数理解析研究所講究録
第 841 巻 1993 年 132-146

数式処理と区間演算の結合[1)]

— 複素区関数の場合 —

愛媛大学 工学部　近藤 祐史 (Yuji KONDOH)
〃　野田 松太郎 (Matu-Tarow NODA)

1. はじめに

最近，数値・数式融合計算 [3] 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る．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．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．そこで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．その点で融合計算で用いる浮動小数演算を区間演算に基づく誤差保証計算 [1][2] 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．そこで，我々は誤差保証付き計算を数式処理システムと結合し，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，数式処理システム risa/asir[4] に対しそのデータ構造に区間数の概念を取り入れることにより数式処理と区間演算の結合 [7] を実現した．本論文ではその機能の拡張として複素区間数への対応を報告する．さらに，本論では，システムの有効性を具体的に連立代数方程式の求解に適用して示す．また，数式処理と区間演算の結合により，記号解での変数の出現を少なくし安定した区関数での解を得る可能性を電気回路の中の梯子型抵抗回路解析に適用し示す．

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．区間演算とは区間数間の演算であり，以下では実区間数，複素区間数及びその演算について述べる．

2.1. 実区間数とその演算

実数全体を $\mathbb{R}$ であらわし，

$$A = \{x | a_1 \leqq x \leqq a_2\} \qquad x, a_1, a_2 \in \mathbb{R}$$

なる A を実区間数と呼び，$A = [a_1, a_2]$ と表す．ただし，$a_1 \leqq a_2$ とする．a_1, a_2 それぞれを実区間数の下限，上限と呼ぶ．また，2 つの実区間数 $A = [a_1, a_2]$, $a_1 \leqq a_2$, $B =$

[1)] A Computer Algebra System Combined with Interval Arithmetic — The Case of Complex Interval Arithmetic — by Yuji Kondoh, Matu-Tarow Noda, Department of Computer Science, Ehime University

$[b_1, b_2]$, $b_1 \leqq b_2$ 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．

$$\begin{cases} A+B &= [a_1,a_2]+[b_1,b_2]=[a_1+b_1, a_2+b_2], \\ A-B &= [a_1,a_2]-[b_1,b_2]=[a_1-b_1, a_2-b_2], \\ A\cdot B &= [a_1,a_2]\cdot[b_1,b_2]=[\min(a_1b_1,a_1b_2,a_2b_1,a_2b_2), \max(a_1b_1,a_1b_2,a_2b_1,a_2b_2)], \\ A\,/\,B &= [a_1,a_2]\,/\,[b_1,b_2]=[a_1,a_2]\cdot[1/b_2, 1/b_1] \\ &= [\min(a_1/b_2, a_1/b_1, a_2/b_2, a_2/b_1), \max(a_1/b_2, a_1/b_1, a_2/b_2, a_2/b_1)], \\ & (\text{ただし，}\ 0 \notin B). \end{cases}$$

実際に計算機で計算する場合には，結果の実区間の下限は切捨て，上限は切り上げる．

上の実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 を実区間数とすると，

$$\begin{aligned} &A+(B+C)=(A+B)+C \\ &A\cdot(B\cdot C)=(A\cdot B)\cdot C \\ &A+B=B+A \\ &A\cdot B=B\cdot A \end{aligned}$$

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．もし，B と C が同符号の実数だけを含むときは分配法則

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

が成立する．

2.2. 複素区間数とその演算

複素区間数の定義には次のような矩形型と円盤型の 2 種類がある．

- 矩形型：　(実区間数 $+\ i$ 実区間数)

$$\begin{aligned} A &= A_1+iA_2 \\ &= \{x=a_1+ia_2 | a_1\in A_1, a_2\in A_2\} \end{aligned}$$

(ただし，A_1, A_2は実区間数を表し，$i=\sqrt{-1}$である．)

また，2 つの矩形型複素区間数 A, B の間の演算を次のように定義する．

$$\begin{cases} A+B &= A_1+B_1+\boldsymbol{i}(A_2+B_2), \\ A-B &= A_1-B_1+\boldsymbol{i}(A_2-B_2), \\ A\cdot B &= A_1B_1-A_2B_2+\boldsymbol{i}(A_1B_2+A_2B_1), \\ A\,/\,B &= \dfrac{A_1B_1+A_2B_2+\boldsymbol{i}(A_2B_1-A_1B_2)}{B_1^2+B_2^2}, \\ & \quad (\text{ただし，} 0 \notin (B_1^2+B_2^2)). \end{cases}$$

- 円盤型： ＜複素数，半径＞

$$A =< a, r >= \left\{ z \in \mathbb{C} \middle|\ |a-z| \leqq r \right\}$$
$$(\text{ただし，}\ a \in \mathbb{C}, r \in \mathbb{R})$$

