(Version: 2014/1/31)

ベキ級数演算について

柏木 雅英

1 はじめに

(本文書は、[1],[2],[3] の内容を元にまとめたものである。)

数値計算において、計算を行うと同時にその結果の誤差評価をも同時に計算するような方法を総称して精度保証付き数値計算と呼び、近年急速な進歩を遂げている。精度保証付き数値計算の実現において最も基本的かつ重要な技法に、区間演算が挙げられる。区間演算とは、実数値を [下限, 上限] という 2 つの浮動小数点数で挟まれた区間で表現し、その区間同士の加減乗除等の演算を「演算結果として有り得る集合を包含するように」定義することにより行われるものである。そのとき、区間の両端を計算する際に丸めの向きを「外向き」にしておくことによって丸め誤差の影響分を区間内に収め、丸め誤差の把握を行うことが出来る。また、区間演算ではその定義の仕方から関数の値域 (区間入力の像) の評価を行うことが出来る。これを利用すると、不動点定理を通じて方程式の解の精度保証を行うことが出来る。

一方、微分方程式などの「未知数が関数」な問題においても、同様の考え方で解の精度保証を行うことが出来る。本稿では、そのような「区間関数」の表現方法の一つであるベキ級数演算について解説する。また、それを応用した常微分方程式の初期値問題、関数値の評価、高階微分の計算、数値積分などの精度保証付き計算法についても述べる。

2 区間演算

区間演算とは、実数値を [下限, 上限] という 2 つの数で挟まれた区間で表現し、その区間同士の加減乗除等の演算を「演算結果として有り得る集合を包含するように」定義することにより行われるものである。つまり、区間演算は、区間 X, Y と、二項演算 $\cdot \subset \{+, -, \times, \div\}$、単項演算 g について、

$$\begin{aligned} X \cdot Y &\supset \{x \cdot y \mid x \in X, y \in Y\} \\ g(X) &\supset \{f(x) \mid x \in X\} \end{aligned}$$

を満たすような集合演算として定められる。例えば、$\underline{X}, \overline{X}$ をそれぞれ区間 X の下限、上限として、

$$\begin{aligned} X + Y &= [\underline{X} + \underline{Y}, \overline{X} + \overline{Y}] \\ X - Y &= [\underline{X} - \overline{Y}, \overline{X} - \underline{Y}] \\ X \times Y &= [\min(\underline{X}\underline{Y}, \underline{X}\overline{Y}, \overline{X}\underline{Y}, \overline{X}\overline{Y}), \max(\underline{X}\underline{Y}, \underline{X}\overline{Y}, \overline{X}\underline{Y}, \overline{X}\overline{Y})] \\ X/Y &= [\min(\underline{X}/\underline{Y}, \underline{X}/\overline{Y}, \overline{X}/\underline{Y}, \overline{X}/\overline{Y}), \max(\underline{X}/\underline{Y}, \underline{X}/\overline{Y}, \overline{X}/\underline{Y}, \overline{X}/\overline{Y})] \quad (Y \not\ni 0) \\ \sqrt{X} &= [\sqrt{\underline{X}}, \sqrt{\overline{X}}] \quad (X \geq 0) \end{aligned}$$

とすれば良い。区間演算が定義されているようなこの二項演算と単項演算の組合せで書かれた全ての関数 f について、区間演算を行えばその関数の値域の評価を行うことが出来る。すなわち、f の

定義域に含まれる区間 X に対して、その関数を構成する全ての二項演算、単項演算に対して区間演算を行うことにより得られた結果を $f(X)$ とすると、

$$f(X) \supset \{f(x) \mid x \in X\}$$

となることは明らかである。

多くの計算機で用いられている浮動小数点計算で区間演算を実現するには、上端と下端を浮動小数点数で保持するのが普通である (下端上端型。中心半径型もある)。下端上端型の場合、例えば区間 X と Y の和 Z は、

$$\begin{aligned} \underline{Z} &\leq \underline{X} + \underline{Y} \\ \overline{Z} &\geq \overline{X} + \overline{Y} \end{aligned}$$

を満たすように計算する。CPU の計算時に発生する丸めの向きを変更できる環境の場合、丸めを $-\infty$ 方向に変更してから $\underline{X} + \underline{Y}$ を計算することにより所望の $\underline{Z}$ を、丸めを $+\infty$ 方向に変更してから $\overline{X} + \overline{Y}$ を計算することにより所望の $\overline{Z}$ を得ることが出来る。このような計算により、区間演算は丸め誤差の把握を行うことが出来る。

3 ベキ級数演算

ベキ級数演算 (Power Series Arithmetic、以下 PSA と略す) は、有限項で打ち切られた多項式

$$x_0 + x_1 t + x_2 t^2 + \cdots + x_n t^n$$

同士の演算を行うものである。高次の項を捨ててしまう Type-I と、高次の項の影響を捨てずに最高次の係数 x_n に入れ込む Type-II の二通りの演算がある。それぞれ、仮数部に入りきらない分を捨ててしまう浮動小数点数による近似計算と、仮数部に入りきらない部分の影響を区間という形で保持する区間演算に対応させて考えると分かりやすい。

3.1 Type-I PSA

Type-I PSA では、n 次のベキ級数

$$x(t) = x_0 + x_1 t + x_2 t^2 + \cdots + x_n t^n$$

同士の演算を行い、その際 $n+1$ 次以上の項は切り捨ててしまう。

加減算は次の様に定義する。

$$x(t) \pm y(t) = (x_0 \pm y_0) + (x_1 \pm x_2)t + \cdots + (x_n + y_n)t^n$$

乗算は、

$$\begin{aligned} x(t) \times y(t) &= z_0 + z_1 t + z_2 t^2 + \cdots + z_n t^n \\ z_k &= \sum_{i=0}^{k} x_i y_{k-i} \end{aligned}$$

