Long Time Integration for Initial Value Problems of Ordinary Differential Equations Using Affine Arithmetic

Takatomi Miyata and Masahide Kashiwagi

Waseda University
Okubo 3-4-1, Shinjuku-ku, Tokyo 169-8555, Japan
Phone: +81-3-5286-3484, Fax:+81-3-5286-3362
Email: miyata@kashi.info.waseda.ac.jp, kashi@mn.waseda.ac.jp

1. Validated one-step method to solve IVPs

In this paper, we consider the following IVPs

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = f(x(t),t), \quad t \in [a,b], \quad a \ll b, \quad x(a) = x_0. \quad (1)$$

Here, $x(t)$ is an n-dimensional vector valued function, $f(x(t))$ is the function for which *Power Series Arithmetic* (see the section **2.**) can be defined. And we assume that b is much greater than a.

Generally, solving ODEs numerically means calculating values $x(t_i)$ at discrete points $\{t_i\}$, where

$$a = t_0 < t_1 < t_2 < \cdots < t_m = b. \quad (2)$$

So we often discretize the equation (1) to a finite dimensional equation with unknown variables x_i which approximate $x(t_i)$. In this process, discretization error arises and it is difficult to estimate the error.

On the other hand, a lot of methods have been proposed to solve IVPs of ODEs with guaranteed accuracy. Lorner's method [1] is one of the most famous method in all these methods. It is a validated one-step method to solve IVPs. In this method, an algorithm is proposed by which we can solve the following equation with guaranteed accuracy:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = f(x(t),t), \quad t \in [t_s,t_e], \quad x(t_s) = v. \quad (3)$$

In other words, an algorithm is proposed to calculate a forward operator $\phi(v,t_s,t_e) : \boldsymbol{R}^n \times \boldsymbol{R} \times \boldsymbol{R} \to \boldsymbol{R}^n$ which returns inclusion of $x(t_e)$ provided that $x(t_s) = v$. By using this ϕ, equation (1) can be represented as follows:

$$x_{i+1} = \phi(x_i, t_i, t_{i+1}), \quad (i = 0, \cdots, m-1). \quad (4)$$

Here, x_i is not an approximation but an inclusion of the value of exact $x(t_i)$. By using (4), we can obtain inclusions $x_1, \cdots, x_m$, one after another.

We also proposed a validated one-step method to solve IVPs [2]. In this method, we calculate a forward operator ϕ by an algorithm proposed by Kashiwagi [3][4], one of the authors. We show the algorithm in the next section.

2. Kashiwagi's Algorithm

In this section, we will introduce Kashiwagi's algorithm [3][4]. It is a method to obtain an inclusion of the exact $x(t_i)$. For his algorithm, two types of arithmetics of power series are defined. We call them power series arithmetics (PSAs).

2.1. Type-I Power Series Arithmetic

Type-I PSA treats n-degree power series and truncate terms higher than n-th degree. In the following, $[a_0, a_1, a_2, \cdots, a_n]$ denotes power series

$$a_0 + a_1 t + a_2 t^2 + \cdots a_n t^n \quad (+O(t^{n+1})). \quad (5)$$

Addition, subtraction and multiplication of Type-I PSA are done as follows:

$$[a_0, \cdots, a_n] \pm [b_0, \cdots, b_n] = [a_0 \pm b_0, \cdots, a_n \pm b_n], \quad (6)$$

$$[a_0, \cdots, a_n] \times [b_0, \cdots, b_n] = [c_0, \cdots, c_n],$$

$$c_k = \sum_{i=0}^{k} a_i b_{k-i}. \quad (7)$$

Functions are applied as follows:

$$f([a_0, \cdots, a_n]) = f(a_0) + \sum_{i=1}^{n} \frac{1}{i!} f^{(i)}(a_0)[0, a_1, \cdots, a_n]^i. \quad (8)$$

Addition and multiplication in(8) are done by Type-I PSA ((6) and (7)).

Division is executed by combining inverse function and multiplication ($x/y = x \times (1/y)$).

Integration is defined as follows:

$$\int_0^t [a_0, \cdots, a_n]\, \mathrm{d}t = [0, a_0, \frac{a_1}{2}, \cdots, \frac{a_n}{n+1}]. \quad (9)$$

Type-I PSA keeps first $(n+1)$ terms of no-truncated arithmetic. Several mathmatical softwares as Mathematica provide such an arithmetic.

2.2. Type-II Power Series Arithmetic

Type-II PSA also treats n-degree power series. In type-II PSA we must specify its domain as $[0,d]$. $[a_0,\cdots,a_n]$ also denotes

$$a_0 + a_1 t + a_2 t^2 + \cdots a_n t^n, \tag{10}$$

but a_n is generally interval and $[a_0,\cdots,a_n]$ represents set of functions defined in $[0,d]$ such that $f(t) \in a_0 + \cdots + a_n t^n$ in all t.