また，2 つの円盤型複素区間数 $A =< a, r_a >$, $B =< b, r_b >$ の間の演算を次のように定義する．

$$\begin{cases} A+B &= < a+b, r_a+r_b >, \\ A-B &= < a-b, r_a-r_b >, \\ A\cdot B &= < ab, |a|r_b+|b|r_a+r_ar_b >, \\ A\,/\,B &= A\cdot\dfrac{1}{B}, \\ & \dfrac{1}{B} =< \dfrac{\overline{b}}{b\overline{b}-r_b^2}, \dfrac{r_b}{b\overline{b}-r_b^2} >, \\ & (\text{ただし，}\ \overline{b}\ \text{は}\ b\ \text{の共役複素数}\ ). \end{cases}$$

この 2 つの複素区間数のうち，矩形型では複素数計算における実数計算を実区間演算に置き換えた形をしている．虚部の表現にも実区間数を用いるため扱いが容易である．また，2 つの区関数の交わり (intersection) が再び矩形型複素区間数になっている利点がある．円盤型では演算の手間が少なく結合法則が常に成り立つという利点がある．しかし，交わりは必ずしも円盤にならず計算に手間がかかる．そのため，本システムでは矩形型複素区間数を考える．

3. 実区間数を結合した数式処理システム

先ず，文献 [7] で実現したシステムの概要を述べる．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 [8]．しかし，この方法は区間演算パッケージの言語 (PASCAL-SC[5]) 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，本論では第 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のデータ構造に区間数を付加して実現する．そのため，区間数のデータ構造を付加する数式処理システムはソースコードが入手可能である必要がある．本研究では現在富士通国際研で開発中である数式処理システム risa/asir[4] を対象とする．

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実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 Fig. 1 のようになっている．ここに実区間数のデー

- 0. 零(0)
- 1. 数 …
  - 0. 有理数 … { 整数, 分数 }
  - 1. 倍精度浮動小数
  - 2. 代数的数
- 2. 多項式
- 3 ～ 8　有理式，リスト，ベクトル，行列，文字列，構造体

Fig. 1.　risa/asir のデータ構造

タ構造を付加する場合，

- 多項式などと同層に入れるか，
- 有理数や倍精度浮動小数と同層に入れるか，

が問題である．ここでは，多項式等の係数に実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などの数の層に入れ，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を利用する方法

を採用した．これにより区関数は次のようになる．

フラグ	下限: double 型のデータ	上限: double 型のデータ

160 bits

Fig. 2.　区間数の格納方法

3.2. 組み込み関数

実区間数に関する以下の関数を作成し組み込んでいる．

- システムからの実区間数 $A = [a_1, a_2]$ の入力: `itv`(a_1, a_2).
- 実区間数 A の中心: `mid`$(A) = (a_1 + a_2)/2$.

