実多項式行列スミス標準形の安定な浮動小数点計算法

白柳 潔[†] 新妻弘崇[††]

実数を成分とする行列 A に対し，その特性行列 $A-xE$ のスミス標準形は，線形代数において基本的である．スミス標準形は，行と列の基本演算とユークリッドの互除法との組合せによって計算できる．しかし，その計算に浮動小数点近似を用いると，不安定性の問題が生じることがある．すなわち，入力行列の各係数の精度桁を上げても，その出力の行列は，正確なスミス標準形に収束するとは限らない．この論文では，著者の1人と M. Sweedler が提案した安定化手法をスミス標準形のアルゴリズムに適用し，スミス標準形を浮動小数点で計算するための安定なアルゴリズムを実現する．さらに，多くの例に対して計算機実験を行い，そのアルゴリズムの安定性を実証する．これによって，安定化手法の有効性が示される．

A Stable Method to Compute Smith Normal Form of Polynomial Matrices Using Floating-point Computation

KIYOSHI SHIRAYANAGI[†] and HIROTAKA NIITSUMA[††]

For a matrix A with real entries, the Smith normal form of the characteristic matrix $A-xE$ is of fundamental use in linear algebra. Smith normal form is obtained by combining row and column reduction with the Euclidean algorithm. Such algorithms are not typically stable with respect to limited precision computation. Namely, if the algorithm is executed with limited precision computation, the output does not necessarily approach the Smith normal form as the precision increases. The first author and M. Sweedler have proposed techniques of stabilizing algorithms to allow limited precision computation in a convergent manner. In this paper we present an algorithm—which is amenable to these techniques—for computing Smith normal form. We then apply the techniques to obtain an algorithm which when executed with limited precision will converge to Smith normal form as the precision increases. The convergence also has the property of being "fast convergence" at zero. More specifically, any coefficient which will converge to zero reaches zero in a finite number of steps. Experimental results are given to show the stability and usefulness of the algorithm.

1. はじめに

大量の情報データを行列で表現し，その上で種々の処理を行うことは多くの場面で出現する．このとき，スミス標準形やジョルダン標準形などの行列の標準形は，効率化や簡単化の点で重要な役割を果たす．本論では，数ある標準形の中からスミス標準形に焦点を当てる．ジョルダン標準形は，スミス標準形から直接導くことができる．

スミス標準形は，行と列の基本的な変形操作とユークリッドの互除法の組合せによって計算できる．しかし，厳密計算でそれを行うと，特に行列のサイズが大きいときは，膨大な時間と記憶容量を必要とする．そこで，浮動小数点近似計算によって，その負荷が軽減できればよい．ところが，上述の計算法は厳密計算用のアルゴリズムであり，それをそのまま愚直に適用すると，入力値の精度桁がどれほど大きくなっても，答が正確なスミス標準形に収束するとは限らない．

本論では，入力の精度を一定ずつ上げて実行を繰り返したとき，その各出力が正確なスミス標準形に収束することを保証するような安定なアルゴリズムを与える．さらに，実際にそれを計算機で実行して，その有効性を示す．提案する方法は，単純に従来のアルゴリズムに浮動小数点計算を導入したものではなく，文献1)の安定化手法のテクニックを導入する．この安定化手法は，アルゴリズムがある条件を満たせば，そのアルゴリズムを新しいアルゴリズムに変換し，その新しいアルゴリズムをある値以上の精度桁で実行すれば，正確な出力に近い答を返すものである．本論では，スミス標準形のアルゴリズムがその適用条件を満たすこ

† NTT コミュニケーション科学基礎研究所
NTT Communication Science Laboratories
†† 奈良先端科学技術大学院大学
Nara Institute of Science and Technology

とを確認し，実際に安定化手法を適用する．なお，文献 2) は文献 1) を日本語で要約したもの，文献 3) は一般読者向けに平易に解説したものである．

まず 2 章では，安定化手法の概略を述べる．3 章では，安定化手法を適用する対象となるアルゴリズムについて述べる．スミス標準形を効率的に求めるアルゴリズムは数多く存在するが，本論では，効率性よりも安定性に重点を置き，不安定性の原因を明確に説明するなどの理由で，最も素朴と思われるアルゴリズムを選んだ．3.2 節では，そのアルゴリズムの不安定性の原因について詳しく述べる．4.1 節では，そのアルゴリズムに安定化手法を適用する．4.2 節で，安定化の効果を見るための簡単な例を示し，4.3 節では，より大きな行列についての計算機実験を報告し，本提案手法の有効性を示す．