のように、高次項を切り捨てて行われる。

sin などの数学関数の適用は、その関数を g として、

$$\begin{aligned} & g(x_0 + x_1 t + \cdots + x_n t^n) \\ & = g(x_0) + \sum_{i=1}^{n} \frac{1}{i!} g^{(i)}(x_0)(x_1 t + \cdots + x_n t^n)^i \end{aligned}$$

のように g の点 x_0 での Taylor 展開に代入することによって得る。但し、途中に現れる加減算や乗算は上記 Type-I PSA によって行う。

除算は、$x \div y = x * (1/y)$ と乗算と逆数関数に分解することによって行う。

不定積分は、

$$\int_0^t x(t)\mathrm{d}t = x_0 t + \frac{x_1}{2} t^2 + \cdots \frac{x_n}{n+1} t^{n+1}$$

のように行う。

Type-I PSA と同様の演算は、

- Mathematica の `Series`
- intlab の taylor toolbox

などで行うことが出来る。また、[4] の 3.9 節で述べられている高階微分を求める方法も実質的にほぼ同一と見なせる。

Type-I PSA の係数は基本的に点 (実数) であるが、浮動小数点数を用いて実装する場合はきちんと精度保証するためには係数を区間にする必要がある。

3.2 Type-I PSA の例

Type-I PSA の簡単な例を示す。次数は 2 とする。

$$\begin{aligned} x(t) &= 1 + 2t - 3t^2 \\ y(t) &= 1 - t + t^2 \end{aligned}$$

に対して、加減算は、

$$\begin{aligned} x(t) + y(t) &= 2 + t - 2t^2 \\ x(t) - y(t) &= 0 + 3 - 4t^2 \end{aligned}$$

となる。乗算は、

$$x(t) \times y(t) = 1 + t - 4t^2 + 5t^3 - 3t^4$$

を t^2 の項までで打ち切って、

$$x(t) \times y(t) = 1 + t - 4t^2$$

を計算結果とする。

数学関数の例を示す。$\log(x(t))$ は、まず、log の 1 ($x(t)$ の定数項) における Taylor 展開 (2 次まで) を作る:

$$0 + (x-1) - \frac{1}{2}(x-1)^2$$

これに $x(t)$ を代入すると、

$$0 + (2t - 3t^2) - \frac{1}{2}(2t - 3t^2)^2$$

となるが、これを Type-I PSA で計算すると (つまり乗算で 3 次以降は削ると)、

$$0 + 2t - 5t^2$$

となる。

除算の例を示す。$x(t) \div y(t)$ は、まず数学関数の計算の要領で $1/y(t)$ を計算する。逆数関数 $1/y$ の 1 ($y(t)$ の定数項) における Taylor 展開

$$1 - (y - 1) + (y - 1)^2$$

に $y(t)$ を代入し、

$$1 - (-t + t^2) + (-t + t^2)^2$$

を Type-I PSA で計算して、

$$1 + t + 0t^2$$

を得る。除算の結果はこれと $x(t)$ の積

$$(1 + 2t - 3t^2)(1 + t + 0t^2)$$

を Type-I PSA で計算して、

$$1 + 3t - t^2$$

とする。

3.3 Type-II PSA

Type-II PSA でも、Type-I PSA と同様に n 次のベキ級数

$$x(t) = x_0 + x_1 t + x_2 t^2 + \cdots + x_n t^n$$

同士の演算を行うが、$n + 1$ 次以降の高次項の情報を最高次の係数 x_n を区間にすることによって吸収する。これを実現するため、Type-II PSA を行うにはそのベキ級数の有効な定義域 (区間) D を $D = [0, d]$ のように予め定める必要がある。

Type-II のベキ級数

$$x(t) = x_0 + x_1 t + x_2 t^2 + \cdots + x_n t^n$$

は、D 上で定義された連続関数 $x^*(t)$ で、全ての $t \in D$ について

$$x^*(t) \in x_0 + x_1 t + x_2 t^2 + \cdots + x_n t^n (\quad \text{右辺は区間演算する})$$

を満たすような関数の集合を表すものとする。

加減算は次の様に定義する。

$$x(t) \pm y(t) = (x_0 \pm y_0) + (x_1 \pm y_1)t + \cdots + (x_n + y_n)t^n$$

乗算は次の手順で行われる。

(1) まず、打ち切り無しで乗算を行う。

$$\begin{aligned} x(t) \times y(t) &= z_0 + z_1 t + z_2 t^2 + \cdots + z_{2n} t^{2n} \\ z_k &= \sum_{i=\max(0,k-n)}^{\min(k,n)} x_i y_{k-i} \end{aligned}$$

(2) $2n$ 次から n 次に減次する。

減次は次のように定義する。

定義 1 (減次) ベキ級数 $x(t) = x_0 + x_1 t + \cdots + x_m t^m$ と次数 $n < m$ に対して、$x(t)$ の n 次への減次を次で定義する*:*

$$\begin{aligned} & z_0 + z_1 t + \cdots + z_n t^n \\ z_i &= x_i \quad (0 \le i \le n-1) \\ z_n &= \left\{ \sum_{i=n}^{m} x_i t^{i-n} \mid t \in D \right\} \quad \square \end{aligned}$$

このように、$n+1$ 次以降の項は n 次の項の係数に吸収するため、Type-II PSA における乗算の結果は真の乗算の結果を含む集合となる。

注意 1

$$\left\{ \sum_{i=n}^{m} x_i t^{i-n} \mid t \in D \right\}$$

の部分は、区間 D 上における多項式の値の評価である。単純に区間演算を行うのではなく、*Horner* 法で計算するとか、区間幅をなるべく狭く計算するような工夫をすると良い。

sin などの数学関数の適用は、その関数を g として、

$$\begin{aligned} & g(x_0 + x_1 t + \cdots + x_n t^n) \\ & = g(x_0) + \sum_{i=1}^{n-1} \frac{1}{i!} g^{(i)}(x_0)(x_1 t + \cdots + x_n t^n)^i \\ & + \frac{1}{n!} g^{(n)} \left(\left\{ \sum_{i=0}^{n} x_i t^i \mid t \in D \right\} \right) (x_1 t + \cdots + x_n t^n)^n \end{aligned}$$