- 実区間数 A の幅: `width(A)` $= a_2 - a_1$.
- 実区間数 A の絶対値: `abs(A)` $= \max(|a_1|, |a_2|)$.
- 実区間数 A, B の間の距離: `distance(A,B)` $= \max(|a_1 - b_1|, |a_2 - b_2|)$.
- 実区間数 A の下限の取り出し: `inf(A)` $= a_1$.
- 実区間数 A の上限の取り出し: `sup(A)` $= a_2$.
- 実区間数 A, B の間の cap: `cap(A,B)` $= [\max(a_1, b_1), \min(a_2, b_2)]$, (ただし，結果が ϕ でない場合).
- 実区間数 A, B の間の cup: `cup(A,B)` $= [\min(a_1, b_1), \max(a_2, b_2)]$.
- 実数 a, 実区間数 A: *if* $a \in A$ *then* 1 *else* 0: `initv(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=itv(1,2)*x^2+itv(2,3);
i[ 1 , 2 ]*x^2+i[ 2 , 3 ]
[1] Q=itv(2,3)*x+itv(0,1);
i[ 2 , 3 ]*x+i[ 0 , 1 ]
[2] P+Q;
i[ 1 , 2 ]*x^2+i[ 2 , 3 ]*x+i[ 2 , 4 ]
[3] P*Q;
i[ 2 , 6 ]*x^3+i[ 0 , 2 ]*x^2+i[ 4 , 9 ]*x+i[ 0 , 3 ]
```

- 記号と区間数を係数に持つ多項式の和と積

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

また，実際に次の連立代数方程式

$$\begin{cases} f_1(x_1, x_2) &= x_1^2 + x_2^2 - 1 = 0 \\ f_2(x_1, x_2) &= x_1 - x_2^2 = 0 \end{cases} \tag{1}$$

の解をシステム上で求める．(1) を実数での Newton 法

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \quad k \geqq 0,$$

および，区間数に拡張した区間 Newton 法

$$X^{(k+1)} = m(X^{(k)}) - \frac{f(m(X^{(k)}))}{F'(X^{(k)})}, \quad k \geqq 0,$$

(ただし，$m(X)$ は区間 X の中点を表す．)

で解き，解の精度と演算回数を比較する．ここで，大文字は区間数およびその関数を表し，小文字は実数表す．実数計算は Sun/ELC 上の risa/asir 上で行ない，実区間数はここで開発したシステムを用いて，やはり Sun/ELC 上で計算したものである．初期値は $x_1^{(0)} = x_2^{(0)} = 1$ またはその区間数化したものを用い，結果を表 1 に示す．

Table 1. 方程式 (1) の解の 1 つの精度と演算回数

方法	解	誤差限界	Iter.	計算時間
実数 Newton (risa/asir)	$x_1 = 6.180339887498947E-01$ $x_2 = 7.861513777574232E-01$	—	5	60 msec
実区間 Newton	$x_1 = [\underline{6.180339887498922}E-01,$ $\underline{6.180339887498974}E-01]$ $x_2 = [\underline{7.861513777574213}E-01,$ $\underline{7.861513777574251}E-01]$	$5.22E-15$	5	80 msec

