数式処理 *J.JSSAC (2005)*
Vol. 12, No. 1, pp. 23 - 31

特集論文

区間演算と数式処理の歴史

近藤 祐史*

詫間電波工業高等専門学校

(Received 2005/07/11 Revised 2005/08/10)

概　要

It becomes the powerful computation tools to combine the interval arithmetic with computer algebra system. Then, we developed to add the interval arithmetic in computer algebra system Risa/Asir. Now, the interval arithmetic is available in other systems such Maple and Mathematica. In this paper, a part of the interval arithmetic in Risa/Asir is explained.

1 はじめに

筆者らは 1992 年に数式処理システム Risa/Asir に区間演算機能を組み込んだことを発表した [6]．発表当時，数値・数式融合計算 [13] が提唱され数式処理と数値計算との結合の重要性がますます増してきていた．融合計算 (ハイブリッド計算) を実現する計算システムは数式処理システムを基本とし数値計算機能を付加する必要があると考えた．しかし，ここで重要なことは浮動小数演算にともなう数値計算の誤差の問題である．融合計算の目的の一つは悪条件問題など誤差に敏感な問題を安定して解くことにある．それには計算途中の数値計算の誤差の取り扱いはより厳密に行なわねばならない．そこで融合計算で用いる浮動小数演算を区間演算に基づく精度保証計算 [1][11][15] に置き直せばより強力なシステムが完成すると期待し得る．そのため，区間演算と数式処理システムを結合し，融合計算を効果的に実現する計算システムを構成することを考えた．数式処理システムと区間演算を結合する場合には，区間演算用のソフトウェアパッケージを取り込み，数式処理と結果との結合をはかる方法と新しい計機システムを作成する方法とが考えられる．前者の例はパソコンで稼働する小型ハイブリッド処理システム SYNC で実験されている [12]．しかし，この方法は区間演算パッケージの言語 (PASCAL–SC[9]) の機能に強く依存しており，汎用性と計算速度の面で検討すべき点が多い．そこで，第 2 の方法として新しい計算機システムの作成を考える．新しい計算システムは既存の数式処理システムで区間演算を実現する．区間演算を実現するためにはデータ構造に区間数を取り入れる必要がある．そこで，国産で

*kondoh@dc.takuma-ct.ac.jp

ソースコードの入手可能な数式処理システムとして，当時富士通研究所で開発されていた数式処理システム Risa/Asir[14] を用いることにした．

本システムと同様に数式処理システムで区間演算が利用可能なシステムとしては，次のようなものがある．

- REDUCE では，J.H.J. Molenkamp らによるパッケージ [10] がある．
- Maple では，R.M. Corless らによる区間演算パッケージ INTPAK[3] が開発されている．これは，Maple の多倍精度浮動小数点数 (bigfloat) を区間の上限下限に利用し，区間数をリストとして表現したものである．その後，Maple 7 では区間演算機能が組み込まれている．
- Mathematica では，J.B. Keiper により組み込まれている [5]．区間の上下限を Machine precision(通常の倍精度浮動小数点数) と arbitary precision floating point (多倍精度浮動小数点数) で実現した実装が行われており，データ構造として区間数を持つ．

2 区間演算の概要

ここでは本システムで組み込む区間演算の概要について述べる．

実数全体を$\boldsymbol{R}$であらわし，

$$A = \{x | \underline{a} \le x \le \overline{a}\} \qquad x, \underline{a}, \overline{a} \in \boldsymbol{R}$$

なる A を区間数と呼び，$A = [\underline{a}, \overline{a}]$ と表す．ただし，$\underline{a} \le \overline{a}$ とする．$\underline{a}, \overline{a}$ それぞれを区間数の下限，上限と呼ぶ．また，2 つの区間数 $A = [\underline{a}, \overline{a}]$, $B = [\underline{b}, \overline{b}]$ の間の演算を次のように定義する．ここで，英大文字は区間数，英小文字は実数を表す．