のように g の点 x_0 での剰余項付きの Taylor 展開に代入することによって得る。但し、途中に現れる加減算や乗算は上記 Type-II PSA によって行う。

$$\left\{ \sum_{i=0}^{n} x_i t^i \mid t \in D \right\}$$

の部分の評価は、乗算の場合と同様工夫すると良い。

除算は、$x \div y = x * (1/y)$ と乗算と逆数関数に分解することによって行う。

不定積分は、

$$\int_0^t x(t) \mathrm{d}t = x_0 t + \frac{x_1}{2} t^2 + \cdots \frac{x_n}{n+1} t^{n+1}$$

のように行う。不定積分の結果は、入力された関数集合の全ての元に対して、原点で値が 0 であるような原始関数を考えると、その全てを含むように定義されている。

Type-II PSA の係数は、基本的に $n-1$ 項目までは点 (実数)、n 項目は区間となるように設計されている。浮動小数点数を用いて実装する場合はきちんと精度保証するためには $n-1$ 項目までの係数も区間にする必要がある。この場合、同じ区間であるが、$n-1$ 項目までは丸め誤差のみに由来する幅の狭い区間、n 項目は幅の広い区間になる。

3.4 Type-II PSA の例

Type-II PSA の簡単な例を示す。次数は 2 とし、定義域を $[0, 0.1]$ とする。

$$\begin{aligned} x(t) &= 1 + 2t - 3t^2 \\ y(t) &= 1 - t + t^2 \end{aligned}$$

に対して、加減算は、Type-I PSA と全く同じで、

$$\begin{aligned} x(t) + y(t) &= 2 + t - 2t^2 \\ x(t) - y(t) &= 0 + 3 - 4t^2 \end{aligned}$$

となる。乗算は、

$$\begin{aligned} x(t) \times y(t) &= 1 + t - 4t^2 + 5t^3 - 3t^4 \\ &= 1 + t + (-4 + 5t - 3t^2)t^2 \end{aligned}$$

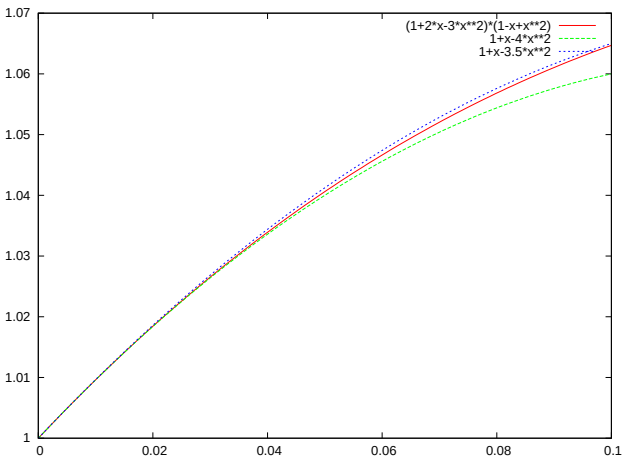
のように t^2 以降の項を t^2 で括り、括弧内を定義域 $[0, 0.1]$ で評価すると、

$$\begin{aligned} & -4 + (5 - 3 \times [0, 0.1]) \times [0, 0.1] \\ \rightarrow \quad & -4 + [4.7, 5] \times [0, 0.1] \\ \rightarrow \quad & -4 + [0, 0.5] \\ \rightarrow \quad & [-4, -3.5] \end{aligned}$$

となるので、

$$x(t) \times y(t) = 1 + t + [-4, -3.5]t^2$$

を計算結果とする。この様子を図示する。



数学関数の例を示す。$\log(x(t))$ の Taylor 展開を作るため、まず、$x(t)$ の変域を計算する。

$$\begin{aligned} & 1 + (2 - 3 \times [0, 0.1]) \times [0, 0.1] \\ \to \quad & 1 + [1.7, 2] \times [0, 0.1] \\ \to \quad & 1 + [0, 0.2] \\ \to \quad & [1, 1.2] \end{aligned}$$

これを用いて、 log の 1 ($x(t)$ の定数項) における Taylor 展開 (2 次まで) を剰余項付きで作る:

$$0 + (x - 1) - \frac{1}{2[1, 1.2]^2}(x - 1)^2$$

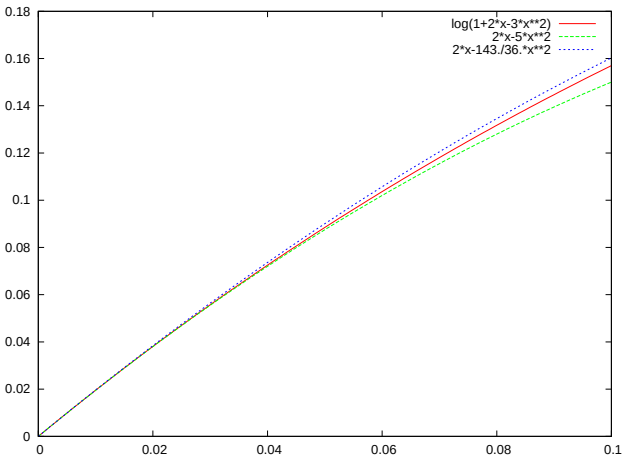
これに $x(t)$ を代入すると、

$$0 + (2t - 3t^2) - \frac{1}{2[1, 1.2]^2}(2t - 3t^2)^2$$

となるが、これを Type-II PSA で計算し、

$$0 + 2t + [-5, -\frac{143}{36}]t^2$$

となる。この様子を図示する。



除算の例を示す。$x(t) \div y(t)$ は、まず数学関数の計算の要領で $1/y(t)$ を計算する。逆数関数 $1/y$ の Taylor 展開を作るため、まず、$y(t)$ の変域を計算する。

$$\begin{aligned} & 1 + (-1 + 1 \times [0, 0.1]) \times [0, 0.1] \\ \to \quad & 1 + [-1, -0.9] \times [0, 0.1] \\ \to \quad & 1 + [-0.1, 0] \\ \to \quad & [0.9, 1] \end{aligned}$$

