A-007

区間演算を用いたODE Solverにおける任意精度演算の導入とパラメタ最適化

Parameter Optimization of an Interval based ODE solver with Multiple Precision Calculation

廣瀬 賢一 † 石井 大輔 ‡ 上田 和紀 ‡
Kenichi Hirose Daisuke Ishii Kazunori Ueda

1 はじめに

常微分方程式の初期値問題 (以下 IVP-ODE) を数値計算を用いて解くことは分子生物学、制御工学、物理学を初めとする様々な分野において重要視される。

IVP-ODE を解く手段として、近似解法であるオイラー法やルンゲクッタ法などを用いるが、これらはあくまでも近似計算でしかなく、得られる解の精度が保証できないという問題が存在する。

そこで、区間演算 [2] を用いて IVP-ODE を解くことで解の精度を保証するという研究がおこなわれている [1, 2, 10] が、区間演算を用いて解くと解の区間が爆発しがちである。

その主な理由として 2 つほど挙げることが出来る。

まず、区間演算による誤差の累積である。区間演算では毎回計算を行うたびに精度保証を行うために区間を広げていく必要がある。その為、計算量によっては容易に区間が爆発してしまうという問題を抱えている。IVP-ODE を解くにあたって、基本的には積分区間を細かく分割して解いていくという方針を採るために、演算回数は多くなりがちであり常に区間爆発の危険性を伴っている。

次に、Wrapping Effect という問題が挙げられる。Wrapping Effect は多次元の ODE を積分する際に起こる現象であり、解を各変数の区間の直積として表すことに起因する。変数の値の変化が大きい問題においてこの問題が起こりやすい傾向がある。

本研究では、区間演算を用いた ODE Solver である VNODE-LP[1] を任意精度演算が可能なように拡張し、それを元に先ほど挙げた問題点の一つ目である区間演算自身の問題点を区間演算に使用する浮動小数点数の仮数部のビット数を上げることにより改善することで、どの程度精度を向上させることが出来るのかを調べた。そして、仮数部のビット数を上げる事による精度への影響の度合によって ODE を 3 パターンに分類し、それぞれのパターンにおける VNODE-LP の最適なパラメタの関係を導き出した。

†早稲田大学大学院 基幹理工学研究科 情報理工学専攻
‡早稲田大学理工学術院

2 区間演算

得られる解の精度を保障する精度保証付き数値計算のための手法として区間演算がある。区間演算では、真解を包囲することを保証する区間について計算をおこなう。

区間を以下のように定義する。

$$\mathbb{IR} = \{x = [\underline{x}, \overline{x}] = \{x \subseteq \mathbb{R} \mid \underline{x} \leq x \leq \overline{x}, \quad \underline{x}, \overline{x} \in \mathbb{R}\}\}$$

ここで、$\underline{x} \leq \overline{x}$ であり、$\underline{x}$ を下限、$\overline{x}$ を上限と呼ぶ。また、上限と下限の差を区間幅と呼ぶ。

2.1 任意精度浮動小数点数区間演算

区間爆発を抑える手段の一つとして区間演算に使用する浮動小数点数の仮数部のビット数（以下、単にビット数と呼ぶ）を上げる手法がある。

本研究においては、GMP[4]、MPFR[5]、MPFRCPP[6]、Boost Interval Arithmetic Library[9] の 4 つのライブラリを使用し任意のビット数を用いた区間演算をおこなった。

計算誤差が極端に累積する例として Rump[3] による以下の式がある。ただし、$a = 77617.0$、$b = 33096.0$ とする。

$$(333.75 - a^2)b^6 + (11a^2b^2 - 121b^4 - 2)a^2 + 5.5b^8 + a/(2b) \qquad (1)$$

この式の真解は $-0.8273960599468213\cdots$ であるが、32-bit 浮動小数点演算 (仮数部 24-bit) による解は $1.172604\cdots$、64-bit 浮動小数点演算 (仮数部 53-bit) による解は $1.1726039400531786\cdots$ となり、浮動小数点演算による解は誤った物となる。

区間演算を用いることにより真解を包囲する区間が得られるが、仮数部 53-bit の浮動小数点を用いた区間演算では区間幅が 7.0835497243e+21 と非常に大きなものとなってしまう。しかし、仮数部を 122-bit とすると区間幅は 3.7615819226e-37 となり、良い精度の解を得ることができる。この様に区間幅が大きくなってしまうような問題でもビット数を上げることで区間幅の増大を抑えることが可能となる。

また、ビット数と得られる精度の関係を調べた所、数学関数や四則演算に関しては基本的に 1 ビット増やす毎に精度は 2 倍となった。

3 常微分方程式

常微分方程式 (以下、ODE) を解くにあたって数値計算では n 階の ODE を一階の ODE に変換し、それを元に計算を行うこととなる。

また、ODE には硬い問題 (stiff problem) と硬くない問題 (non-stiff problem) が存在し、一般に硬い ODE は解くのが困難とされる。