Iter. は反復回数を表す．

表 1 から明らかなように例題の方程式は区間演算を行なうと安心した誤差限界を持つ結果が得られ，収束するに要する反復回数も実数計算，区間演算で等しい．区間演算は，区間の上下限を計算するため，一般には実数計算の 2 倍以上の時間を必要とする．しかし，ここで開発したシステムでは risa/asir 上の実数計算の 1.3 倍程度となっている．これは区間数をデータ構造に加え，新しい計算システムとした成果が現れていると思われる．

次に同じ問題を通常の FORTRAN 計算 (Sun/ELC 上，倍精度) をした結果を示す．risa/asir のものと同じ初期値で

$x_1 = 6.180339887498983E-01, \quad x_2 = 7.861513777574278E-01$
反復回数 (Iter.): 8回
計算時間: 5 msec

の結果を得る．数式処理システムを経由しないので当然計算速度ははるかに高速になる．しかし，FORTRAN 計算での x_1, x_2 の数値結果は (1) が良条件問題であるにもかかわらず区間演算で保証した解の範囲に含まれていないことを注意する．

以上をまとめると，ここで開発したシステムには以下のような特徴がある．

- データ構造に実区間数を持ち，計算時間が実数計算に比較して 1.3 倍程度である．
- 多項式や有理式の係数に実区間数や記号を利用できる．
- 計算結果を再び数式処理の対象とし得る．

4. 複素区間数との結合

数式処理システム risa/asir の機能拡張に伴い複素数演算が可能になった [4]．この機能を利用して実区関数に対して **3.** で開発したシステムを複素区間数と数式処理とを結合したシステムに拡張する．方法は **3.** で述べた実区間数の結合の場合と同様に行うがデータ構造への注意が入要となる．

4.1. データ構造

複素数を含めることにより，risa/asir の数のデータ構造は Fig. 1 に加え，

3. bigfloat
4. 複素数

が追加されている．複素数のデータ構造は実部，虚部がそれぞれ複素数以外の数へのポインタで構成されている．矩形型複素区間数は **2.2.** でも述べたように複素数の実部，虚部をそれぞれ実区間数に置き換えた形をしているため，複素区間数を実現するのに既存の複素数のデータ構造が利用可能である．このため区間数のデータ構造を構築する場合，実区関数と同じに挿入すると，2 と 3 の間，3 と 4 の間，4 の後ろへ挿入などが考えられる．どの場合でも複素数のデータ構造が利用可能であるが，高速さと risa/asir のコードの変更が最小なように，3 と 4 の間に挿入する方法が最良である．

4.2. 組み込み関数

実区間数用のものと同様に複素区間数の演算を行う組み込み関数として次のように実区間数のものを拡張する．引き数が実区間数か複素区間数かによって自動的に対応する結果を出力する．

- 複素区間数 A の中点: $\texttt{mid}(A) = \mathrm{mid}(A_1) + i\,\mathrm{mid}(A_2)$
- 複素区間数 A の幅: $\texttt{width}(A) = \sqrt{\{\mathrm{width}(A_1)\}^2 + \{\mathrm{width}(A_2)\}^2}$
- 複素区間数 A の絶対値: $\texttt{abs}(A) = \sqrt{A_1^2 + A_2^2}$