逆数関数 $1/y$ の 1 ($y(t)$ の定数項) における剰余項付きの Taylor 展開

$$1 - (y - 1) + \frac{1}{[0.9, 1]^3}(y - 1)^2$$

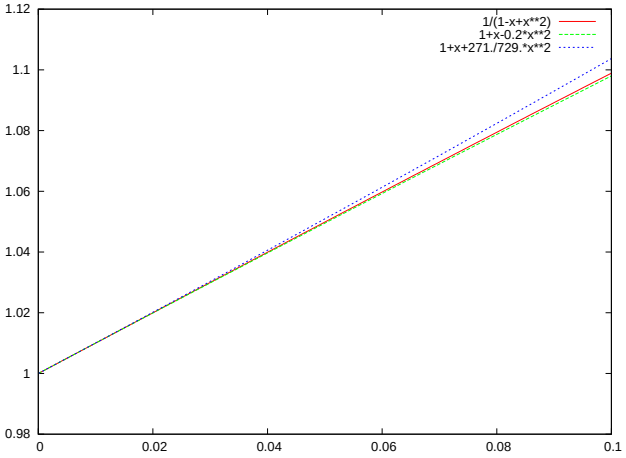
に $y(t)$ を代入し、

$$1-(-t+t^2)+\frac{1}{[0.9,1]^3}(-t+t^2)^2$$

を Type-II PSA で計算して、

$$1+t+[-0.2,\frac{271}{729}]t^2$$

を得る。この様子を図示する。



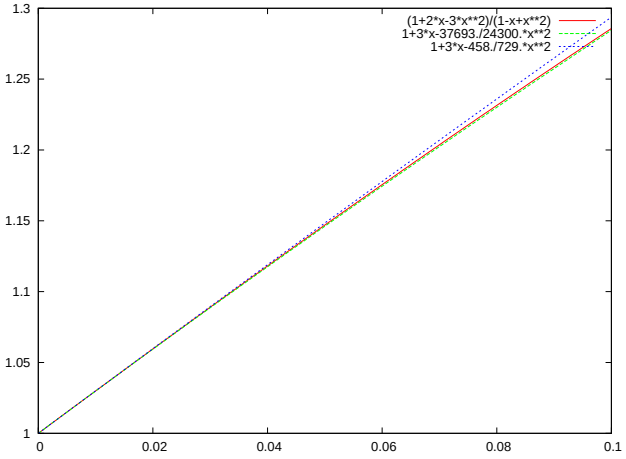
除算の結果はこれと $x(t)$ の積

$$(1+2t-3t^2)(1+t+[-0.2,\frac{271}{729}]t^2)$$

を Type-II PSA で評価して、

$$1+3t+[-\frac{37693}{24300},-\frac{458}{729}]t^2$$

となる。この様子を図示する。



4 関数の高階微分

一変数関数 $f:\mathbb{R}\to\mathbb{R}$ に対して、点 $c\in\mathbb{R}$ における f の高階微分 $f^{(k)}(c)$ を Type-I PSA を用いて計算することが出来る。f を c だけ平行移動した $f(c+x))$ を考え、

$$x(t)=0+t\quad(+0t^2+\cdots 0t^n)$$

を n 次のベキ級数形式とし、

$$y(t) = f(c + x(t))$$

を Type-I PSA で計算する。計算結果 $y(t)$ を

$$y(t) = y_0 + y_1 t + y_2 t^2 + \cdots y_n t^n$$

とすると、

$$f^{(k)}(c) = k! y_k$$

である。

例を挙げる。関数 $f(x) = \frac{1}{1+x^2}$ の点 2 における高階微分 $f'(2), f''(2), \ldots$ を計算する。次数は 3 次までとする。ベキ級数 $x(t)$ を

$$0 + t + 0t^2 + 0t^3$$

とし、$f(2 + x(t))$ を Type-I PSA で計算すると、

$$\begin{aligned}
2 + x(t) &: 2 + t + 0t^2 + 0t^3 \\
(2 + x(t))^2 &: 4 + 4t + t^2 + 0t^3 \\
1 + (2 + x(t))^2 &: 5 + 4t + t^2 + 0t^3 \\
\frac{1}{1 + (2 + x(t))^2} &: \frac{1}{5} - \frac{4}{25}t + \frac{11}{125}t^2 - \frac{24}{625}t^3
\end{aligned}$$

となる。これにより、

$$\begin{aligned}
f(2) &= \frac{1}{5} \\
f'(2) &= -\frac{4}{25} \\
f''(2) &= \frac{22}{125} \\
f^{(3)}(2) &= -\frac{144}{625}
\end{aligned}$$

が分かる。実際、手計算すると、

$$\begin{aligned}
f'(x) &= -\frac{2x}{(x^2+1)^2} \\
f''(x) &= \frac{6x^2 - 2}{(x^2+1)^3} \\
f^{(3)}(x) &= -\frac{24x(x^2-1)}{(x^2+1)^4}
\end{aligned}$$

