restricted by both

$$|T_{k-1}(\lambda h)| \quad \text{and} \quad \frac{|\lambda h|^k}{k!},$$

while in the IHO method, the stepsize is limited mainly by

$$\frac{\gamma_{p,q}}{|Q_{p,q}(\lambda h)|} \frac{|\lambda h|^k}{k!}.$$

In the latter case, $\gamma_{p,q}/Q_{p,q}(\lambda h)$ is usually much smaller than one; thus, the stepsize limit for the IHO method is usually much larger than for the ITS method.

An important point to note here is that an interval version of a standard numerical method that is suitable for stiff problems may still have a restriction on the stepsize. To obtain an interval method without a stepsize restriction, we must find a stable formula not only for the propagated error, but also for the associated truncation error.

In §5.1, we show numerical results comparing the ITS and IHO methods on problem (38) for $\lambda = -10$.

4.2. The $\boldsymbol{n}$-Dimensional Constant Coefficient Case

Consider the IVP

$$y' = By, \qquad y_0 \in [y_0], \tag{46}$$

where $B \in \mathbb{R}^{n \times n}$ and $n > 1$.

We compare one step of an ITS method, which uses Lohner's technique for reducing the wrapping effect, and one step of the IHO method, which uses a similar technique for reducing the wrapping effect. We assume that in addition to an enclosure $[y_j]$ of the solution at t_j, we also have a representation of the enclosure in the form

$$\left\{\hat{y}_j + A_j r_j \mid r_j \in [r_j]\right\},$$

where $\hat{y}_j \in [y_j]$, $A_j \in \mathbb{R}^{n \times n}$ is nonsingular, and $[r_j]$ is an interval vector. We also assume that we have an a priori enclosure of the solution $[\tilde{y}_j]$ on $[t_j, t_{j+1}]$, where $h = t_{j+1} - t_j$.

4.2.1. The Interval Taylor Series Method Using (39), we can write an ITS method, with Lohner's coordinate transformation, as

$$[y_{j+1}^{ITS}] = T_{k-1}(hB)\hat{y}_j + \big(T_{k-1}(hB)A_j\big)[r_j] + [z_{j+1}],$$

where

$$[z_{j+1}] = \frac{h^k}{k!}B^k\,[\tilde{y}_j]\,. \tag{47}$$

The width of $[y_{j+1}^{ITS}]$ is

$$w([y_{j+1}^{ITS}]) = \big|T_{k-1}(hB)A_j\big|w([r_j]) + w([z_{j+1}]). \tag{48}$$

4.2.2. The Interval Hermite-Obreschkoff method Let

$$R_{p,q}(hB) \;=\; \left(\sum_{i=0}^{q} c_i^{q,p}\frac{(-hB)^i}{i!}\right)^{-1}\left(\sum_{i=0}^{p} c_i^{p,q}\frac{(hB)^i}{i!}\right) \quad \text{and} \tag{49}$$

$$Q_{p,q}(hB) \;=\; \sum_{i=0}^{q} c_i^{q,p}\frac{(-hB)^i}{i!}. \tag{50}$$

Using (47) and (49–50), the IHO method can be expressed by

$$\begin{aligned}[y_{j+1}^{IHO}] = R_{p,q}(hB)\hat{y}_j &+ \big(R_{p,q}(hB)A_j\big)[r_j] \\ &+(-1)^q\gamma_{p,q}\big(Q_{p,q}(hB)\big)^{-1}[z_{j+1}].\end{aligned}$$

(Note that for h small, we can compute the inverse of the matrix $Q_{p,q}(hB)$.) The width of $[y_{j+1}^{IHO}]$ is given by

$$w([y_{j+1}^{IHO}]) = \big|R_{p,q}(hB)A_j\big|w([r_j]) + \gamma_{p,q}\big|\big(Q_{p,q}(hB)\big)^{-1}\big|w([z_{j+1}]). \tag{51}$$

Comparing (51) and (48), we see that in the IHO method we multiply the width of the error term, $w([z_{j+1}])$, from the ITS method by $\gamma_{p,q}\big|\big(Q_{p,q}(hB)\big)^{-1}\big|$. If, for example, $p = q = 8$, then $\gamma_{8,8} \approx 7.8\times 10^{-5}$. Consider $\big(Q_{p,q}(hB)\big)^{-1}$ and suppose that $q > 0$. For small h,

$$\big(Q_{p,q}(hB)\big)^{-1} \approx (I - c_1^{q,p}hB)^{-1} \approx I + c_1^{q,p}hB.$$

This implies that for small h, multiplying by the matrix $\big|\big(Q_{p,q}(hB)\big)^{-1}\big|$ does not significantly increase $w([z_{j+1}])$. Moreover, it often happens that

$$\|\big(Q_{p,q}(hB)\big)^{-1}\| < 1.$$

Hence, multiplying by this matrix may reduce $w([z_{j+1}])$ still further.