- 複素区間数 A, B の距離: $\texttt{distance}(A,B) = \mathrm{distance}(A_1,B_1) + \mathrm{distance}(A_2,B_2)$
- 複素区間数 A の下限の取り出し: $\texttt{inf}(A) = \mathrm{inf}(A_1) + i\,\mathrm{inf}(A_2)$

- 複素区間数 A の上限の取り出し: $\mathtt{sup}(A) = \sup(A_1) + i\sup(A_2)$
- 複素区間数 A, B の cap: $\mathtt{cap}(A,B) = A_1 \cap B_1 + i(A_2 \cap B_2)$
- 複素区間数 A, B の cup: $\mathtt{cup}(A,B) = A_1 \cup B_1 + i(A_2 \cup B_2)$
- 複素数 c, 複素区間数 A: *if* $c \in A$ *then* 1 *else* 0: $\mathtt{initv}(A,c)$.

5. 実行例

5.1. 連立代数方程式の求解

区間演算を適用することの利点と既に本論で示した新しいシステム上での区間演算の実行の長所をみるために，再び連立代数方程式を取り上げる．方程式は

$$\begin{cases} f_1(x_1,x_2) &= x_1^2 + x_2^2 - 1 = 0, \\ f_2(x_1,x_2) &= x_1^2 - x_1x_2 - x_2^2 - 3 = 0 \end{cases} \tag{2}$$

であり，これを複素数での Newton 法

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}, \quad k \geqq 0,$$

および，区間数に拡張した複素区間 Newton 法

$$X^{(k+1)} = m(X^{(k)}) - \frac{f(m(X^{(k)}))}{F'(X^{(k)})}, \quad k \geqq 0,$$

(ただし，$m(X)$ は区間 X の中点を表す．)

で解き，解の精度と演算回数を比較する．計算は **3.3.** の場合と同じ計算環境で行なった．また，初期値は $x_1^{(0)} = x_2^{(0)} = 1 + i$ またはその区間数化したものを用い，結果をそれぞれ表 2 に示す．

表 1 の場合と同様，複素演算でも区間演算を行なうと安心した誤差限界を持つ結果を得ることができる．収束するに要する反復回数も複素数計算，複素区間演算で等しい．計算時間は実区間数の場合に **3.3.** で述べたと同様，risa/asir 上の複素数計算の 1.3 倍程度になっている．上で見るように本システムを用いると，

1. 計算時間が実数計算に比較して 1.3 倍程度であり高速である．
2. 浮動小数計算にはない誤差保証を十分な精度で与え得る．
3. 係数が区間数の多項式や有理式の数式処理が可能である．

などの特徴がある．なお，実区間数の場合と同様の FORTRAN 計算を行うと，

$x_1 = 1.3207676521629064 + 0.21077758656915083i$
$x_2 = -0.31179094839324301 + 0.89286818484026953i$
反復回数 (Iter.): 8回
計算時間: 11 msec

と得ることを付言する．

Table 2. 方程式 (2) の解の 1 つの精度と演算回数

方法	解	誤差限界	Iter.	計算時間
複素 Newton 法 (risa/asir)	$x_1 = 1.3207676521629061$ $+ 0.21077758656915082i$ $x_2 = -0.31179094839324300$ $+ 0.89286818484026964i$	—	6	150 msec