なので、正しく計算されていることが分かる。

5 関数の値域の評価

Type-II PSA を用いて、1 変数関数 f の区間 I の像を精密に評価することが出来る。$c \in I$ とし (例えば $c = \mathrm{mid}(I)$)、f を c だけ平行移動して

$$\{f(x) | x \in I\} = \{f(c + x) | x \in I - c\}$$

を考え、これを $I-c$ を定義域とした Type-II PSA で計算すると、f の包含が得られる。すなわち、n 次のベキ級数

$$x(t) = 0 + t \quad (+0t^2 + \cdots 0t^n)$$

を考え、

$$y(t) = f(c + x(t))$$

を $I-c$ を定義域とした Type-II PSA で計算すると、$y(t)$ は f を含む区間多項式となっている。計算結果 $y(t)$ を

$$y(t) = y_0 + y_1 t + y_2 t^2 + \cdots y_n t^n$$

とすると、$f(I)$ の包含は $y(I-c)$ を計算することで得られる。$y(I-c)$ の評価は、Type-II PSA の乗算の場合と同様に工夫すると良い。

6 定積分

Type-II PSA を用いて、1 変数関数 f の区間 $[a,b]$ における定積分

$$\int_a^b f(t)dt$$

を計算できる。前節の方法と全く同様の方法で f を包含する多項式が得られるので、それを不定積分して原始関数を得、区間端の値を代入して定積分の値を得る。

(1) $c \in [a,b]$ を選ぶ。(例えば $c = \frac{a+b}{2}$)

(2) n 次のベキ級数

$$x(t) = 0 + t \quad (+0t^2 + \cdots 0t^n)$$

に対して、

$$y(t) = \int_0^t f(c + x(t))dt$$

を $[a,b]-c$ を定義域とした Type-II PSA で計算する。

(3) 計算結果 $y(t)$ を

$$y(t) = y_1 t + y_2 t^2 + \cdots y_{n+1} t^{n+1}$$

とすると、積分値は $y(b-c) - y(a-c)$ で得られる。

$c = \frac{a+b}{2}$ と選んだ場合、計算結果は $r = \frac{b-a}{2}$ として $X(r) - X(-r)$ で得られる。この場合、奇数次の項はキャンセルするので計算が省けると考えたくなるが、

(1) 係数 y_i が区間であればキャンセルしない。

(2) c が区間 $[a,b]$ の中心からわずかでもずれていればキャンセルしない。

ので、計算を省くには慎重な判断が要求される (計算を省くのはお勧めしない) 。
定積分の例を示す。

$$\int_{1.5}^{2.5} \frac{1}{1+x^2} dx$$

の計算を考える。次数を 2 とする。$c = 2$ として、

$$\int_{-0.5}^{0.5} \frac{1}{1+(2+x)^2} dx$$

と平行移動する。定義域を $[-0.5, 0.5]$ とし、

$$x(t) = 0 + t + 0t^2$$

に対して、

$$y(t) = \int_0^t f(c + x(t))dt$$

を計算すると、

$$\begin{aligned}
2 + x(t) &: 2 + t + 0t^2 \\
(2 + x(t))^2 &: 4 + 4t + t^2 \\
1 + (2 + x(t))^2 &: 5 + 4t + t^2 \\
\frac{1}{1 + (2 + x(t))^2} &: \frac{1}{5} - \frac{4}{25}t + [-\frac{5589}{609725}, \frac{31069}{33275}]t^2 \\
\int_0^t \frac{1}{1 + (2 + x(t))^2} dt &: 0 + \frac{1}{5}t - \frac{2}{25}t^2 + [-\frac{1863}{609725}, \frac{31069}{99825}]t^3
\end{aligned}$$

となる。これに対して $y(0.5) - y(-0.5)$ を計算すると、

$$[\frac{485917}{2438900}, \frac{110929}{399300}]$$

が得られる。

実際の数値計算法として用いるには、適切に積分区間を分割するなどの工夫が必要となる。

7 常微分方程式の初期値問題

7.1 Picard 型の不動点形式への変換

以下、次のような正規系の一階連立常微分方程式の初期値問題を精度保証付きで解くことを考える。

$$\begin{aligned}
\frac{dx(t)}{dt} &= f(x(t), t) && (1) \\
x(t_s) &= v && (2) \\
t &\in [t_s, t_e] && (3)
\end{aligned}$$

ここで $x(t)$ は l 次元ベクトル値関数である。

解 $x(t)$ を精度保証するため、$[t_s, t_e]$ を $[0, t_e - t_s]$ に平行移動し、両辺を積分して Picard 型の不動点形式に変換する:

$$\begin{aligned}
x(t) &= v + \int_0^t f(x(s), t + t_s)\mathrm{d}t, && (4) \\
t &\in [0, t_e - t_s]
\end{aligned}$$

X を閉区間 $[0, t_e - t_s]$ から $\mathbb{R}^l$ への連続関数全体の集合、$P : X \to X$ を (4) の右辺とする。$Y \subset X$ を閉集合とする。このとき、もし $P(Y) = \{P(x) \mid x \in Y\} \subset Y$ が成立するならば、Schauder の不動点定理により Y 内に P の不動点が存在することが保証され、それは (1) の解の存在を保証する。

7.2 解の Taylor 展開の生成

Type-I PSA と Picard 型反復を用いて、解の Taylor 展開を計算することが出来る。(4) に対して、Type-I PSA 型の変数 $X_0 = v$, $T = t$ を用いて、

(1) 次数 i の Type-I PSA で

$$X_{i+1} = v + \int_0^t f(X_i, T + t_s)\mathrm{d}t \tag{5}$$

を計算する。

(2) 次数 $i = i + 1$ とする。

を n 回繰り返すと、X_n として (4) の解の n 次の Taylor 展開が得られる。

7.3 解の精度保証

Type-II PSA と Picard 型反復を用いて、解の精度保証を行うことが出来る。Type-II PSA の定義域を $D = [0, t_s - t_e]$ と設定し、Type-I PSA の反復で得られた n 次の Taylor 近似

$$X_n = x_0 + x_1 t + x_2 t^2 + \cdots + x_n t^n$$

と $T = t$ を用いて、

(1) X_n の最終項の係数を膨らませた候補者集合

$$Y = x_0 + x_1 t + x_2 t^2 + \cdots + V t^n$$

を作成する。

(2) $v + \int_0^t f(Y, T + t_s)\mathrm{d}t$ を次数 n の Type-II PSA で計算し、$n+1$ 次から n 次に減次したものを

$$Y_1 = x_0 + x_1 t + x_2 t^2 + \cdots + V_1 t^n \tag{6}$$

とする。$n-1$ 次までの係数は X_n と全く同じになることに注意。

(3) $V_1 \subset V$ なら Y_1 内に (4) の解の存在が保証される。

更に、

(1) $v + \int_0^t f(Y_i, T + t_s)\mathrm{d}t$ を次数 n の Type-II PSA で計算し、$n+1$ 次から n 次に減次したものを $Y_{i+1} = x_0 + x_1 t + x_2 t^2 + \cdots + V_{i+1} t^n$ とする。