In Lohner's method, we propagate $\big(T_{k-1}(hB)A_j\big)[r_j]$, where $T_{k-1}(hB)$ is an approximation of the matrix exponential of order $O(h^k)$. In the IHO method, we propagate $\big(R_{p,q}(hB)A_j\big)[r_j]$, where $R_{p,q}(hB)$ is a rational approximation to the matrix exponential of order $O(h^{p+q+1}) = O(h^k)$. If the width of $[r_j]$ is small and hB is small, then we do not have a significant difference in propagating the error in both methods. For example, if $w\left([r_j]\right) = O(h^r)$, then this difference would be $O(h^{k+r})$.

Furthermore, $R_{p,q}(hB)$ has better stability properties than $T_{k-1}(hB)$. We briefly discuss them here. For simplicity, assume that the matrix B in (46) is diagonalizable and can be represented in the form

$$B = X^{-1} D X,$$

where $D = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is a diagonal matrix and $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ are the eigenvalues of B. Suppose that $\mathrm{Re}(\lambda_i) < 0$ for $i = 1, ..., n$. The matrices $T_{k-1}(hD)$ and $R_{p,q}(hD)$ are diagonal with diagonal elements $T_{k-1}(h\lambda_i)$ and $R_{p,q}(h\lambda_i)$, respectively. As h increases, $\|T_{k-1}(hD)\|$ will eventually become greater than one, and then the ITS method is asymptotically unstable. However, for any $h > 0$, $\|R_{p,q}(hD)\| < 1$ for $q = p$, $p+1$, or $p+2$, and $\|R_{p,q}(hD)\| \to 0$ as $h \to \infty$ for $q = p+1$ or $q = p+2$ (see §4.1).

We can also show for the ITS method that

$$w([y_{j+1}^{ITS}]) \geq \frac{h^k}{k!} |B^k| w([y_j^{ITS}]),$$

and for the IHO method that

$$w([y_{j+1}^{IHO}]) \geq \left(\gamma_{p,q} \left| \left(Q_{p,q}(hB)\right)^{-1} \right| \right) \left(\frac{h^k}{k!} |B^k| w([y_j^{ITS}]) \right).$$

In the latter case, the stability restriction on the stepsize usually occurs at values significantly larger than in the former case.

To summarize, we have shown that on constant coefficient problems, our IHO scheme has a much smaller local error and better stability than an ITS method with the same order and stepsize.

In the nonlinear case, we must enclose the solution of a nonlinear system (see §3.2). Computing such an enclosure normally introduces additional errors. It can be shown that if the predicted enclosure is computed with an ITS method of order $(q+1)$, then these errors are sufficiently small [26]. As in the constant coefficient case, the error propagation in the IHO method is similar to that of Lohner's method, but the local error term in the former is much smaller than in the latter [26]. Furthermore, the IHO method requires roughly half the work for generating high-order Jacobians, the computation of which is usually the most expensive part in the IHO and ITS methods (see §4.3).

In §5.2, we show numerical results comparing the two methods on two nonlinear problems.

4.3. Work per Step

We briefly discuss the most expensive parts of the ITS and IHO methods: generating high-order Jacobians, matrix-matrix multiplications, and enclosing the inverse of a point matrix. We measure the work by the number of floating-point operations. However, the time spent on memory operations may not be insignificant for the following reasons.

- The packages for automatic differentiation are often implemented through operator overloading [3], [4], [13], which may involve many memory allocations and deallocations.
- In generating Taylor coefficients, there may be a significant overhead caused by reading and storing the Taylor coefficients, $f^{[i]}$, and their Jacobians [11].