複素区間 Newton 法	$x_1 = [1.3207676521629061,$ $1.3207676521629065]$ $+ [0.21077758656915074,$ $0.21077758656915096]i$ $x_2 = [-0.31179094839324317,$ $-0.31179094839324289]$ $+ [0.89286818484026941,$ $0.89286818484026986]i$	$4.44E-16$	6	200 msec

Iter. は反復回数を表し，$i = \sqrt{-1}$ である．

5.2. 梯子型抵抗回路の解析

奥村ら [6] は区間演算を電気回路の中の梯子型抵抗回路の解析に適用し，その有効性を見ている．この問題を数式処理と併用して本システムで解析してみる．Fig. 3 のような 3 段の梯子型抵抗回路を考える．このような回路を実際に製作した場合には各抵抗器は理論値とはわずかに違っており正確な解析は困難である．この問題では直流電源と抵抗器で構成されている回路であるので複素区間数は用いないが，交流電源でコイルやコンデンサなどが使われている電気回路では複素数計算での解析をする必要がある．

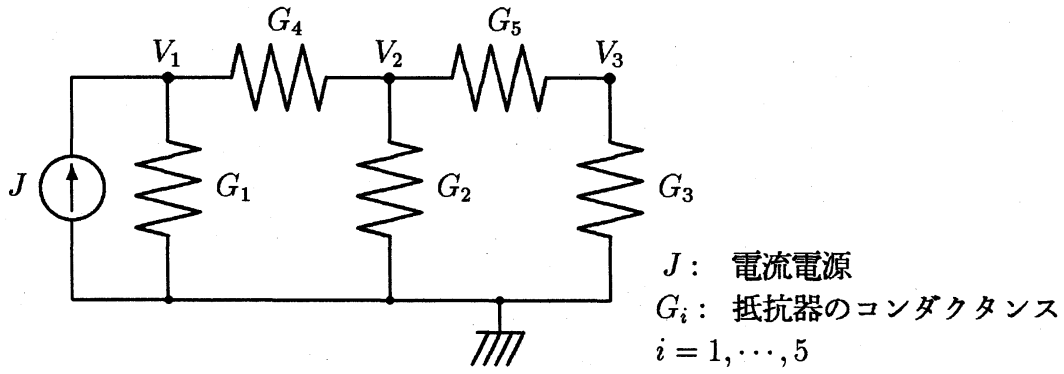


Fig. 3. 梯子型抵抗回路

Fig. 3 のような回路の回路方程式はキルヒホッフの法則により次のような連立 1 次方程

式になる.

$$\begin{cases} V_1G_1 + (V_1 - V_2)G_4 - J = 0 \\ V_2G_2 + (V_2 - V_1)G_4 + (V_2 - V_3)G_5 = 0 \\ V_3G_3 + (V_3 - V_2)G_5 = 0 \end{cases}$$

ここで, J, G_i, $i = 1,\cdots,5$ は $J = 10[1-\varepsilon, 1+\varepsilon]$, $G_i = [1-\varepsilon, 1+\varepsilon]$, $i = 1,\cdots,5$, $0 \leqq \varepsilon \leqq 0.1$ とする.

この方程式を解く場合次の 2 通りの方法が考えらえる.

1. 数値を代入して, ガウス消去により数値解を求める.

2. 記号的にガウス消去をして, 数値を代入する.

後者が可能なことが本システム適用の利点である. 方程式の記号解を数式処理により求めると,

$$\begin{cases} V_1 = \dfrac{(JG_3 + JG_5)G_4 + (JG_3 + JG_5)G_2 + JG_3G_5}{((G_3 + G_5)G_4 + (G_3 + G_5)G_2 + G_3G_5)G_1 + ((G_3 + G_5)G_2 + G_3G_5)G_4} \\ V_2 = \dfrac{(JG_3 + JG_5)G_4}{((G_3 + G_5)G_4 + (G_3 + G_5)G_2 + G_3G_5)G_1 + ((G_3 + G_5)G_2 + G_3G_5)G_4} \\ V_3 = \dfrac{JG_4G_5}{((G_3 + G_5)G_4 + (G_3 + G_5)G_2 + G_3G_5)G_1 + ((G_3 + G_5)G_2 + G_3G_5)G_4} \end{cases} \tag{3}$$

となる.

ここで, 2 の方法ではこの記号解に区間数の数値を代入するわけであるが, 一般に区間演算では各変数が一度しか現れない場合には正確にその範囲を表すが, 各変数が 2 回以上現れる場合には区間数を代入したときの評価は正確な値より広くなる事が知られている [1][2].

