端点特異性を持つ関数の精度保証付き数値積分

柏木 雅英 [1]

[1] 早稲田大学基幹理工学部応用数理学科

e-mail : kashi@waseda.jp

1 はじめに

ベキ級数演算は常微分方程式の精度保証付き数値計算を行うために開発したものであるが [1]、これを使うと精度保証付きの数値積分を容易に行うことが出来る。また、ベキ級数の「約分」をうまく使うと、特異点を持った数値積分を精度保証付きで行うことが出来る。本報告では、積分区間の端点について、ベキ型特異点、log 型特異点、除去可能特異点を持った数値積分を精度保証付きで行う方法を示す。

2 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。紙面の都合上 Type-II の演算のみを説明する。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。

加減算は次の様に定義する。

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

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。

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

(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:

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

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。この計算中に現れる加算や乗算は Type-II PSA で行う。

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。

3 精度保証付き数値積分

積分区間内に特異点を持たない数値積分の方法を示す。区間 $[x_i, x_i + \Delta t]$ における積分

$$\int_{x_i}^{x_i + \Delta t} f(t) dt$$

を次のように計算する。

(1) n 次のベキ級数

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

に対して、

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

を $[0, \Delta t]$ を定義域とした Type-II PSA で計算する。

(2) 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 $y(\Delta t)$ で得られる。

ステップ幅は、n を展開の次数、ε を machine epsilon として、Taylor 展開の係数を元に

$$\Delta t = \frac{\varepsilon^{\frac{1}{n}}}{\max(|x_{n-1}|^{\frac{1}{n-1}}, |x_n|^{\frac{1}{n}})}$$

のように計算し修正したものを用いている (詳細略)。

4 端点特異性を持つ数値積分の精度保証

簡単のため、積分区間の左端に特異点がある場合で説明する。

4.1 除去可能特異点

例えば $\int_0^1 \frac{\sin(t)}{t} dt$ のように、端点で $\frac{0}{0}$ が現れるケース。この場合、分母分子を psa 型で評価すると、

$$\frac{y_0 + y_1 t + \cdots + y_n t^n}{x_0 + x_1 t + \cdots + x_n t^n}$$

の x_0, y_0 がともに 0 になっているはず。これを約分し、

$$\frac{y_1 + y_2 t + \cdots + y_n t^{n-1}}{x_1 + x_2 t + \cdots + x_n t^{n-1}}$$

とすれば、$x_1 \neq 0$ なら計算を続行できる。

4.2 ベキ型特異点

例えば $\int_0^1 \sqrt{\sin(t)} \cos(t) dt$ のような、$f(t)^\alpha g(t)$ (ただし $f(t)$ は端点で 0、α は非整数) の形。n を、端点で $f^{(n)}(t) \neq 0$ となるような最小の n とする。

$$f(t)^\alpha g(t) = \left(t^n \frac{f(t)}{t^n}\right)^\alpha g(t) = t^{n\alpha} g(t) \left(\frac{f(t)}{t^n}\right)^\alpha$$

と変形し、分数の中を約分処理する。これにより、

$$g(t) \left(\frac{f(t)}{t^n}\right)^\alpha = y_0 + y_1 t + \cdots + y_n t^n$$

と得られたならば、

$$y_0 t^{n\alpha} + y_1 t^{1+n\alpha} + \cdots + y_n t^{n+n\alpha}$$

という多項式に対して積分を実行する。

4.3 log 型特異点

例えば $\int_0^1 \log(\sin(t)) \cos(t) dt$ のような、$\log(f(t)) g(t)$ (ただし $f(t)$ は端点で 0) の形。n を、端点で $f^{(n)}(t) \neq 0$ となるような最小の n とする。

$$\begin{aligned} \log(f(t)) g(t) &= \log\left(t^n \frac{f(t)}{t^n}\right) g(t) \\ &= n g(t) \log t + \left(\log \frac{f(t)}{t^n}\right) g(t) \end{aligned}$$

と変形し、分数の中を約分処理する。このとき、後半の項は特異性が無いので通常の方法で、前半の項は

$$n g(t) = y_0 + y_1 t + \cdots + y_n t^n$$

と得られたならば、

$$y_0 \log t + y_1 t \log t + \cdots + y_n t^n \log t$$

に対して

$$\int_0^t t^n \log t dt = \frac{t^{n+1}((n+1)\log(t) - 1)}{(n+1)^2}$$

を用いて項別積分を行う。

5 数値例

例として挙げた $\int_0^1 \frac{\sin(t)}{t} dt$、$\int_0^1 \sqrt{\sin(t)} \cos(t) dt$、$\int_0^1 \log(\sin(t)) \cos(t) dt$ は、それぞれ

$$\begin{aligned} &[0.94608307036684913, 0.94608307036909978] \\ &[0.51459724773239412, 0.51459724773239902] \\ &[-0.98671202916248546, -0.98671202916247868] \end{aligned}$$

と包含された。使用した級数の次数は 12 次。

本アルゴリズムの実装例を含む精度保証付き数値計算のためのライブラリ集を [2] で公開しており、この計算は kv-0.4.22 で実行した。

参考文献

[1] 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).

[2] kv – C++による精度保証付き数値計算ライブラリ, `http://verifiedby.me/kv/`.