2. アルゴリズムの安定化手法

文献 1) に基づき，次のクラスのアルゴリズムについて安定化手法を説明する．

- 入力，中間ステップ，出力のすべてのデータは，多変数多項式環 $R[x_1,\ldots,x_m]$ に属する．R は実数体の部分体である．
- アルゴリズムで行われる操作は，$R[x_1,\ldots,x_m]$ における加算，減算，乗算，剰余計算のみである．
- 述語の不連続点は，あるとすれば 0 のみである．

ここで，剰余計算とは，多項式を多項式で割ったときの剰余を計算することである．変数が 1 つしかない場合は，普通の多項式の除算のアルゴリズムを使う．剰余計算において，割られる側の多項式の各係数を，割る側の多項式の主項の係数で割るときに，体 R での除算が行われる．多項式の主項とは，その多項式を構成する「係数が 0 でない」単項式のうち，次数が最大のものをいう．2 変数以上の場合は，主項を定義するための項順序が必要となるが，スミス標準形を計算する場合は 1 変数の多項式しか取り扱わないので，ここでは項順序についての詳細に立ち入らない．

次に述語の不連続点について説明する．述語は多項式の集合から

$$\{\mathrm{TRUE}, \mathrm{FALSE}\}$$

への写像である．述語 p が $f \in R[x_1,\ldots,x_m]$ で不連続であるとは，$f_i \to f$ となる $R[x_1,\ldots,x_m]$ の要素の列 $\{f_i\}_i$ が存在して，$p(f_i) \not\to p(f)$（すなわち $p(f_i) \neq p(f)$）となることをいう．ここに，$f_i \to f$ とは f_i が係数ごとに f に収束することを示す．係数の収束は普通の実数集合における収束である．したがって，0 が述語 p の不連続点であるとは，0 多項式（すべての係数が 0 である多項式）において述語 p が不連続であることである．

上述の 3 つの条件を満たすアルゴリズムを不連続点 0 の代数的アルゴリズムと呼ぼう．多項式を扱う大抵の数式処理のアルゴリズムは，不連続点 0 の代数的アルゴリズムであるか，またはそれに変換できるものである．たとえば，$X = 3$? という述語は，$Y := X - 3$ という変数置換えと $Y = 0$? という不連続点 0 の述語に変換できる．

簡単のために，アルゴリズムに現れる操作は，多項式演算と剰余計算しかないとしているが，文献 1) では平方根や微分などの他の多くの関数についても議論している．

$\mathcal{A}$ を不連続点 0 の代数的アルゴリズムとする．$\mathcal{A}$ は近似入力に対して，不安定となることがある．たとえば，次のようなアルゴリズム $\mathcal{B}$ を考える．

初期設定：	(X)
1.	$Y = 3X - 1$
2.	goto 4. if $Y \geq 0$
3.	stop (0)
4.	stop (1)

アルゴリズム $\mathcal{B}$ は，入力 X に対して，$3X - 1 \geq 0$ ならば 1 を返し，そうでなければ 0 を返す．入力が $X = 1/3$ の場合を考えよう．$\mathcal{B}(1/3)$ は 1 の値を返す．しかし，任意の精度桁 μ に対し，浮動小数近似 $(1/3)_\mu$ は $0.333\cdots3$（μ 桁）となり，$\mathcal{B}((1/3)_\mu)$ はいつも 0 を返してしまう．これが不安定性である．換言すれば，$(1/3)_\mu \to (1/3)$ であるが $\mathcal{B}((1/3)_\mu) \not\to \mathcal{B}(1/3)$ である．$\mathcal{B}$ は，不連続点として $\{0\}$ を持つ述語 $Y \geq 0$ を持っているので，不連続点 0 の代数的アルゴリズムである．この不連続点があるため，述語の評価（条件分岐）が正しく行われず，アルゴリズムの正しい実行過程（元のアルゴリズムを厳密計算で実行したときの過程）をたどることができない．これが不安定性の原因である☆．