4.3.1. Generating High-Order Jacobians To obtain an approximate bound for the number of floating point operations to generate $k-1$ Jacobians, $\partial f^{[i]}/\partial y$ for $i = 1, \ldots, k-1$, we assume that they are computed by differentiating the code list of the corresponding $f^{[i]}$ and using information from the previously computed $\partial f^{[l]}/\partial y$, for $l = 1, \ldots, i-1$. The FADBAD/TADIFF [3], [4] and IADOL-C [16] packages compute $\partial f^{[i]}/\partial y$ by differentiating the code list of $f^{[i]}$ (IADOL-C is an interval version of ADOL-C [13]). We also assume that the cost of evaluating $\partial f^{[i]}/\partial y$ is roughly n times the cost of evaluating $f^{[i]}$ [11].

For simplicity, suppose that f contains only arithmetic operations. If N is the number of these operations, and c_f is the ratio of multiplications and divisions in N, then to generate $k-1$ coefficients $f^{[i]}$ $(i = 1, \ldots, k-1)$, we need $c_f N k^2 + O(Nk)$ operations [24, pp. 111–112]; to generate $k-1$ Jacobians in an ITS method, we use

$$c_f n N k^2 + O(nNk) \tag{52}$$

arithmetic operations [26]. Let $p = q$ $(k = p + q + 1)$. In the IHO method we generate $p = (k-1)/2$ terms for the forward solution and $q = p = (k-1)/2$ terms for the backward one—the corresponding work is

$$c_f n N k^2/2 + O(nNk). \tag{53}$$

4.3.2. Matrix Inverses and Matrix-Matrix Multiplications In Lohner's method and in the IHO method with the QR-factorization technique, we compute an enclosure of the inverse of a point matrix, which is a floating-point approximation to an orthogonal matrix. However, in the IHO method, we also enclose the inverse of a point matrix (see §3.2). In general, enclosing the inverse of an arbitrary point matrix is more expensive than enclosing the inverse of a floating-point approximation to an orthogonal matrix. However, we can still enclose the inverse of an arbitrary point matrix in $O(n^3)$ operations [2].

Lohner's method has 2 matrix-matrix multiplications, while the IHO method has 6 matrix-matrix multiplications.

To summarize, in the IHO method, we reduce the work for generating Jacobians, but increase the number of matrix operations. Suppose that $N \approx n^2$. This number can be easily achieved if, for example, each component of f contains approximately

n operations. Then, (52) and (53) become

$$c_f n^3 k^2 + O(n^3 k) \quad \text{and} \quad c_f n^3 k^2/2 + O(n^3 k).$$

Therefore, we should expect the IHO method to outperform ITS methods in terms of the amount of work per step when the right side of the problem contains many terms. If the right side contains a few terms only, an ITS method may be less expensive for low orders, but we expect that the IHO method will perform better for higher orders. Note also that we expect the IHO method to allow larger stepsizes for the same order of the truncation error, thus saving computation time during the whole integration. In addition, the IHO method (with $p = q$) needs half the memory for storing the point Taylor coefficients and the high-order Jacobians.

In §5.2.1, we study empirically the amount of work per step on Van der Pol's equation.

5. Numerical Results

The numerical results in this section are produced with the VNODE (Validated Numerical ODE) package [26], which we are developing at the University of Toronto. VNODE is a set of C++ classes that implement existing and new techniques (including the IHO method) used in validated ODE solving. The underlying interval-arithmetic and automatic differentiation packages are PROFIL/BIAS [17] and FADBAD/TADIFF [3], [4], respectively. We compiled VNODE with the GNU C++ compiler version 2.7.2 on a Sun Ultra 2/2170 workstation with an 168 MHz UltraSPARC CPU.

For each of the examples, we measured the (global) error at the end of the interval of integration by taking the maximum norm of the width of the enclosure. Since we are interested mainly in the performance of Algorithm II, we report only the CPU time in seconds spent in this algorithm. For convenience, we denote an ITS method with k terms by ITS(k) and an IHO method with parameters p and q by IHO(q, p). In this section $p = q$ ($k = p + q + 1$).