Addition and subtraction are same as Type-I PSA. Multiplication is executed by the following steps:

(1) Multiply $[a_0,\cdots,a_n]$ and $[b_0,\cdots,b_n]$ without truncation. Result $C = [c_0,\cdots,c_{2n}]$ is $2n$-degree.

(2) Reduce the degree of C from $2n$ to n.

Degree reduction is defined as follows:

Definition 1 (Degree Reduction = Rounding) Let $A = [a_0,\cdots,a_k]$ be power series and n be smaller than k. Then degree reduction of A from $2n$ to n is defined as the following $B = [b_0,\cdots,b_n]$:

$$b_i = a_i \quad (0 \le i \le n-1), \tag{11}$$

$$b_n = \{\sum_{i=n}^{k} a_i t^{i-n} \mid t \in [0,d]\}. \tag{12}$$

Higher order remainder is added to b_n of (12), by transforming terms higher than n-th degree $a_i t^i$ to $(a_i t^{i-n})t^n$ and estimating $a_n + a_{n+1}t + \cdots + a_k t^{k-n}$. Thus the result of multiplication C contains all possible results derived by multiplication without truncation.

Functions are applied as follows:

$$f([a_0,\cdots,a_n]) = f(a_0) + \sum_{i=1}^{n-1} \frac{1}{i!} f^{(i)}(a_0)[0,a_1,\cdots,a_n]^i + \frac{1}{n!} f^{(i)}(\xi)[0,a_1,\cdots,a_n]^n. \tag{13}$$

Here, ξ is an interval represented as follows:

$$\xi = \{\sum_{i=0}^{n} a_i t^i \mid t \in [0,d]\}. \tag{14}$$

Addition and multiplication in (13) are done by Type-II PSA. This algorithm uses the Lagrange's remainder term.

Like Type-I PSA, division is done by calculating $x \times (1/y)$. Of course, multiplication and inverse function are done by Type-II PSA.

Integration is defined like Type-I PSA:

$$\int_0^t [a_0,\cdots,a_n]\,\mathrm{d}t = [0, a_0, \frac{a_1}{2}, \cdots, \frac{a_n}{n+1}]. \tag{15}$$

2.3. How to Calculate $\phi(v,t_s,t_e)$

Now, we will introduce Kashiwagi's algorithm [3]. It is a method to calculate $\phi(v,t_s,t_e)$ which returns inclusion of $x(t_e)$ provided that $x(t_s) = v$.

$\phi : \boldsymbol{R}^n \times \boldsymbol{R} \times \boldsymbol{R} \to \boldsymbol{R}^n$ can be calculated by obtaining verified solution of the following initial value problem:

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = f(x(t),t), \quad x(t_s) = v,\ t \in [t_s,t_e]. \tag{16}$$

If $x(t)$ is calculated exactly, $\phi(v,t_s,t_e)$ is obtained by $x(t_e)$.

In this algorithm, we obtain enclosure of $x(t)$ based on Picard's iterative method. That is, i) we convert (16) to equivalent fixed point form:

$$x(t) = v + \int_{t_s}^{t} f(x(s),s)\ \mathrm{d}s, \tag{17}$$

ii) obtain an approximate solution by iteration started from constant function $x(t) = v$, and iii) prove existence of exact solution by Schauder's fixed point theorem.

Now we introduce the algorithm. It is noticed that independent variable t is shifted ($[t_s,t_e] \to [0,t_e - t_s]$) in the following.

Algorithm 1 We assume that $t_s < t_e$. Let $\Delta = t_e - t_s$ and domain for Type-II PSA be $[0,\Delta]$.

Step-1 Initialize power series vector X as

$$X = ([v_1],\cdots,[v_n])^t. \tag{18}$$

Here, superscription t denotes the transposition of vectors.

Step-2 (generate approximate solution) Repeat Step-2(1)-Step-2(3) appropriate times.

Step-2(1) Set power series T as $(t_s + t)$ truncated within k-degree:

$$T = \begin{cases} [t_s] & (k=0) \\ [t_s,1] & (k=1) \\ [t_s,1,0,\cdots,0] & (k \ge 2). \end{cases} \tag{19}$$

Step-2(2) Calculate $X = f(X,T)$ by Type-I PSA. Calculate $X = v + \int_0^t X(s)\,\mathrm{d}s$. By these operations degrees of X increases from k to $k+1$.

Step-2(3) $k = k+1$

Step-3 Set $T = [t_s,1,0,\cdots,0]$ (k degree).

Step-4 Calculate $Y = f(X,T)$ by Type-II PSA, calculate $Y = v + \int_0^t Y(s)\,\mathrm{d}s$, and reduce the degree of Y from $k+1$ to k by Definition 1.