アルゴリズムの安定化は，アルゴリズムの構造を変えずにデータ集合を変えることで行われる．データ集合は，元の係数から区間係数に変えられる．区間係数は，2 つの浮動小数点数のペア $[A, \alpha]$ で表され，A は

☆ ただし，この例はアルゴリズムに潜む不安定性を説明するだけのためである．実際の数値計算による 0 判定にはいろいろな工夫がなされている．たとえば，適当なガード桁を用意して，その桁数以下に浮動小数の値が収まれば，その数を 0 と見なすという便法がある．しかし，ガード桁をどの程度にすればよいかという難しい問題があり，数値計算の分野では誤差解析などでさかんに研究されている．

中心，α は半径を意味する．$[A,\alpha]$ は，$|x-A|\leq\alpha$ なる実数 x の全体を表現する．区間には，下界と上界で挟む形式（矩形区間）もあるが，本論では中心と半径による円区間を採用する．区間の詳細については，文献 4) を参照されたい．元のアルゴリズムにおいて係数の間で演算が行われる部分は，区間係数の間で区間演算が行われる．キーポイントは，述語を評価する直前に，「ゼロ書換え」を行うことである．具体的には，区間係数が 0 を含むとき，その区間係数を 0 区間（中心 0 半径 0 の区間）に書き換える．それ以外のときは，何も変えず，そのままとする．ゼロ書換えを行った後，その区間係数の第 1 要素，つまり区間の中心において述語を評価する．ゼロ書換えの重要な特徴は，区間係数の列の収束性を保存すること，そして，区間係数の列が 0 に "近似的に" 収束する（定義は後述）ならば，ゼロ書換えされた区間係数の列は有限回で 0 に「到達する」という点である．したがって，述語がまさに不連続点の 0 で評価できるようになる．もしゼロ書換えを行わなかったとしたら，述語は 0 の値で評価されない可能性があり，もとより 0 はその述語の不連続点であったため，アルゴリズムは元のアルゴリズムと同じ実行過程をたどらなくなる．

アルゴリズムの安定化手法についてまとめよう．$\mathcal{A}$ を不連続点 0 の代数的アルゴリズム，$Int(\mathcal{A})$ を文献 1) のテクニックによって安定化されたアルゴリズムとする．アルゴリズム $Int(\mathcal{A})$ は次の特徴を持つ．

区間領域 データ領域は区間を係数に持つ多項式の集合である．区間係数は $[A,\alpha]$ の形をとる．ここで $A\in R$ であり，α は非負の実数である．$[A,\alpha]$ は，集合 $\{x\in R\mid |x-A|\leq\alpha\}$ を表現する．

区間演算 区間係数の加減乗除に対しては，区間演算を用いる[4]．区間係数間の二項演算子 $*\in\{+,-,\times,\div\}$ に対し，

$$[A,\alpha]*[B,\beta]=[A*B,\gamma_*]$$

である．ここに，γ_* は，

$$|x-A|\leq\alpha,$$
$$|y-B|\leq\beta\Rightarrow|x*y-A*B|\leq\gamma_*$$

を満足するように定義される．

ゼロ書換え 不連続点 0 を持つ述語が評価される直前に，ゼロ書換えを行う．すなわち，各区間係数 $[C,\gamma]$ に対して，$|C|\leq\gamma$ ならば $[C,\gamma]$ を [0,0] に書き換える．そうでなければ，$[C,\gamma]$ のままとする．

$Int(\mathcal{A})$ は区間係数多項式（の集合）を入力として受け入れ，区間係数多項式（の集合）を出力として掃き出す．$\mathcal{A}$ に対する入力 $f\in R[x_1,\ldots,x_m]$ は，$Int(\mathcal{A})$ に対する入力のために，区間係数多項式の列 $\{Int(f)_j\}_j$ に近似される．具体的には，f を $\sum_{i_1,\ldots,i_m}a_{i_1\ldots i_m}x_1^{i_1}\cdots x_m^{i_m}$ と表すと，$Int(f)_j$ は次のように定義される．

$$\sum_{i_1,\ldots,i_m}[(a_{i_1\ldots i_m})_j,(\alpha_{i_1\ldots i_m})_j]\,x_1^{i_1}\cdots x_m^{i_m}$$