5.1. Constant Coefficient Problem

5.1.1. Constant Stepsizes We integrated

$$y' = -10y, \quad y(0) = 1 \tag{54}$$

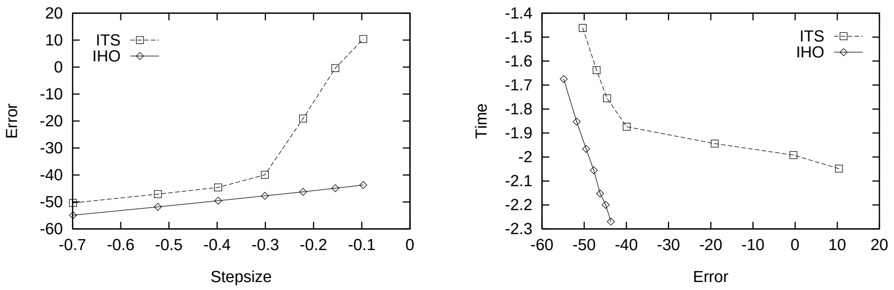
on $[0, 10]$ with constant stepsizes $h = 0.2, 0.3, \ldots, 0.8$. To avoid possible stepsize reductions in Algorithm I [25], we computed a priori enclosures on each step by $[\tilde{y}_j] = [e^{-10h}\underline{y}_j, \bar{y}_j]$. In Algorithm II, we used the ITS(17) and IHO(8, 8) methods.

For stepsizes $0.2, 0.3, \ldots, 0.8$, Table 1 shows the errors at $T = 10$, the ratio of the error of the IHO method to the error of the ITS method, $\gamma_{8,8}/Q_{8,8}(-10h)$, and the CPU time spent in Algorithm II. We compute the base-10 logarithm of the data and plot in Figure 1 the error versus the stepsize and the time versus the error.

Consider Table 1 and Figure 1(a). For "small" stepsizes, $h = 0.2, 0.3, 0.4$, the error in the IHO method is approximately $\gamma_{8,8}/Q_{8,8}(-10h) \approx 10^{-5}$ times the error

Table 1. ITS(17) and IHO(8,8) on $y' = -10y$, $y(0) = 1$, $t \in [0, 10]$.

h	Error		Reductions		Time	
	ITS	IHO	IHO/ITS	$\gamma_{8,8}/Q_{8,8}(-10h)$	ITS	IHO
0.2	$4.4e-51$	$1.3e-55$	$3.0e-05$	$3.0e-05$	$3.5e-02$	$2.1e-02$
0.3	$8.5e-48$	$1.6e-52$	$1.9e-05$	$1.9e-05$	$2.3e-02$	$1.4e-02$
0.4	$2.5e-45$	$2.9e-50$	$1.1e-05$	$1.2e-05$	$1.8e-02$	$1.1e-02$
0.5	$1.2e-40$	$1.8e-48$	$1.5e-08$	$7.8e-06$	$1.3e-02$	$8.8e-03$
0.6	$8.5e-20$	$5.9e-47$	$6.9e-28$	$5.2e-06$	$1.1e-02$	$7.0e-03$
0.7	$4.0e-01$	$1.3e-45$	$3.2e-45$	$3.5e-06$	$1.0e-02$	$6.3e-03$
0.8	$2.4e+10$	$2.0e-44$	$8.4e-55$	$2.4e-06$	$8.9e-03$	$5.4e-03$



(a) (b)

Figure 1. ITS(17) and IHO(8,8) on $y' = -10y$, $y(0) = 1$, $t \in [0, 10]$.

in the ITS method. As h increases beyond 0.4, the ITS method produces enclosures with rapidly increasing widths, while the IHO method computes good enclosures for those stepsizes.

5.1.2. Variable Stepsizes We integrated (54) for $t \in [0, 100]$ with a simple stepsize selection scheme [26] based on controlling the width of the enclosure of the truncation error per unit step. We used an absolute tolerance of 10^{-10}. In Figure 2, we plot the stepsizes against the step number for the two methods. The reduction in the stepsize on the last step for the ITS method is because our program decreases the stepsize to hit the endpoint exactly.

The ITS method is asymptotically unstable for stepsizes h such that

$$|T_{16}(-10h)| + \frac{(10h)^{17}}{17!} > 1$$

(see §4.1). For $h = 0.695$,

$$|T_{16}(-10h)| + \frac{(10h)^{17}}{17!} \approx 1.008.$$

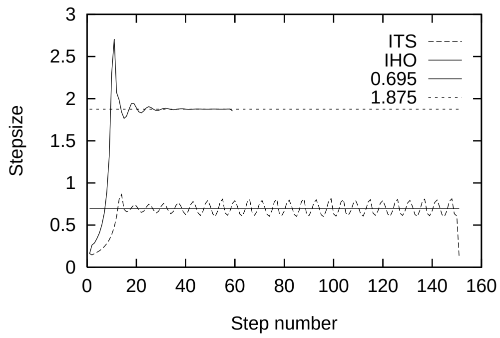


Figure 2. ITS(17) and IHO(8,8) on $y' = -10y$, $y(0) = 1$, $t \in [0, 100]$, variable stepsize control with $Tol = 10^{-10}$.