3.1 一階常微分方程式

一階の常微分方程式は式 (2) の様に定義できる。

$$\begin{cases} y_1' = f_1(t, y_1, \cdots, y_n) \\ y_2' = f_2(t, y_1, \cdots, y_n) \\ \vdots \\ y_n' = f_n(t, y_1, \cdots, y_n) \end{cases} \tag{2}$$

通常はこれをベクトルとしてまとめ、$y' = f(t, y)$ とする。ここで、関数 f の中に t が存在しないものを自励系と呼び、$y' = f(y)$ と書くことが出来る。

また、初期時間 t_0 と、その時点での解 $y(t_0)$ が与えられたとき、ある時間 t における解 $y(t)$ を求める問題を常微分方程式の初期値問題 (以下、IVP-ODE) と呼ぶ。

4 VNODE-LP

VNODE-LP[1] は N. S. Nedialkov によって作成された区間演算を用いた ODE Solver であり、テイラー展開をベースとした一段階法と Hermite-Obreschkoff 法を使用している。

また、精度に影響するパラメタとしてテイラー展開をおこなう次数 (order)、絶対許容誤差 (atol)、相対許容誤差 (rtol) がある。atol と rtol は 1 ステップ毎の真解からのずれを指定するパラメタであり、現在の解の大きさを $\|y\|$ としたとき、1 ステップ後の解の大きさはおおよそ $\text{atol} + \|y\|_\infty * \text{rtol}$ に比例することとなる。

4.1 任意精度演算の導入

VNODE-LP では区間演算ライブラリとして FILIB++[8] または PROFIL/BIAS[7] を使用できるようになっている。今回は 2.1 章で使用したライブラリを組み込む事で、任意のビット数を用いた区間演算による求解を可能にし、ビット数を上げることによる解の精度への影響を調べる実験を行った。結果、ビット数を上げることで ODE によっては非常に効率よく区間幅を削減することに成功した一方で、ほとんどビット数の影響を受けない ODE も存在した。

また、効率よく精度を上げるにはビット数と共に VNODE-LP のパラメタである order、atol、rtol の 3 つを上手く調整する必要があった。

なお、実験にはグリッドコンピューティング環境である InTrigger[12] の chiba 拠点と hongo 拠点を使用した。CPU は Pentium M 1.8GHz、メモリは 1GBytes であり、実行時間は約 32700CPU 時間である。

4.2 ビット数と order の関係について

ビット数と order がどのように互いに関係しあうかについて実験をおこなった。

その結果、今回実験に使用した ODE は 3 つのタイプに分けることが出来るということがわかった。

グラフにおいて縦軸がビット数であり、横軸が order である。

4.2.1 ほぼビット数のみに精度が依存するパターン

以下のような ODE においては精度は order にほとんど依存せずに、ビット数に比例する形で上昇していった。

図 1 は「自由落下の式」のグラフであり、単純にビット数を上げていけば精度が上がっていくことが分かる。

このようになる理由として、これらの ODE は有限回の微分によって値が 0 になる為に、ある程度高次のテイラー展開を用いることで完全な近似が可能となるためであるという事が考えられる。

・自由落下の式

$$\begin{cases} y_0' = y_1 \\ y_1' = -10 \end{cases}$$

・10 次微分の式

$$\begin{cases} y_i' = y_{i+1} \quad [i = 0..9] \\ y_{10}' = 2 \end{cases}$$

4.2.2 ビット数と order の比率が精度に影響するパターン

以下のような ODE においては、ビット数と order の値を 1:1 から 2:1 程度の間の割合で変化させると良い精度が得られるという結果となった。

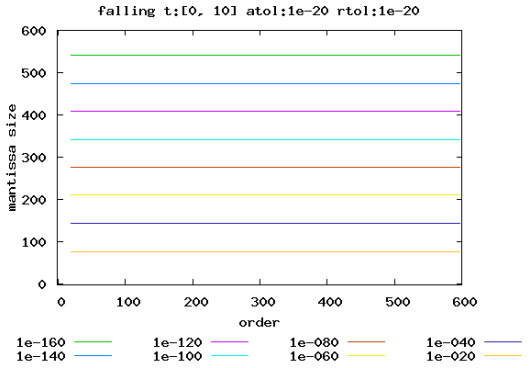

図 1: ほぼビット数のみに精度が依存するパターン

図 2 は「e に関する式 2」のグラフであり、このグラフにおいてはビット数と order を約 3:2 の割合で変化させると良いということが分かる。

・円運動の式

$$\begin{cases} y_0' = y_1 \\ y_1' = -y_0 \end{cases}$$

・e に関する式 1

$$y_0' = -y_0$$

・e に関する式 2

$$\begin{cases} y_0' = y_0 - 2y_1 \\ y_1' = 3y_0 - 4y_1 \end{cases}$$

・文献 [10] の式 (stiff)