Step-5 Calculate

$$r = \max_{1 \le i \le n} \left| Y_i^{(k)} - X_i^{(k)} \right|, \tag{20}$$

where suffix $^{(j)}$ denotes coefficient of t^j.

Step-6 Let $X_i^{(k)} = X_i^{(k)} + [-2r, 2r]$ for all $1 \le i \le n$.

Step-7 Calculate $Y = f(X, T)$ by Type-II PSA, calculate $Y = v + \int_0^t Y(s)\,ds$, and reduce the degrees of Y from $k+1$ to k by Definition 1.

Step-8 (existence test) If $Y_i^{(k)} \subset X_i^{(k)}$ for all $1 \le i \le n$, then the existence of exact solution in Y is guaranteed by Shauder's fixed point theorem.

If this test has failed, let the value of step length Δ decrease (back to **Step-1**), or let the number of iteration (**Step-2**) increase to generate a better approximate solution(back to **Step-2**).

Step-9 (refinement of solution)

Step-9(1) Calculate $X = f(Y, T)$ by Type-II PSA, calculate $X = v + \int_0^t X(s)\,ds$, and reduce the degrees of X from $k+1$ to k by Definition 1.

Step-9(2) Let $Y_i^{(k)} = Y_i^{(k)} \cap X_i^{(k)}$ for all $1 \le i \le n$.

Step-10 $\phi(v, t_s, t_e)$ is obtained by

$$\begin{aligned} x(t_e) &= Y(t_e - t_s) \\ &= \left(\sum_{i=0}^{k} Y_1^{(i)} \Delta^i, \cdots, \sum_{i=0}^{k} Y_n^{(i)} \Delta^i \right)^t. \end{aligned} \tag{21}$$

∎

At Step-8, $Y_i^{(j)} = X_i^{(j)}$ for $0 \le j \le k-1$ always holds in this algorithm, hence only the last term is needed to test.

3. Affine Arithmetic

Affine arithmetic (AA) [5] is a variant of IA. AA keeps correlation between quantities. So AA can overcome the error explosion problem often observed in standard IA, and it often produces much tighter bounds than a bound standard IA yields.

3.1. Affine form

In AA, a quantity x is represented as the following affine polynomial:

$$x = x_0 + x_1 \varepsilon_1 + \cdots + x_n \varepsilon_n. \tag{22}$$

Here, coefficients $x_i\,(i = 0, 1, \cdots, n)$ are known real numbers, and $\varepsilon_i\,(i = 1, \cdots, n)$ are symbolic real variables whose values are unknown but assumed to lie in the interval $[-1, 1]$. Only coefficients $x_i\,(i = 0, 1, \cdots, n)$ are memoried in the computer. Remark that n, the number of the noise symbol ε_k, changes through the calculation.

3.2. Initialization

Given an interval $[\underline{x}, \overline{x}]$ representing some quantity x in standard IA, an equivalent affine form for the same quantity x is given by the following:

$$\frac{\underline{x} + \overline{x}}{2} + \frac{\overline{x} - \underline{x}}{2} \varepsilon, \quad (-1 \le \varepsilon \le 1). \tag{23}$$

ε represents the uncertainty in the value of x that is implicit in its standard interval representation. For a vector or a matrix, we must give a separate noise symbol ε_k for each component, because there are no relation between components in general.

3.3. Conversion from AA to IA

If a quantity x is described by the affine form like the formula (22), then it can be always converted to the following standard interval representation:

$$x \in [x_0 - \delta_x, x_0 + \delta_x], \quad \delta_x = \sum_{i=1}^{n} (|x_i|). \tag{24}$$

And the interval $[x_0 - \delta_x, x_0 + \delta_x]$ in the formula (24) is the smallest interval that contains all possible values of affine form x in the formula (22). But we had better not to convert quantities from affine form to standard interval representation, since we must lose any knowledge of correlations between quantities that was preserved in their affine forms.

3.4. Affine operations

For affine polynomials x, y and a real number α, rules of affine operations for AA is defined as follows:

$$x \pm y = (x_0 \pm y_0) + \sum_{i=1}^{n} (x_i \pm y_i)\, \varepsilon_i \tag{25}$$

$$x \pm \alpha = (x_0 \pm \alpha) + \sum_{i=1}^{n} x_i \varepsilon_i \tag{26}$$

$$\alpha\, x = (\alpha x_0) + \sum_{i=1}^{n} (\alpha x_i)\, \varepsilon_i \tag{27}$$

3.5. Non-Affine operations

Let f be a nonlinear unary operation. Generally, $f(x)$ for some affine polynomial x can not be expressed as an affine polynomial. Then we consider to approximate f by a linear operation and express the approximation error by introducing a new noise symbol ε_{new}.