$$\left\{\begin{array}{lcl} A + B & = & [\underline{a}, \overline{a}] + [\underline{b}, \overline{b}] = [\underline{a} + \underline{b}, \overline{a} + \overline{b}], \\ A - B & = & [\underline{a}, \overline{a}] - [\underline{b}, \overline{b}] = [\underline{a} - \overline{b}, \overline{a} - \underline{b}], \\ A \cdot B & = & [\underline{a}, \overline{a}] \cdot [\underline{b}, \overline{b}] = [\min(\underline{ab}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b}), \max(\underline{ab}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b})], \\ A \,/\, B & = & [\underline{a}, \overline{a}] \,/\, [\underline{b}, \overline{b}] = [\underline{a}, \overline{a}] \cdot [1/\overline{b}, 1/\underline{b}] \\ & = & [\min(\underline{a}/\overline{b}, \underline{a}/\underline{b}, \overline{a}/\overline{b}, \overline{a}/\underline{b}), \max(\underline{a}/\overline{b}, \underline{a}/\underline{b}, \overline{a}/\overline{b}, \overline{a}/\underline{b})], \\ & & (\text{ただし，}\ 0 \notin B). \end{array}\right.$$

実際に計算機で計算する場合には，IEEE 標準 754 の丸めモードを用いて結果の区間の下限は切捨て，上限は切り上げる．そのため，計算機で実現できる区間演算の結果は正確な計算と異なる．これを機械区間演算という．以下，この機械区間演算を単に区間演算ということにする．

上の区間数の加法と乗法に関してはともに結合法則と交換法則を満足する．すなわち，A, B, C を区間数とすると，

$$A + (B + C) = (A + B) + C$$
$$A \cdot (B \cdot C) = (A \cdot B) \cdot C$$
$$A + B = B + A$$
$$A \cdot B = B \cdot A$$

が成立する．しかし，区間演算の場合分配法則は必ずしも成立しない．たとえば，

$$[1,2]\cdot([1,2]-[1,2]) = [1,2]\cdot[-1,1] = [-2,2]$$

であるが，

$$[1,2]\cdot[1,2]-[1,2]\cdot[1,2] = [1,4]-[1,4] = [-3,3]$$

となる．分配法則の代わりに，

$$A\cdot(B+C)\subseteq A\cdot B+A\cdot C$$

が成立する．また，加法と乗法の逆元は存在しない．

3 区間数を結合した数式処理システム

文献 [6] で実現したシステムの概要を述べる．

3.1 データ構造の構築

数式処理システムではデータ構造は数値計算のデータと異なりデータ自身にその型を表すフラグを持つため，データ構造どのように構築するかが大変重要である．ここでは，演算の高速化や多項式等の係数に区間数を使用できるように，また Risa/Asir のコードの変更が少なくなるようにデータ構造を構築する．

Risa/Asir のデータ構造は階層的に図 1 のようになっている．ここに区間数のデータ構造を付

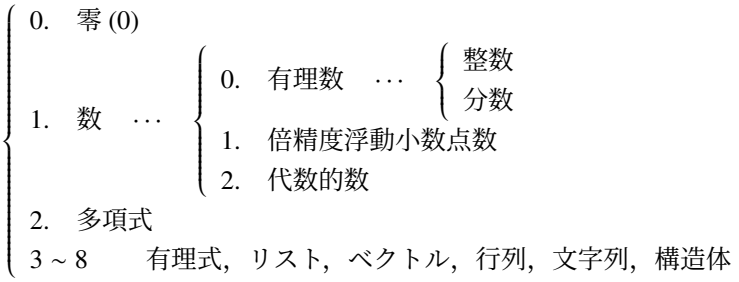

図 1: Risa/Asir のデータ構造（開発当時）

加する場合，

- 多項式などと同層に入れるか，
- 有理数や倍精度浮動小数点数と同層（数オブジェクト）に入れるか，

が問題である．ここでは，多項式等の係数に区間数を使用可能に，また Risa/Asir のコードの変更が少なくなるように有理数や倍精度浮動小数点数などの数オブジェクトの層に入れ，その末尾に区間数を加えた．

また実際のデータの格納方法は，bit 長が少なく計算の高速化を目指して，区間の下限，上限に数値計算と同様の倍精度浮動小数点数を利用する方法を採用した．これにより区間数は図 2 のようになる．

図 2: 区間数の格納方法

3.2 組み込み関数

区間数に関する以下の関数を作成し組み込んでいる.