$$y_0' = -10(y_0 - \sin(t)) + \cos(t)$$

・文献 [11] の a2 問題の式 (stiff)

$$\begin{cases} y_0' = -1800y_0 + 900y_1 \\ y_i' = y_{i-1} - 2y_i + y_{i+1} \quad [i = 1..7] \\ y_8' = 1000y_7 - 2000y_8 + 1000 \end{cases}$$

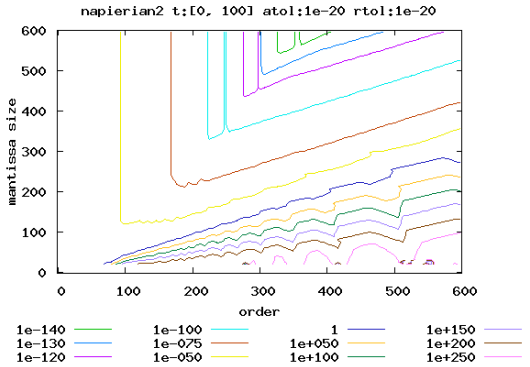

図 2: ビット数と order の比率が精度に影響するパターン

4.2.3 ビット数を上げても精度がほとんど向上しないパターン

以下のような ODE においては、order の最適値は非常に若い所にあり、場合によっては VNODE-LP で指定可能な最小値である'3' が最小値の場合もあった。ビット数を上げることによる精度上昇は最初のうちしか見られないが、atol や rtol の値を下げることで精度上昇が頭打ちになってしまうビット数の値を上げる事は出来た。

図 3 は Rössler 方程式のグラフであり、このグラフにおいては order の値は 3 の時が最適値となった。

・Logistic 方程式

$$y_0' = y_0(1 - y_0)$$

・振り子の方程式

$$\begin{cases} y_0' = y_1 \\ y_1' = -\sin(y_0) \end{cases}$$

・Van der Pol 方程式 (stiff)

$$\begin{cases} y_0' = y_1 \\ y_1' = \mu(1 - y_0^2)y_1 - y_0 \end{cases}$$

・Rössler 方程式 (chaos)

$$\begin{cases} y_0' = -y_1 - y_2 \\ y_1' = y_0 + 0.2y_1 \\ y_2' = 0.2 + y_2(y_0 - \mu) \end{cases}$$

・Lorenz 方程式 (chaos)

$$\begin{cases} y_0' = 10(y_1 - y_0) \\ y_1' = y_0(28 - y_2) - y_1 \\ y_2' = y_0y_1 - 8/3y_2 \end{cases}$$

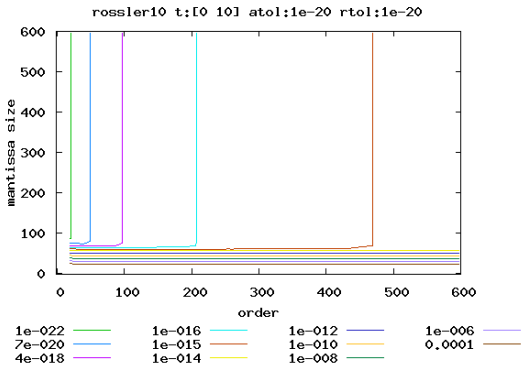

図 3: ビット数を上げても精度がほとんど向上しないパターン

4.3 atol と rtol の関係について

order とビット数を様々な値に固定し atol と rtol を変化させることでどの様に精度が変化するかについての実験をおこなった。

その結果、図 4 の様に atol と rtol は基本的には 1:1 の割合で変化させれば良いということがわかり、値を小さくすればするほど精度は向上するという事も判明した。しかし、下げすぎると逆に精度は下がってしまうという現象が起こった。これは、atol と rtol の値を小さくすると基本的にステップ幅が狭まる為に計算量が増大し、逆に区間が広がってしまう為であると考えられる。その為、この現象はビット数を上げる事で防ぐことが出来た。