Let the following affine polynomial:

$$z_0 + z_1\varepsilon_1 + \cdots + z_n\varepsilon_n \tag{28}$$

be the image of a linear-approximated operation of f, and δ be its approximation error. And we obtain the following inclusion of $f(x)$ by introducing a new noise symbol ε_{n+1} for the maximum error term:

$$(z_0 + z_1\varepsilon_1 + \cdots + z_n\varepsilon_n) + \delta\,\varepsilon_{n+1}. \tag{29}$$

The inclusion is not determined uniquely. We can vary the values of $z_0, \cdots, z_n$. The tightness of the inclusion has great influence on the efficiency of the interval evaluation. δ is optimized as follows:

$$\delta = \max_{-1\le\varepsilon_i\le 1}\left[f(x) - (z_0 + z_1\varepsilon_1 + \cdots + z_n\varepsilon_n)\right] \tag{30}$$

If it is difficult to evaluate the value of optimized δ in the formula (30), we substitute its upper bound for the optimized δ.

We usually approximate $f(x)$ by a linear operation $ax + b$, and determine the value of approximation error as follows:

$$\delta' = \max_{x\in[\underline{x},\overline{x}]}\left[f(x) - (ax + b)\right] \tag{31}$$

Remark that

$$\min_{a,b\in\boldsymbol{R}}\delta' = \min_{z_i\in\boldsymbol{R}}\delta \tag{32}$$

always holds if f is a nonlinear unary operation.

The way for nonlinear binomial operation is basically the same way as the way for nonlinear unary operation. But remark that

$$\min_{z_i\in\boldsymbol{R}}\left(\max_{-1\le\varepsilon_i\le 1}\left[f(x,y) - (z_0 + z_1\varepsilon_1 + \cdots + z_n\varepsilon_n)\right]\right)$$
$$= \min_{a,b,c\in\boldsymbol{R}}\left(\max_{\substack{x\in[\underline{x},\overline{x}]\\ y\in[\underline{y},\overline{y}]}}\left[f(x,y) - (ax + by + c)\right]\right) \tag{33}$$

does not hold in general.

4. Introduction of our new method

In our old method [2], we obtain the inclusion of the exact $x(t_i)$ as an interval. Of course, we use interval arithmetic (IA) in our calculation. But the inclusion, owing to the property of interval arithmetic, becomes wider and wider as i increases. Hence the inclusion, at last, becomes too wide from a practical point of view.

To overcome this problem, in this method, we combined Kashiwagi's algorithm with mean value form, and considered the wrapping effect, too. But the wrapping effect was not eliminated in this method, though it was certainly reduced.

Now we propose a new method. In this method, we use Kashiwagi's algorithm, too. But we calculate the value of inclusion of the exact $x(t_i)$ by Affine Arithmetic (AA, see the section 3) instead of a conventional IA. We will show the efficiency of our new method in our talk.

References

[1] R.J.Lohner: "Enclosing the Solutions of Ordinary Initial and Boundary Value Problems", In E.Kaucher, U.Kulisch and Ch.Ullrich (eds.): "Computer Arithmetic, Scientific Computation and Programming Languages", B.G.Teubner, Stuttgart, pp.255-286, 1987.

[2] Takatomi Miyata, Yasutaka Nagatomo and Masahide Kashiwagi: "Long Time Integration for Initial Value Problems of Ordinary Differential Equations Using Power Series Arithmetic", IEICE Trans. Vol.E84-A, No.9, pp.2230-2237, September 2001.

[3] M.Kashiwagi, S.Oishi: "Numerical Validation for Ordinary Differential Equations -Iterative Method by Power Series Arithmetic-", Proc. 1994 International Symposium on Nonlinear Theory and its Applications (NOLTA '94), pp.243-246 (1994-10).

[4] Masahide Kashiwagi: "Power Series Arithmetic and its Application to Numerical Validation", Proc. 1995 International Symposium on Nonlinear Theory and its Applications (NOLTA '95 Symposium), pp.251–254 (Las Vegas, U.S.A., 10–14 December 1995).

[5] Marcus Vinícius A.Andrade, João L. D. Comba and Jorge Stolfi: "Affine Arithmetic", INTERVAL '94, St. petersburg (Russia), March 5-10, 1994.

[6] Luiz H. de Figueiredo and Jorge Stolfi: "Self-Validated Numerical Methods and Applications", Brazilian Mathematics Colloquium monograph, IMPA, Rio de Janeiro, Brazil; July 1997.

[7] Takatomi Miyata and Masahide Kashiwagi: "On Range Evaluation of Polynomials of Affine Arithmetic", Proc. of International Symposium on Nonlinear Theory and its Applications (NOLTA2001), pp.227-230, (2001-10).