- システムからの区間数 $A = [\underline{a}, \overline{a}]$ の入力: `intval(`$\underline{a}$`,`$\overline{a}$`) (= itv(`$\underline{a}$`,`$\overline{a}$`))`.
- 区間数 A の中心: `mid(`A`)` $= (\underline{a} + \overline{a})/2$.
- 区間数 A の幅: `width(`A`)` $= \overline{a} - \underline{a}$.
- 区間数 A の絶対値: `absintval(`A`)` $= \max(|\underline{a}|, |\overline{a}|)$.
- 区間数 A, B の間の距離: `distance(`A`,`B`)` $= \max(|\underline{a} - \underline{b}|, |\overline{a} - \overline{b}|)$.
- 区間数 A の下限の取り出し: `inf(`A`)` $= \underline{a}$.
- 区間数 A の上限の取り出し: `sup(`A`)` $= \overline{a}$.
- 区間数 A, B の間の cap: `cap(`A`,`B`)` $= [\max(\underline{a}, \underline{b}), \min(\overline{a}, \overline{b})]$, (ただし，結果が ϕ でない場合).
- 区間数 A, B の間の cup: `cup(`A`,`B`)` $= [\min(\underline{a}, \underline{b}), \max(\overline{a}, \overline{b})]$.
- 実数 a, 区間数 A: *if* $a \in A$ *then* 1 *else* 0: `inintval(`A`,`a`)`.

これらは通常の組み込み関数と同様にプログラム中で使用できる.

3.3 実行例

開発したシステムでの実行例の幾つかを以下に示す.

- 区間数を係数に持つ多項式の和と積

```
[0] P=intval(1,2)*x^2+intval(2,3);
[1,2]*x^2+[2,3]
[1] Q=intval(2,3)*x+intval(0,1);
[2,3]*x+[0,1]
[2] P+Q;
[1,2]*x^2+[2,3]*x+[2,4]
[3] P*Q;
[2,6]*x^3+[0,2]*x^2+[4,9]*x+[0,3]
```

- 記号と区間数を係数に持つ多項式の和と積

```
[0] P=intval(1,2)*x^2+a*x+intval(2,3);
[1,2]*x^2+a*x+[2,3]
[1] Q=b*x+intval(0,1);
```

```
b*x+[0,1]
[2] P+Q;
[1,2]*x^2+(a+b)*x+[2,4]
[3] P*Q;
[1,2]*b*x^3+(b*a+[0,2])*x^2+([0,1]*a+[2,3]*b)*x+[0,3]
```

このように，区間数を係数に持つ多項式の処理を簡単に行なうことができる．

4 アルゴリズムの安定化技術

白柳らは 1995 年にアルゴリズムの安定化理論を発表した [21][18]．安定化理論は，正確な計算を前提としているアルゴリズムに近似を導入した時に起こる不安定性を克服する．そのキーポイントは，

(1) アルゴリズムの構造は変えない．
(2) データ領域において普通の係数をブラケット係数に変える．
(3) 条件文等の述語の直前においてブラケット係数の書換え（ゼロ書換え）を行なう．

にある．これにより得られた安定化アルゴリズムを近似計算で実行する時，精度を高めて計算を繰り返せば，元のアルゴリズムの真の出力に収束する．精度を高めながら再計算するには，任意多倍長浮動小数点数 (bigfloat) を用いる必要がある．

安定化理論は，対象となるアルゴリズムが代数的アルゴリズムでは収束することが証明されている．安定化定理を用いて既存のアルゴリズムを安定化した研究が近年多く報告されている [20][19][17]．しかし，どの精度で計算すれば安定するかという問題は未解決である．また，一般的なアルゴリズムを対象に適用した場合も興味深い．

Asir においても安定化技術を用いた計算を可能にするため，倍精度浮動小数点数の区間演算に加え，bigfloat による区間演算を実現した [7][8]．

5 bigfloat 区間演算の実現

文献 [6] で実現した double 型の区間数に加え，上下限を bigfloat とする区間数の型を実現する．現在配布されている Risa/Asir の bigfloat は，PARI[2] を用いている．そのため，本システムで実現した区間演算の上下限も PARI の bigfloat を用いた．具体的には，図 3 に示すように上限下限をそれぞれ bigfloat へのポインタとし，実際の計算には既に組み込まれているルーチンを利用した．ここで，PARI には丸めの制御を実現した関数が用意されていないため，

フラグ	下限: bigfloat へのポインタ	上限: bigfloat へのポインタ

図 3: bigfloat 区間数

結果を保証できないという点が問題となった．この問題点は，下限は演算結果の最下位桁の単位を引き，上限は加えるという Maple の INTPAK と同様の方法を用いることにより回避した．また，区間数用の組み込み関数を bigfloat 区間数に対応した．