例えば,

$$f_1(x,y) = \frac{xy}{1-x} \qquad x \neq 1, x \neq 0$$

$$f_2(x,y) = \frac{y}{\dfrac{1}{x} - 1} \qquad x \neq 1, x \neq 0$$

$$X = [2,3], Y = [0,1]$$

では, $f_1(x,y)$ と $f_2(x,y)$ は同じ式である. 今, $x = X$, $y = Y$ での $f_1(x,y)$ を評価すると,

$$\begin{aligned} f_1(X,Y) &= [\min_{x\in X, y\in Y}\{f_1(x,y)\}, \max_{x\in X, y\in Y}\{f_1(x,y)\}] \\ &= [-2,0] \end{aligned}$$

となる. しかし, $f_1(x,y)$,$f_2(x,y)$ に $x = X$, $y = Y$ を代入すると,

$$f_1(X,Y) = \frac{[0,1][2,3]}{1-[2,3]} = [-3,0]$$

$$f_2(X,Y) = \frac{[0,1]}{\dfrac{1}{[2,3]} - 1} = [-2,0]$$

となり，f_2 の方が区間の幅が小さく，また，その下限，上限はそれぞれ $x = X, y = Y$ での f_1 及び f_2 の正確な最小値，最大値となっている．このように，区間演算では各変数の出現回数を最小にした後，代入する必要がある．このため記号解 (3) 中の各変数の出現回数を最小にする．この過程で数式処理を活用する．結果は，

$$\begin{cases} V_1 = \dfrac{J}{G_1 + \dfrac{1}{\dfrac{1}{G_4} + \dfrac{1}{G_2 + \dfrac{1}{\dfrac{1}{G_5} + \dfrac{1}{G_3}}}}} \\ V_2 = \dfrac{J}{G_1 + (G_2 + \dfrac{1}{\dfrac{1}{G_5} + \dfrac{1}{G_3}})(1 + \dfrac{G_1}{G_4})} \\ V_3 = \dfrac{J}{(1 + \dfrac{G_3}{G_5})(G_2(1 + \dfrac{G_1}{G_4}) + G_1) + G_3(1 + \dfrac{G_1}{G_4})} \end{cases} \tag{4}$$

となる．

これらにより，

1. 区間数の数値を代入して，ガウス消去により数値解を求める場合，
2. 記号解 (3) に区間数の数値を代入する場合，
3. 各変数の出現回数を最小にした記号解 (4) に区間数の数値を代入する場合

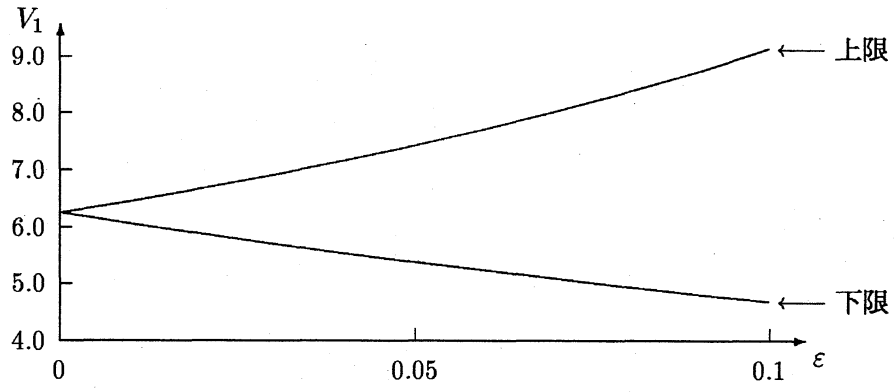
に於ける結果を Fig. 4~6 に示す．

これらより，V_1, V_2, V_3 のどれに於いても 3 の場合が一番狭い範囲を求められている．これは本システムのように数式処理と区間演算を結合したシステムの利点であると言える．この程の問題に対しては 3 の場合のように各変数の出現回数を最小にする必要があることを示している．複雑な式中の変数の出現回数を最小にするアルゴリズムは現在では確立されていないが，本システムでの結果の良さからみて，今後このような構成的アルゴリズムを設計する必要が大きいと言える．

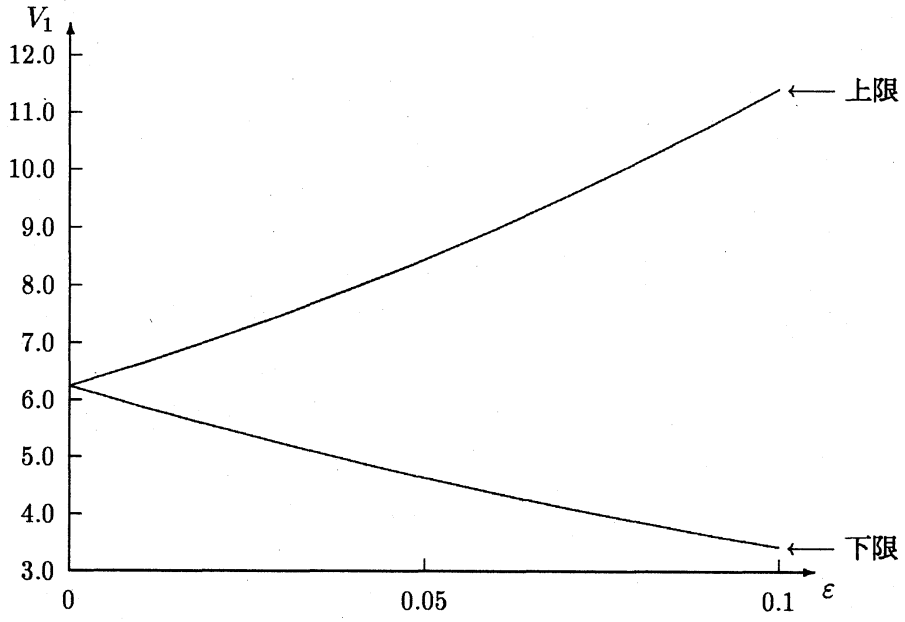
6. まとめ

本論文では数式処理システムへの区間演算を付加した科学計算システムの開発を行なった．この中，実区間数との結合についてはすでに文献 [7] で報告したが，これを複素区間数との結合まで拡張した．本システムの特徴には，