For the IHO method, the stepsize oscillates around 1.875, which is about 2.7 times bigger than 0.695. For $h = 1.875$,

$$|R_{8,8}(-10h)| + \frac{\gamma_{8,8}}{|Q_{8,8}(-10h)|}\frac{(10h)^{17}}{17!} \approx 0.996.$$

Although the IHO method permits larger stepsizes, they are still limited by its local error term. This observation confirms the conclusions in §4.1.

In the next two examples, we compare the ITS and IHO methods with a variable stepsize control and our version of a Taylor series method for validating existence and uniqueness of the solution [26]. The stepsize control used is as in the previous subsection. The Taylor series approach [24, pp.100–103], [6] for implementing Algorithm I often allows larger stepsizes than the constant enclosure method [9], which has been the most commonly used method for validation [20], [32].

5.2. Nonlinear Case

5.2.1. Van der Pol's equation We integrated Van der Pol's equation, written as a system,

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= \mu(1 - y_1^2)y_2 - y_1, \end{aligned} \tag{55}$$

with

$$y(0) = (2, 0)^T, \tag{56}$$

for $t \in [0, 20]$, where $\mu = 5$. We used the ITS(11) and IHO(5,5) methods and tolerances $10^{-7}, 10^{-8}, \ldots, 10^{-12}$.

From Table 2 and Figure 3, we see that, for approximately the same error, VNODE using the IHO method took fewer steps than the solver using the ITS method, thus saving computation time. Note that we control the local error per unit step and report the global error in Table 2. Thus the global error can be larger than the tolerance.

In Figure 3, we plot the logarithms of the error, time, and tolerance. In Figure 3(d), the stepsize corresponding to the IHO method is not as smooth as the one corresponding to the ITS method. In the regions where the stepsize is not smooth, the Taylor series method for validation could not verify existence and uniqueness with the supplied stepsizes, but verified with reduced stepsizes.

Table 2. ITS(11) and IHO(5,5) on Van der Pol's equation, Taylor series for validation, variable stepsize control.

Tol	Error		Steps		Time	
	ITS	IHO	ITS	IHO	ITS	IHO
$1.0e-07$	$4.0e-08$	$2.8e-07$	352	231	1.5	1.1
$1.0e-08$	$1.3e-08$	$1.6e-08$	384	274	1.6	1.3
$1.0e-09$	$1.2e-09$	$1.1e-09$	473	324	2.0	1.6
$1.0e-10$	$1.3e-10$	$4.4e-10$	587	372	2.6	1.8
$1.0e-11$	$1.3e-11$	$4.8e-11$	731	428	3.1	2.1
$1.0e-12$	$1.8e-12$	$3.5e-12$	912	531	3.9	2.5

We also integrated (55–56) on $[0, 0.1]$ with an input stepsize of 0.01 to Algorithm I. We used orders $k = 3, 7, 11, 17, 25, 31, 37, 43$, and 49 for the ITS method and $p = q = (k-1)/2$ for the IHO method. Algorithm I did not reduce the input stepsize. As a result, the solver could take the same number of steps with the ITS and IHO methods. In Figure 4, we plot the logarithm of the time against the logarithm of the order for these two methods. Although on this problem, the IHO method is more expensive per step for "low" orders, including $k = 11$, we still have savings in time due to the fewer stepsizes taken.