安定化理論の研究では多くの場合，Maple 上の INTPAK が用いられている．Maple では bigfloat データを 10 進数の桁数で精度を指定できる．我々が実現した Asir では，PARI を用いているため 10 進の桁数を指定しても内部ではワード単位の計算となる．そのため，有効桁を指定する既存の関数 setprec() に加え，精度をワード単位で設定する関数 setprecword() を組み込んだ．

また，既存の組み込み関数の内部で発生する述語に対応するため，ゼロ書き換えモードを作成した．これは，

```
ctrl("zerorewrite",1);
```

とすることで有効になる．ゼロ書き換えモード時には，区間演算の結果にゼロが含まれる区間数をゼロに書き換える．モードを無効にするには，

```
ctrl("zerorewrite",0);
```

と入力する．

なお，[17] などの研究は本システムを用いて行われた．

6 現状

Asir の区間演算機能は，初期のソースコードに対して付加された．Risa/Asir の開発は別に並行して行われていたため，区間演算機能をソースコードにマージした時には，Asir には多くの数のデータ構造が追加されていた．そのためマージ時に，区間演算機能を標準で有効にすることができなかった．現在も区間演算機能は，ソースコード上は存在するが，多くの場合有効になっていない．唯一，FreeBSD で配布されている ports/packages システム上にある asir2000 では，区間演算機能を有効にし，コンパイルするように設定されている．Linux で利用するには，ソースコードからコンパイルしなおす必要がある．方法としては，

```
./configure --enable-interval --enable-pari --enable-plot
```

としてから，make すれば良い．

7 区間係数多項式のグラフ

区間係数多項式を扱う上でその取り得る値の範囲を見ることは問題解決のために非常に重要である．そのため，1 変数の区間係数多項式のグラフ描画プログラムを試作した．今，1 変数の区間係数多項式を

$$F(x) = \sum_{k=0}^{n} [\underline{a}_k, \overline{a}_k] x^k$$

とする．このとき，この多項式の取り得る範囲は，

$$\begin{aligned} F(x) &= [\underline{f}(x), \overline{f}(x)], \\ \underline{f}(x) &= \begin{cases} f_1(x) & (x \geq 0) \\ f_3(x) & (x \leq 0) \end{cases}, \end{aligned}$$

$$
\begin{aligned}
\overline{f}(x) &= \begin{cases} f_2(x) & (x \geq 0) \\ f_4(x) & (x \leq 0) \end{cases}, \\
f_1 &= \sum_{k=0}^{n} \underline{a}_k x^k, \\
f_2 &= \sum_{k=0}^{n} \overline{a}_k x^k, \\
f_3 &= \sum_{k=0}^{\lfloor n/2 \rfloor} \underline{a}_{2k} x^{2k} + \sum_{k=0}^{\lfloor (n-1)/2 \rfloor} \overline{a}_{2k+1} x^{2k+1}, \\
f_4 &= \sum_{k=0}^{\lfloor n/2 \rfloor} \overline{a}_{2k} x^{2k} + \sum_{k=0}^{\lfloor (n-1)/2 \rfloor} \underline{a}_{2k+1} x^{2k+1},
\end{aligned}
$$

となる [4]．ただし，$\lfloor a \rfloor$ は，a を越えない最大整数を表す．例えば，

$$F(x) = [1, 1.1] * x^3 + [-2.2, -2] * x^2 + [-1.1, -1] * x + [2, 2.2]$$

での $y = F(x)$ のグラフは，図 4 のようになる．

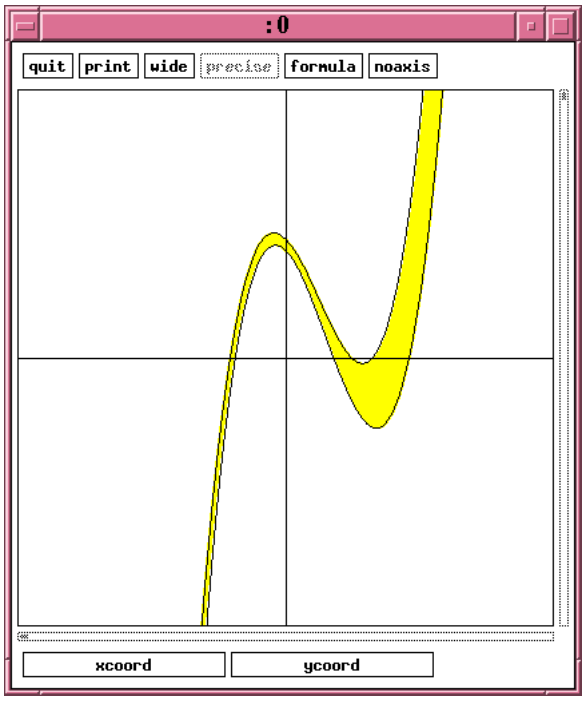

図 4: 区間係数多項式のグラフ例 ($-5 \leq x \leq 5, -5 \leq y \leq 5$)