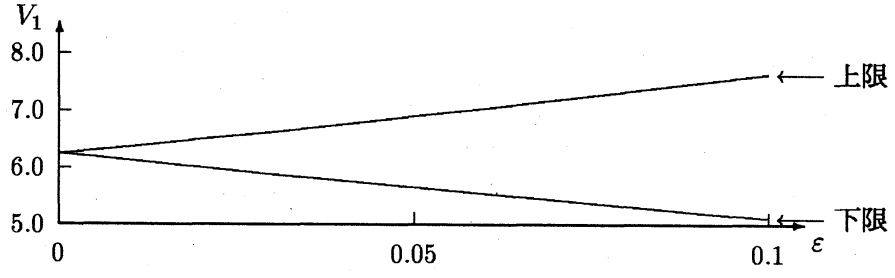
- 実区間数及び複素区間数を係数に持つ多項式や要素に持つ行列等の記号計算 (数式処理) が可能である．
- 数式処理を使用するので通常の数値計算より高精度で安定した解が得られる．



(a) 区間演算 (数値計算)

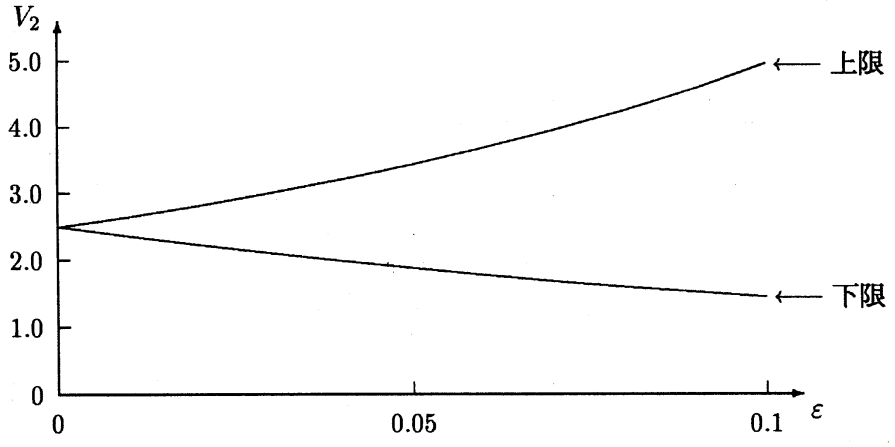


(b) 記号解 (3) に代入した場合

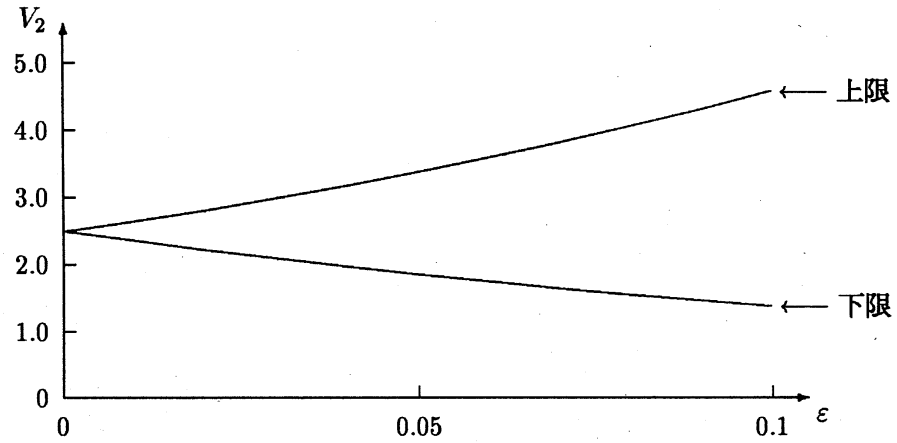


(c) 記号解 (4) に代入した場合

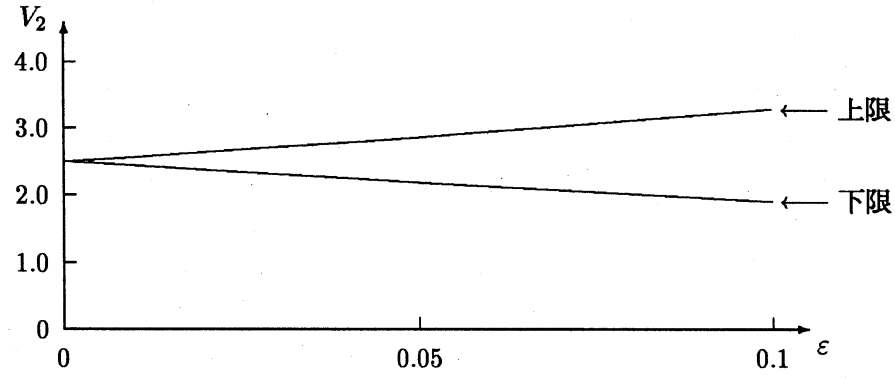
Fig. 4. V_1 (Fig. 3) の結果



(a) 区間演算 (数値計算)

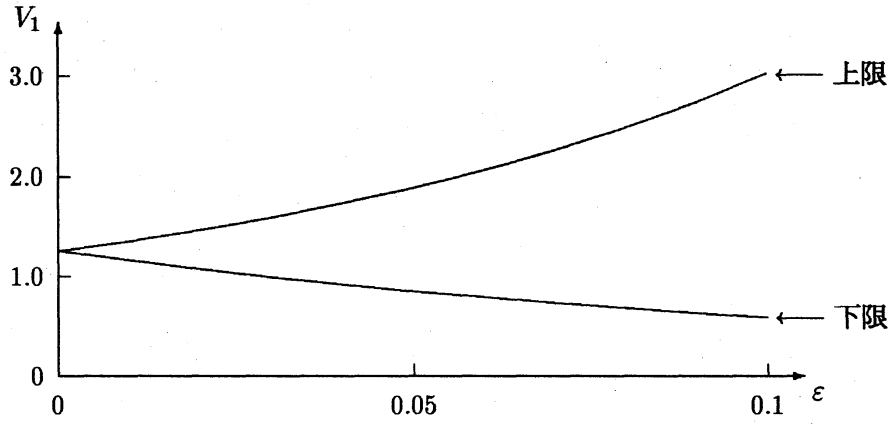


(b) 記号解 (3) に代入した場合

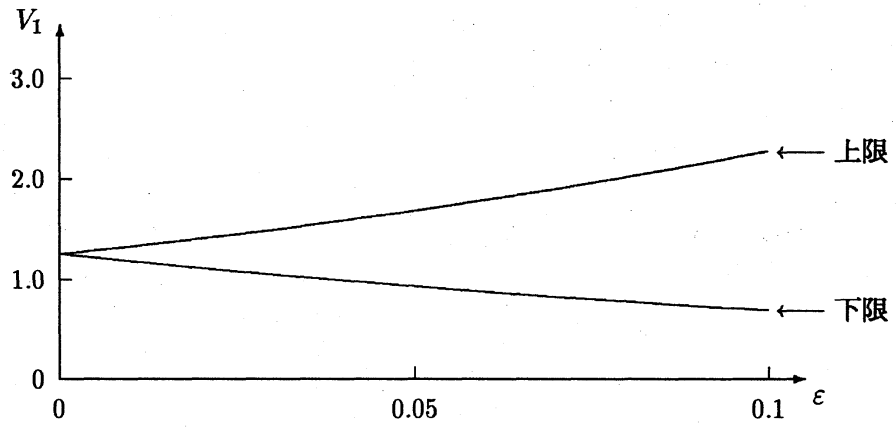


(c) 記号解 (4) に代入した場合

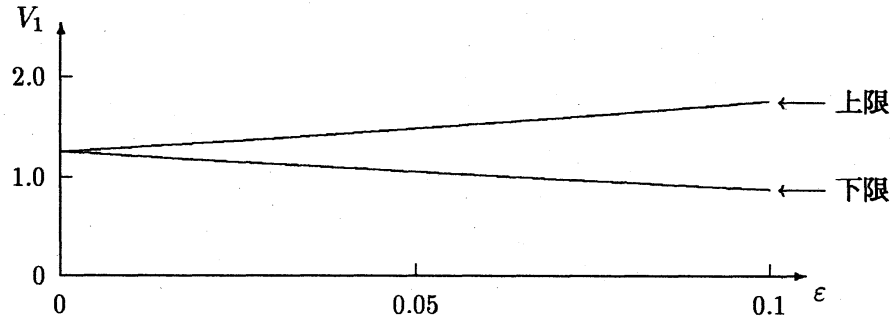
Fig. 5. V_2 (Fig. 3) の結果



(a) 区間演算 (数値計算)



(b) 記号解 (3) に代入した場合



(c) 記号解 (4) に代入した場合

Fig. 6. V_2 (Fig. 3) の結果

- データ構造に注意したので通常の区間演算プログラムより高速な演算が期待できる.

などがあることを実行例とともに示した. 今後システムを完備するための課題として三角関数, 指数関数, 対数関数などの関数への対応がある.

今後本システムを悪条件問題を中心とした問題に適用することによって, 今まで困難だった多くの問題の解を数値的にあるいは数式的に求め得る可能性がある. また, このような場合に本システムはさらに強力になるであろう.

参 考 文 献

[1] G. Alefeld and J. Herzberger, Introduction to Interval Computations (J. Rokne, Trans.), Academic Press, New York, 1983.

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

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

[4] M. Noro and T. Takeshima, Risa/Asir: A Computer Algebra System, in Proc. ISSAC '92, 1992 at Berkeley, CA.

[5] PASCAL-SC (U. Kulisch, Ed.), John Wiley & Sons, 1987.

[6] K. Okumura and A. Kishima, On applications of interval arithmetic to circuit analysis, 数理解析研究所講究録 673 "自己検証的算法とその応用", 1988.

[7] 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, Vol. 1, No. 2, pp. 52–55.

[8] 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, Vol. 30, No. 4, pp. 419–426, 1989.