5.2.2. Stiff Detest Problem We integrated the Stiff Detest problem D1 [10],

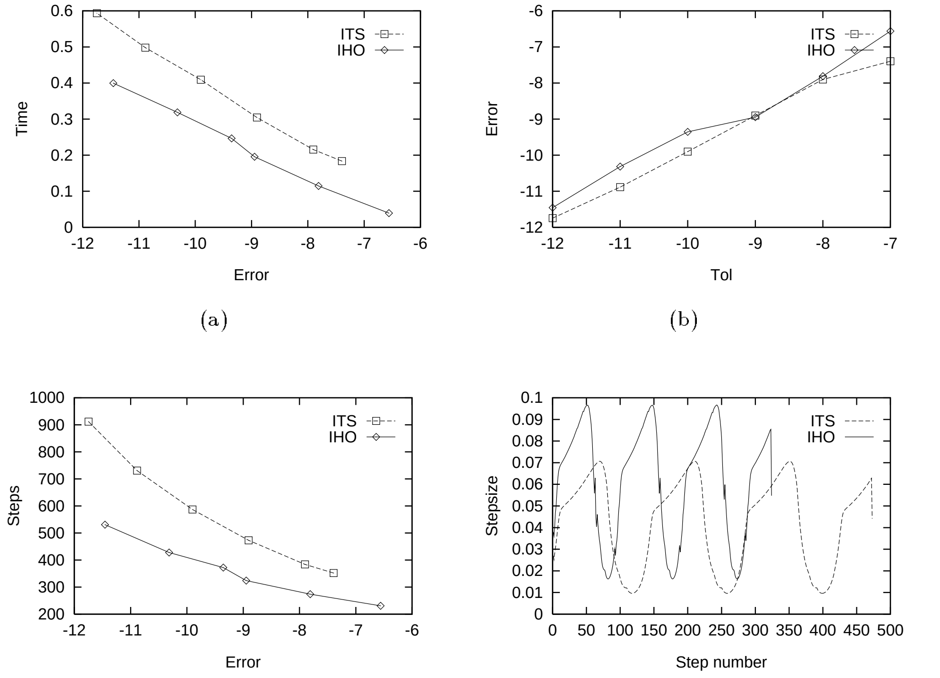
$$\begin{aligned} y_1' &= 0.2(y_2 - y_1) \\ y_2' &= 10y_1 - (60 - 0.125y_3)y_2 + 0.125y_3 \\ y_3' &= 1, \end{aligned} \tag{57}$$

with

$$y(0) = (0,0,0)^T, \quad \text{for} \quad t \in [0, 400]. \tag{58}$$

Here, we used the ITS(17) and IHO(8,8) methods, Taylor series for validation, and a variable stepsize control with tolerances $10^{-6}, 10^{-7}, \ldots, 10^{-10}$.

With the IHO method, we computed tighter bounds with fewer stepsizes, than with the ITS method; see Table 3 and Figure 5. The reduction in the stepsize on the last step for the IHO method seen in Figure 5(d) is a result of our program reducing the stepsize to hit the endpoint exactly.



(a) (b)

(c) (d) $Tol = 10^{-9}$

Figure 3. ITS(11) and IHO(5,5) on Van der Pol's equation, Taylor series for validation, variable stepsize control with $Tol = 10^{-7}, 10^{-8}, \ldots, 10^{-12}$.

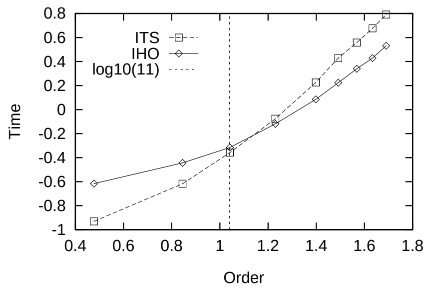
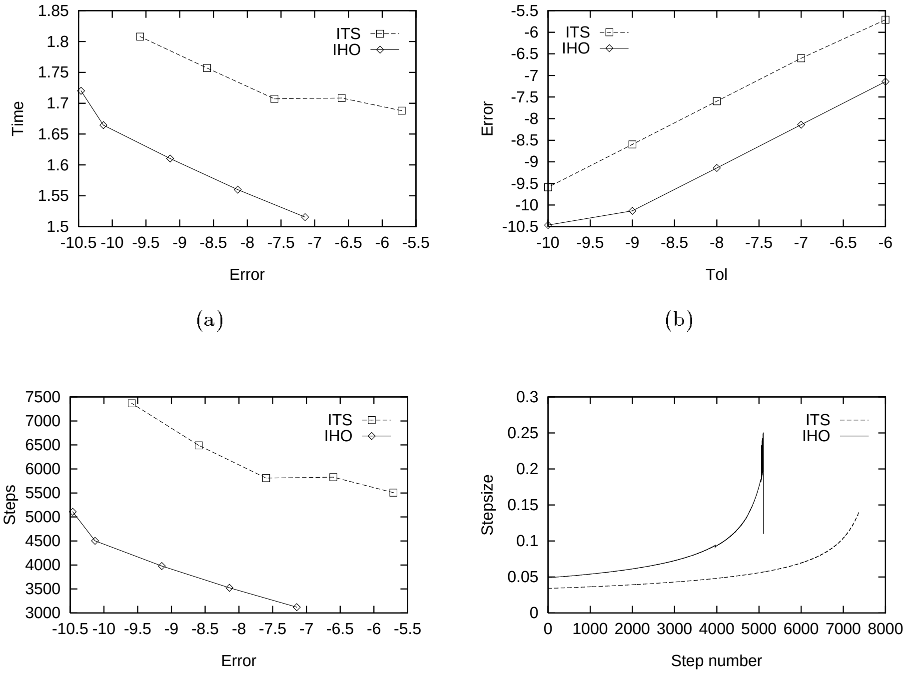