8 まとめ

本論文では，国産の数式処理システム Risa/Asir を中心に区間演算と数式処理についてまとめた．Asir の区間演算機能は未完であるが，今後多くのユーザに利用していただきたい．そのためには，

- 多くのプラットフォームで区間演算機能を標準とする，
- 精度保証付き計算機能を充実させる，

など行っていきたい．また，精度保証付き数値計算の分野では，素朴な区間演算によるものではなく，丸め制御による精度保証付き数値計算 [15] が行われており，これらの結果も容易に利用できる環境を構築したい．

参 考 文 献

[1] G. Alefeld and J. Herzberger, *Introduction to Interval Computations*, Academic Press, New York, 1983.

[2] C. Batut, D. Bernardi, H. Cohen and M. Olivier, User's Guide to PARI–GP.

[3] A. E. Connell and R. M. Corless, An Experimental Interval Arithmetic Package in Maple, Interval Computations, **2**, 1993, 120–134.

[4] E.R. Hansen, G.W. Walster, Sharp Bounds on Interval Polynomial Roots,*Reliable Computing*, **8**, 2002, 115–122.

[5] J.B. Keiper: Interval Arithmetic in Mathematica, *Interval Computationns*, **3**, 1993, 76–87.

[6] 近藤祐史, 野田松太郎, 数式処理と区間演算との結合, 数式処理, **1**(2), 1992, 2–5.

[7] 近藤祐史, 野田松太郎, 数式処理システムにおける区間演算パッケージとその応用, 数式処理, **7**(1), 1998, 13–14.

[8] Y. Kondoh, M.-T. Noda, A software system for algorithm stabilization technique, *Proceedings of the 7th Asian Technology Conference in Mathmatics*, ATCM Inc., 2002, 360–367.

[9] U. Kulisch, John Wiley & Sons, *PASCAL–SC*, 1987.

[10] J.H.J. Molenkamp, J.A. van Hulzen and V.V.Goldman, An arbitary precision real interval arithmetic package in REDUCE, Memorandum INF–94–14, University of Twente, Enschede, The Netherlands, 1994.

[11] R.E. Moore, *Methods and Applications of Interval Analysis*, SIAM Studies in Applied Mathematics, SIAM, Philaderphia, 1979.

[12] 野田松太郎, 岩下英俊, パソコンで稼働する小型ハイブリッドシステム SYNC の設計", 情報処理学会論文誌, **30**(4), 1989, 419–426.

[13] M.-T. Noda and T. Sasaki, Approximate GCD and its application to ill–conditioned algebraic equations, Journal CAM, **38**, 1991, 335–351.

[14] M. Noro, T. Takeshima, Risa/Asir — A Computer Algebra System, *Proceedings of ISSAC'92*, 1992, 387–396.

[15] 大石進一，精度保証付き数値計算，コロナ社，2000.

[16] H. Sekigawa and K. Shirayanagi, Automatic Algorithm Stabilization System, *Josai Mathematical Monographs 2*, *NLA'99 Computer Algebra*, 2000, 159–168.

[17] K. Shiraishi, H. Kai, M.-T. Noda, Symbolic-numeric computation of Wu's method using stabilizing algorithm, *Proceedings of ATCM2001*, ATCM Inc., USA, 2001, 444–451.

[18] 白柳 潔，アルゴリズムの安定化理論，数式処理 **5**(2), 1997, 2–21.

[19] 白柳潔，新妻弘崇，実多項式行列スミス標準形の安定な浮動小数点計算法，情報処理学会論文誌，**40**(3), 1999, 1006–1017.

[20] 白柳潔，関川 浩，ゼロ書き換えに基づいた区間法と Sturm のアルゴリズムへの応用，電子情報通信学会論文誌 A, **J80-A**(5), 1997, 791–802.

[21] K. Shirayanagi, M. Sweedler, A Theory of Stabilizing Algebraic Algorithms, *Tech. Rep. 95-28*, Cornell Univ., 1995, 1–92.