(2) $V_{i+1} = V_{i+1} \cap V_i$ とする。

(3) $i = i + 1$

を繰り返すことにより精度を上げることも出来る。

候補者集合の作成は、例えば次の手順で行う。

(1) $v + \int_0^t f(X_n, T + t_s)\mathrm{d}t$ を次数 n の Type-II PSA で計算し、$n+1$ 次から n 次に減次したものを $Y_0 = x_0 + x_1 t + \cdots + V_0 t^n$ とする。

(2) $r = ||V_0 - x_n||$ とし、

$$V = x_n + 2r\left([-1,1],\ldots,[-1,1]\right)^T$$

とする。

この方法は、Lohner の方法と違って大雑把な解の包含を必要としないので、ステップ幅 $t_e - t_s$ を Lohner 法より大きく取れる利点がある。

7.4 Lohner 法

Lohner の方法 [5] は、以下の通りである。自動微分法 [4] や、それと同等の前述の Type-I PSA による方法により、初期値 v を元に解 $x(t)$ の Taylor 展開を得ることが出来る。この解を、特に v の関数であることに注意して、

$$v + \alpha_1(v)t + \alpha_2(v)t^2 + \cdots + \alpha_n(v)t^n$$

と書くことにする。

次に、大雑把な解の包含を得る。$[0, t_e - t_s]$ における解 $x(t)$ を包含する候補者区間 $V \subset \mathbb{IR}^l$ を考える。

$$\begin{aligned} P(V) &\subset v + \int_0^t f(V, [0, t_e - t_s] + t_s)\mathrm{d}t \\ &\subset v + f(V, [t_s, t_e])t \\ &\subset v + f(V, [t_s, t_e])[0, t_e - t_s] \end{aligned} \tag{7}$$

により、$V_1 = v + f(V, [t_s, t_e])[0, t_e - t_s] \subset V$ が成立すれば V_1 内に $x(t)$ が包含されることが分かる。反復

$$V_{i+1} = V_i \cap (v + f(V_i, [t_s, t_e])[0, t_e - t_s])$$

で更に精度を上げることも出来る。こうして得た候補者区間 V も初期値 v に依存するため、$V(v)$ と書くことにする。

この $V(v)$ を初期値と見て再度解の Taylor 展開

$$V(v) + \alpha_1(V(v))t + \cdots + \alpha_n(V(v))t^n$$

を計算し (ただし式 (5) の t_s を $[t_s, t_e]$ と置き換える)、v を初期値として計算した結果と合わせて、

$$v + \alpha_1(v)t + \cdots + \alpha_{n-1}(v)t^{n-1} + \alpha_n(V(v))t^n \tag{8}$$

を精密な解の包含とする。最終項は Taylor 展開の Lagrange の剰余項に相当する。

候補者区間 V の作成は、例えば次のように行えばよい。

(1) $r = ||f(v, [t_s, t_e])[0, t_e - t_s]||$ とし、

(2) $V = v + 2r\left([-1,1],\ldots,[-1,1]\right)^T$ とする。

7.5 初期値問題の精度保証の例

以下、簡単な例題を、Lohner 法と PSA 法を用いて解いたものを示す。

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= -x^2 \\ x(0) &= 1, \quad t \in [0, 0.1] \end{aligned}$$

ただし、展開の次数は $n = 2$ とし区間は 10 進 3 桁程度で外側に丸めた。

7.5.1 PSA 法

Type-I PSA による Taylor 展開の生成

$$\begin{aligned} X_0 &= \boxed{1} \\ X_1 &= 1 + \int_0^t (-X_0^2)\mathrm{d}t = 1 + \int_0^t (-1)\mathrm{d}t \\ &= \boxed{1 - t} \\ X_2 &= 1 + \int_0^t (-X_1^2)\mathrm{d}t = 1 + \int_0^t (-(1-t)^2)\mathrm{d}t \\ &= 1 + \int_0^t (-(1-2t))\mathrm{d}t \\ &= \boxed{1 - t + t^2} \end{aligned}$$

候補者集合の生成

$$\begin{aligned} & 1 + \int_0^t (-X_2^2)\mathrm{d}t \\ = \; & 1 + \int_0^t (-(1 - t + t^2)^2)\mathrm{d}t \\ = \; & 1 + \int_0^t (-(1 - 2t + [2.8, 3]t^2))\mathrm{d}t \\ = \; & 1 - t + t^2 + [-1, -0.933]t^3 \end{aligned}$$