Figure 4. ITS and IHO with orders 3, 7, 11, 17, 25, 31, 37, 43, and 49 on Van der Pol's equation.

Table 3. ITS(17) and IHO(8,8) on (57–58), Taylor series for validation, variable stepsize control.

Tol	Error		Steps		Time	
	ITS	IHO	ITS	IHO	ITS	IHO
$1.0e-06$	$1.9e-06$	$7.2e-08$	5506	3122	49	33
$1.0e-07$	$2.5e-07$	$7.2e-09$	5829	3524	51	36
$1.0e-08$	$2.5e-08$	$7.2e-10$	5811	3977	51	41
$1.0e-09$	$2.5e-09$	$7.4e-11$	6492	4502	57	46
$1.0e-10$	$2.6e-10$	$3.4e-11$	7367	5107	64	52



(a) (b)

(c) (d) $Tol = 10^{-10}$

Figure 5. ITS(17) and IHO(8,8) on (57–58), Taylor series for validation, variable stepsize control with $Tol = 10^{-6}, 10^{-7}, \ldots, 10^{-10}$.

Acknowledgments

The authors thank Dr. George Corliss for his many helpful comments and suggestions on an earlier draft of this paper and for his advice and support throughout the first author's Ph.D. research.

References

1. E. Adams, D. Cordes, and R. Lohner. Enclosure of solutions of ordinary initial value problems and applications. In E. Adams, R. Ansorge, Chr. Großmann, and H. G. Roos, editors, *Discretization in Differential Equations and Enclosures*, pages 9–28. Akademie-Verlag, Berlin, 1987.
2. G. Alefeld and J. Herzberger. *Introduction to Interval Computations.* Academic Press, New York, 1983.
3. Claus Bendsten and Ole Stauning. FADBAD, a flexible C++ package for automatic differentiation using the forward and backward methods. Technical Report 1996-x5-94, Department of Mathematical Modelling, Technical University of Denmark, DK-2800, Lyngby, Denmark, August 1996. FADBAD is available at http://www.imm.dtu.dk/fadbad.html.
4. Claus Bendsten and Ole Stauning. TADIFF, a flexible C++ package for automatic differentiation using Taylor series. Technical Report 1997-x5-94, Department of Mathematical Modelling, Technical University of Denmark, DK-2800, Lyngby, Denmark, April 1997. TADIFF is available at http://www.imm.dtu.dk/fadbad.html.
5. G. Birkhoff and R. S. Varga. Discretization errors for well-set Cauchy problems: I. *J. Math. and Phys.*, 44:1–23, 1965.
6. 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.
7. G. Darboux. Sur les dèveloppements en série des fonctions d'une seule variable. *J. des Mathématique pures et appl.*, pages 291–312, 1876. 3ème série, t. II.
8. B. L. Ehle. On Padé approximations to the exponential function and A-stable methods for the numerical solution of initial value problems. *SIAM J. Math. Anal.*, 4:671–680, 1973.
9. P. Eijgenraam. *The Solution of Initial Value Problems Using Interval Arithmetic.* Mathematical Centre Tracts No. 144. Stichting Mathematisch Centrum, Amsterdam, 1981.
10. W. H. Enright, T. E. Hull, and B. Lindberg. Comparing numerical methods for stiff systems of ODEs. *BIT*, 15:10–48, 1975.
11. Andreas Griewank. ODE solving via automatic differentiation and rational prediction. In D. F. Griffiths and G. A. Watson, editors, *Numerical Analysis 1995*, volume 344 of *Pitman Research Notes in Mathematics Series.* Addison-Wesley Longman Ltd, 1995.
12. Andreas Griewank, George F. Corliss, Petra Henneberger, Gabriella Kirlinger, Florain A. Potra, and Hans J. Stetter. High-order stiff ODE solvers via automatic differentiation and rational prediction. In *Lecture Notes in Comput. Sci., 1196*, pages 114–125. Springer, Berlin, 1997.
13. Andreas Griewank, David Juedes, and Jean Utke. ADOL-C, a package for the automatic differentiation of algorithms written in C/C++. *ACM Trans. Math. Software*, 22(2):131–167, June 1996.
14. E. Hairer, S. P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I. Nonstiff Problems.* Springer-Verlag, 2nd revised edition, 1991.
15. Ch. Hermite. Extrait d'une lettre de M. Ch. Hermite à M. Borchardt sur la formule d'interpolation de Lagrange. *J. de Crelle*, 84(70):70, 1878. Oeuvres, tome III, p. 432–443.
16. Ron Van Iwaarden. IADOL-C, personal communications, 1997.
17. O. Knüppel. PROFIL/BIAS – a fast interval library. *Computing*, 53(3–4):277–287, 1994. PROFIL/BIAS is available at http://www.ti3.tu-harburg.de/Software/PROFIL/Profil.texinfo_1.html.
18. F. Krückeberg. Ordinary differential equations. In Eldon Hansen, editor, *Topics in Interval Analysis*, pages 91–97. Clarendon Press, Oxford, 1969.