4.4 複数のパターンを混ぜ合わせた場合

異なったパターンに分類される IVP-ODE を同時に解くと、どの様になるかについての実験を行った。

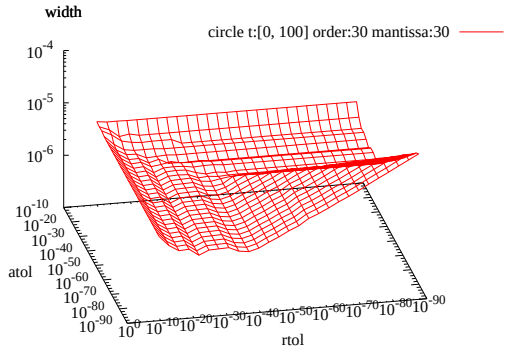

図 4: atol と rtol の関係について

例えば、式 (3) は自由落下の式、円運動の式、Logistic 方程式の 3 つを混ぜ合わせた物であるが、y_0, y_1 は元の自由落下の式によるグラフとほぼ同じ物となり、y_2, y_3 は元の円運動の式によるグラフとほぼ同じ物となり、y_4 は元の Logistic 方程式によるグラフとほぼ同じものとなると言う様に、元の混ぜ合わせる前のグラフとほとんど違いが無い物となった。

他の組合せに関しても同じような結果となった。

$$\begin{cases} y_0' = y_1 \\ y_1' = -10 \\ y_2' = y_3 \\ y_3' = -y_2 \\ y_4' = y_4(1 - y_4) \end{cases} \tag{3}$$

4.5 ODE 内の変数の違いに対する最適なパラメータの変化

例えば Lorenz 方程式内の変数 y_0, y_1, y_2 の様に、同一の ODE 内の変数によって最適なパラメタの関係が異なってくるかどうかについて調べた。その結果、どの変数のグラフもほぼ同じ物となった。これは、同じ ODE 内の変数は互いに参照しあっているためであると思われる。むろん 4.4 章のように互いに独立している変数が存在する場合には同一のグラフとはならない。

5 まとめと今後の課題

本研究により、区間演算は使用する浮動小数点の仮数部のビット数を上げることで、区間爆発を防ぐことが出来るという事が確認できた。

次に、IVP-ODE においてもビット数を上げる戦略は有効ではあるが、まったく意味のない問題も存在するという事が分かり、ビット数を上げることによる影響の仕方によって ODE を 3 パターンに分類することが出来た。3 パターンの内「ビット数と order の比率が精度に影響するパターン」と「ビット数を上げても精度がほとんど向上しないパターン」の 2 つの違いが出る原因は不明であるが、硬い ODE でもビット数を上げる戦略が有効な物も存在した為に、硬さはあまり関係が無い可能性もある。ただし、カオスの式といわれるような難しい式は全てビット数を上げてもあまり意味が無いという結果となった。

そして、ビット数を上げる戦略が有効でない IVP-ODE における区間爆発が起こってしまう原因は区間演算自体の精度ではなく、例えば Wrapping Effect の様な他の要素が原因であるということが言える。

今後はそれぞれのパターンに分かれる条件を調べ、ODE の式を見るだけで判別する方法を調べあげる必要性があると共に、ビット数を上げる以外の手法により精度の良い解を得る方法を調べる必要性が存在する。

参考文献

[1] N. S. Nedialkov. VNODE-LP: A Validated Solver for Initial Value Problems in Ordinary Differential Equations, Technical Report CAS-06-06-NN, Department of Computing and Software, McMaster University, July 2006

[2] R. E. Moore, Interval Analysis, Prentice-Hall, 1966

[3] S.M. Rump. Algorithms for Verified Inclusions - Theory and Practice. In R.E. Moore, editor, Reliability in Computing, volume 19 of Perspectives in Computing, pp 109–126, Academic Press, 1988

[4] The GNU MP Bignum Library
http://gmplib.org/

[5] The MPFR Library
http://www.mpfr.org/

[6] Alexey V. Beshenov, MPFRCPP // The Multiple Precision Floating-Point Reliable Library C++ Interface
http://beshenov.ru/mpfrcpp/

[7] O. Knüppel, PROFIL/BIAS–A Fast Interval Library, Computing 53, pp 277–287, 1994

[8] M. Lerch, G. Tischler, J. Wolff von Gudenberg, W. Hofschuster, W. Krämer, The Interval Library filib++ 2.0 Design, Features and Sample Programs, Wissenschaftliches Rechnen/Softwaretechnologie, April 2001

[9] Hervé Bönnimann, Guillaume Melquiond, Sylvain Pion, The design of the Boost interval arithmetic library, Theoretical Computer Science 351, pp 111-118, February 2006

[10] Yves Deville, Micha Janssen, Pascal Van Hentenryck, Consistency Techniques in Ordinary Differential Equations, Technical Report TR98-06, Universit'e catholique de Louvain, July 1998

[11] W. H. Enright, T. E. Hull, B. Lindberg, Comparing numerical methods for stiff systems of O.D.E:s, BIT Numerical Mathematics, Volume 15, Number 1, pp 10–48, March 1975

[12] InTrigger プラットフォーム
https://www.logos.ic.i.u-tokyo.ac.jp/intrigger/