2 次に減次して、

$$Y_0 = 1 - t + [0.9, 1]t^2$$

$r = ||[0.9, 1] - 1|| = 0.1$ なので、

$$Y_0 = \boxed{1 - t + [0.8, 1.2]t^2}$$

Type-II PSA による精度保証

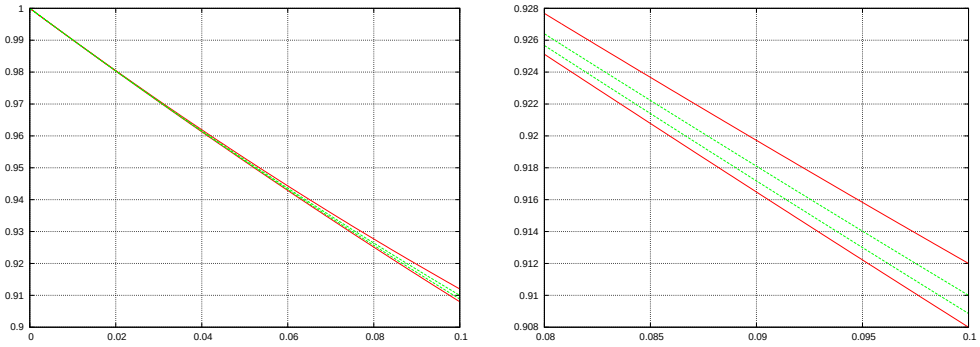
$$\begin{aligned} & 1 + \int_0^t (-Y_0^2)\mathrm{d}t \\ = \; & 1 - t + t^2 + [-1.133, -0.786]t^3 \end{aligned}$$

2 次に減次して、

$$Y_1 = \boxed{1 - t + [0.886, 1]t^2}$$

$[0.886, 1] \subset [0.8, 1.2]$ なので、Y_1 内に真の解が存在する。

この様子を図示する。



7.5.2 Lohner 法

Taylor 展開の生成 PSA 法と同じ。

$$X_2 = \boxed{1 - t + t^2}$$

大雑把な解の包含の生成

$$\begin{aligned} r &= ||(-1^2)[0, 0.1]|| = 0.1 \\ V &= 1 + 2 \times 0.1 \times [-1, 1] = \boxed{[0.8, 1.2]} \\ & \quad 1 + (-[0.8, 1.2]^2)[0, 0.1] = \boxed{[0.856, 1]} \end{aligned}$$

$[0.856, 1] \subset [0.8, 1.2]$ なので $[0.856, 1]$ 内に真の解が存在する。

解の包含の生成 初期値を $[0.856, 1]$ として Taylor 展開を行う。

$$\begin{aligned} X_0 &= [0.856, 1] \\ X_1 &= [0.856, 1] + [-1, -0.732]t \\ X_2 &= \boxed{[0.856, 1] + [-1, -0.732]t + [0.627, 1]t^2} \end{aligned}$$

初期値を 1 とした Taylor 展開と合成した、

$$\boxed{1 - t + [0.627, 1]t^2}$$

内に、真の解が存在する。

7.6 長い区間における初期値問題の精度保証

長い区間に渡って初期値問題の解を計算することを考える。以下、$t = t_s$ における値 $v = x(t_s)$ に対して、$x(t_e)$ を対応させる写像

$$\phi_{t_s,t_e} : \mathbb{R}^l \to \mathbb{R}^l, \quad \phi_{t_s,t_e} : x(t_s) \mapsto x(t_e)$$

を推進写像と呼ぶことにする。長い区間に渡る初期値問題の解は、$t_0 < t_1 < t_2 < \cdots$ に対して

$$\begin{aligned} x(t_1) &= \phi_{t_0,t_1}(x_0) \\ x(t_2) &= \phi_{t_1,t_2}(x(t_1)) \\ &\vdots \end{aligned}$$

のように計算していく。

Lohner 法で得られた (8) または PSA 法で得られた (6) に $t = t_e - t_s$ を代入すると、$\phi_{t_s,t_e}(v)$ の包含が得られる。しかし、こうすると $x(t_{i+1})$ は $x(t_i)$ に値を加算する形になり、区間幅は増大する一方となる。長い区間に渡って精度を保ったまま計算するには、推進写像の微分を利用して推進写像を書き換える方法がある。

7.6.1 推進写像の微分

推進写像の微分を得るには、$x^*(t)$ を v を初期値とした (1) の真の解として、(1) の初期値に関する変分方程式

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}y(t) &= f_x(x^*(t),t)y(t), \quad y \in \mathbb{R}^{l\times l} \\ y(t_s) &= I, \quad t \in [t_s,t_e] \end{aligned} \tag{9}$$

を考えることが基本となる (I は単位行列)。この解 $y(t)$ (これを基本解行列と呼ぶ) が求まれば、$\phi'_{t_s,t_e}(v) = y(t_e)$ である。

また、$x(t)$ と $y(t)$ を連立させて $l + l \times l$ 変数の初期値問題と考えて

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}x(t) &= f(x(t),t), \\ \frac{\mathrm{d}}{\mathrm{d}t}y(t) &= f_x(x(t),t)y(t), \\ x(t_s) &= v, \\ y(t_s) &= I, \quad t \in [t_s,t_e] \end{aligned} \tag{10}$$

を解いて、$x(t)$ と $y(t)$ を同時に求めることも出来る。

7.6.2 推進写像の書き直し

推進写像を次のように書き直す。x_i を時刻 t_i における解を含む区間、$c_i \in x_i$ を x_i の内部の点 (一般的には x_i の中心) とする。このとき、

$$\phi_{t_i,t_{i+1}}(c_i) + \phi'_{t_i,t_{i+1}}(x_i)(x - c_i) \tag{11}$$

が x_{i+1} の包含となる。この形は一般に平均値形式と呼ばれる。$\phi_{t_i,t_{i+1}}(c_i)$ は初期値を c_i として (8) または (6) を計算して $t = t_{i+1} - t_i$ とすればよい。$\phi'_{t_i,t_{i+1}}(x_i)$ は初期値を x_i として (9) または (10) に対して (8) または (6) を計算して $t = t_{i+1} - t_i$ とすればよい。なお、$\phi'_{t_i,t_{i+1}}(x_i)$ の計算は $\phi_{t_i,t_{i+1}}(x_i)$ の計算に初期値 x_i に関する自動微分 [4] を適用することによって効率よく計算することも可能である。