19. J. D. Lambert. *Computational Methods in Ordinary Differential Equations.* John Wiley & Sons, 1977.
20. Rudolf J. Lohner. *Einschließung der Lösung gewöhnlicher Anfangs– und Randwertaufgaben und Anwendungen.* PhD thesis, Universität Karlsruhe, 1988.
21. Rudolph J. Lohner. Enclosing the solutions of ordinary initial and boundary value problems. In Edgar W. Kaucher, Ulrich W. Kulisch, and Christian Ullrich, editors, *Computer Arithmetic: Scientific Computation and Programming Languages*, pages 255–286. Wiley-Teubner Series in Computer Science, Stuttgart, 1987.
22. Ramon E. Moore. The automatic analysis and control of error in digital computation based on the use of interval numbers. In Louis B. Rall, editor, *Error in Digital Computation, Vol. I*, pages 61–130. Wiley, New York, 1965.
23. Ramon E. Moore. Automatic local coordinate transformations to reduce the growth of error bounds in interval computation of solutions of ordinary differential equations. In Louis B. Rall, editor, *Error in Digital Computation, Vol. II*, pages 103–140. Wiley, New York, 1965.
24. Ramon E. Moore. *Interval Analysis.* Prentice-Hall, Englewood Cliffs, N.J., 1966.
25. N. S. Nedialkov, K. R. Jackson, and G. F. Corliss. Validated solutions of initial value problems for ordinary differential equations. To appear in *Applied Mathematics and Computation.* Available at http://www.cs.toronto.edu/NA/reports.html.
26. Nedialko Stoyanov 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. Available at http://www.cs.toronto.edu/NA/reports.html.
27. N. Obreschkoff. Neue Quadraturformeln. *Abh. Preuss. Akad. Wiss. Math. Nat. Kl.*, 4, 1940.
28. N. Obreschkoff. Sur le quadrature mecaniques. *Spisanie Bulgar. Akad. Nauk. (Journal of the Bulgarian Academy of Sciences)*, 65:191–289, 1942.
29. A. Ralston. *A First Course in Numerical Analysis.* McGraw-Hill, New York, 2nd edition, 1978.
30. Robert Rihm. On a class of enclosure methods for initial value problems. *Computing*, 53:369–377, 1994.
31. J. Rohn. NP-hardness results for linear algebraic problems with interval data. In J. Herzberger, editor, *Topics in Validated Computations*, volume 5 of *Studies in Computational Mathematics*, pages 463–471, North-Holland, Amsterdam, 1994.
32. Ole Stauning. Automatic validation of numerical solutions. Technical Report IMM-PHD-1997-36, IMM, Lyngby, Denmark, October 1997.
33. R. S. Varga. On higher order stable implicit methods for solving parabolic differential equations. *J. Math. and Phys.*, 40:220–231, 1961.
34. G. Wanner. On the integration of stiff differential equations. Technical report, Université de Genéve, Section de Mathematique, 1211 Genéve 24th, Suisse, October 1976.
35. G. Wanner. On the integration of stiff differential equations. In *Proceedings of the Colloquium on Numerical Analysis*, volume 37 of *Internat. Ser. Numer. Math.*, pages 209–226, Basel, 1977. Birkhäuser.