Lohner は、直接 $\phi'_{t_i,t_{i+1}}(x_i)$ を用いず、その $n-1$ 次までの近似を用いて計算量を削減している。具体的には、区間 $V(x_i)$ を初期値 x_i に対する (7) を満たす区間とし、(9) の $x^*(t)$ を $V(x_i)$ に置き換えたものの解 $y(t)$ の Taylor 展開を

$$I + \beta_1(I)t + \beta_2(I)t^2 + \cdots + \beta_{n-1}t^{n-1}$$

としたとき、

$$\begin{aligned} & c_i + \alpha_1(c_i)t + \cdots + \alpha_{n-1}(c_i)t^{n-1} \\ & (I + \beta_1(I)t + \cdots \beta_{n-1}(I)t^{n-1})(x - c_i) \\ & +\alpha_n(V(x_i))t^n \end{aligned} \tag{12}$$

に $t = t_{i+1} - t_i$ を代入したものを用いている。これは、ϕ_{t_s,t_e} を $n-1$ 次までの Taylor 級数で表現できる項とそれ以外に分解し、前者のみに平均値形式を用いたと考えられる。

7.6.3 解の接続

(11) または (12) を用いて長い区間に渡る初期値問題の精度保証を行う。(11),(12) は、区間行列 $A_i \in \mathbb{IR}^{l\times l}$ と区間ベクトル $B_i \in \mathbb{IR}^l$ を用いて、

$$x_{i+1} = A_i(x_i - c_i) + B_i$$

と書ける。一般に 次元 $l > 1$ の場合、この計算を単純に区間演算で行うと wrapping effect と呼ばれる問題を引き起こし、区間幅が増大してしまう。

[6] では、この計算を affine arithmetic[7] で行うと高精度に計算できることを示している。affine arithmetic を使うと計算が進むにつれて遅くなっていく問題があるが、[8] によって解消出来る。

Lohner[5] は、次のような QR 分解に基づく方法を示している。$c_0 = \mathrm{mid}(x_0)$, $y_0 = x_0 - c_0$, $Q_0 = I$, $i = 0$ とし、

(1) $c_{i+1} = \mathrm{mid}(B_i)$

(2) $y_{i+1} = (Q_{i+1}^{-1}A_iQ_i)y_i + Q_{i+1}^{-1}(B_i - c_i)$

(3) $x_{i+1} = Q_{i+1}y_{i+1} + c_{i+1}$

を繰り返す。ただし、Q_{i+1} は A_iQ_i の中心を

$$\mathrm{mid}(A_iQ_i) \simeq QR$$

のように QR 分解したものの Q とする。また、Q_{i+1}^{-1} は Q_{i+1} の真の逆行列またはそれを含む区間行列でなければならない。

8 ベキ級数演算の無駄の削減

7.2 節のアルゴリズムのように、同じ関数に対するベキ級数演算を次数を変えながら繰り返すとき、うまく工夫すると計算の無駄を大きく削減することが出来る。

例えば、

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= -x^2 \\ x(0) &= 1 \end{aligned}$$

の点 $t = 0$ における展開を得るとき、

$$\begin{aligned} X_0 &= \boxed{1} \\ X_1 &= 1 + \int_0^t (-X_0^2)\mathrm{d}t = 1 + \int_0^t (-1)\mathrm{d}t \\ &= \boxed{1 - t} \\ X_2 &= 1 + \int_0^t (-X_1^2)\mathrm{d}t = 1 + \int_0^t (-(1-t)^2)\mathrm{d}t \\ &= 1 + \int_0^t (-(1-2t))\mathrm{d}t \\ &= \boxed{1 - t + t^2} \\ X_3 &= 1 + \int_0^t (-X_2^2)\mathrm{d}t = 1 + \int_0^t (-(1-t+t^2)^2)\mathrm{d}t \\ &= 1 + \int_0^t (-(1-2t+3t^2))\mathrm{d}t \\ &= \boxed{1 - t + t^2 - t^3} \\ X_4 &= 1 + \int_0^t (-X_3^2)\mathrm{d}t = 1 + \int_0^t (-(1-t+t^2-t^3)^2)\mathrm{d}t \\ &= 1 + \int_0^t (-(1-2t+3t^2-4t^3))\mathrm{d}t \\ &= \boxed{1 - t + t^2 - t^3 + t^4} \end{aligned}$$

のように計算が進むが、最後に新しい項が加わるのみで低次の項の係数は変わらないことが分かる。このことを利用して、常微分方程式の右辺の評価時のベキ級数演算の全ての中間変数を保存しておき、加算、減算、乗算時に既に計算されている低次の項は計算せずに前回の中間変数からコピーすれば、同じ値を何度も計算する無駄を省くことが出来る。

数学関数や除算は加算、減算、乗算の組み合わせに展開されるので、これらも問題なく無駄を削減できる。但し、Taylor 展開の展開項数が増えているであろうから、これに注意する必要がある。

参考文献

[1] Masahide Kashiwagi and Shin'ichi Oishi : "Numerical Validation for Ordinary Differential Equations — Iterative Method by Power Series Arithmetic —", Proc. 1994 Symposium on Nonlinear Theory and its Applications (NOLTA'94 Symposium), pp.243–246 (1994.10.7).

[2] 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).

[3] 柏木 雅英: “常微分方程式の精度保証”, 日本応用数理学会監修 応用数理ハンドブック 朝倉書店, pp.442–445 (2013).

[4] 久保田 光一, 伊理 正夫 : “アルゴリズムの自動微分と応用”, コロナ社 (1998).

[5] 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).

[6] 柏木啓一郎, 柏木雅英, “平均値形式とアフィン演算を用いた常微分方程式の精度保証法”, 日本応用数理学会論文誌, Vol. 21, No. 1, pp.37–58 (2011).

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

[8] Masahide Kashiwagi, “An algorithm to reduce the number of dummy variables in affine arithmetic”, 15 ℔h GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic and Verified Numerical Computations (SCAN2012).