ここに，ある l があって，$j\geq l$ ならば，

$$|a_{i_1\ldots i_m}-(a_{i_1\ldots i_m})_j|\leq(\alpha_{i_1\ldots i_m})_j$$

であり，$j\to\infty$ のとき，すべての $i_1\ldots i_m$ に対して $(\alpha_{i_1\ldots i_m})_j\to 0$ である．このとき，$Int(f)_j$ は f に近似的に収束すると呼ぶ．各 j に対し，$Int(\mathcal{A})$ を入力 $Int(f)_j$ で実行したとき，その出力を $Int(\mathcal{A})(Int(f)_j)$ と表記する．

次の定理は $Int(\mathcal{A})$ の特徴を述べている．

定理 1（係数収束[1]） アルゴリズム $\mathcal{A}$ は入力 $f\in R[x_1,\ldots,x_m]$ に対して正常終了し，f に近似的に収束する区間係数多項式の列 $\{Int(f)_j\}_j$ が与えられているとする．このとき，ある n が存在して，$j\geq n$ ならば，$Int(\mathcal{A})$ は入力 $Int(f)_j$ に対して正常終了し，かつ，$j\to\infty$ のとき，

$$Int(\mathcal{A})(Int(f)_j)\to\mathcal{A}(f)$$

となる．

しかし，この係数ごとの収束だけでは，実は正確なスミス標準形を計算するという目的にはまだ不十分である．最終的にもう 1 回ゼロ書換えを行う必要がある．すなわち，出力 $Int(\mathcal{A})(Int(f)_j)$ の各区間係数にゼロ書換えを行う．そして，各区間係数の第 1 要素のみを取り出して，出力を普通の実係数多項式に戻す．このように，基本的には $Int(\mathcal{A})$ と同じであるが，最後にその出力をゼロ書換えし，実係数多項式への変換を行うアルゴリズムを以下 $Int(\mathcal{A})_R$ と表す．

最後のゼロ書換えの効果は，「台収束」と呼ばれる，係数収束よりも強い収束を実現することである．台収束について説明するために多項式の台を定義する．多項式

$$f=\sum_{i_1,\ldots,i_m}a_{i_1\ldots i_m}x_1^{i_1}\cdots x_m^{i_m}$$

の台とは，0 でない係数を持つ変数の巾積（係数 1 の単項式）の集合

$$\{x_1^{i_1}\cdots x_m^{i_m}\mid a_{i_1\ldots i_m}\neq 0\}$$

であり，$Supp(f)$ と書く．同様に区間係数多項式 F に対しても $Supp(F)$ は，$[0,0]$ でない区間係数を持つ巾積の集合である．直観的にいえば，台は「多項式の形」である．$Int(\mathcal{A})_R(Int(f)_j)$ の形は有限回のステップで（ある有限の j で），$\mathcal{A}(f)$ の形と同じになる．換言すれば，$Int(\mathcal{A})_R(Int(f)_j)$ の係数で，0 に

収束するものは，有限ステップで0に到達する．これを台収束と呼ぶ．これが，正確なスミス標準形の計算のために必要となる．

次の定理が，$Int(\mathcal{A})_R$ の安定性を述べるものである．

定理2（台収束[1]） 定理1と同じ仮定を置く．このとき，ある n が存在して，$j \geq n$ のとき，$Int(\mathcal{A})_R$ は入力 $Int(f)_j$ に対して正常終了し，かつ，

(1) $Int(\mathcal{A})_R(Int(f)_j) \to \mathcal{A}(f)$ $(j \to \infty)$，そして，

(2) ある N があって，$j \geq N$ ならば，

$$Supp(Int(\mathcal{A})_R(Int(f)_j)) = Supp(\mathcal{A}(f))$$

となる．

係数収束や台収束の実用例として，ブッフバーガーのアルゴリズム[5),6)]やスツルムのアルゴリズム[7)]などがある．その他の例は，文献3)を参照されたい．

3. スミス標準形

3.1 アルゴリズム

ここでは，成分が $R[x]$ の要素である正方行列に対して，スミス標準形と単因子に関する理論を復習する．この理論は単項イデアル整域上の矩形行列に対しても成立する[8),9)]．成分が R の要素である $n \times n$ の正方行列の集合を $M_n(R)$ で表す．$M_n(R[x])$ は，x についての R 係数の多項式を成分に持つ行列の集合である．$A \in M_n(R)$ とする．本論の目標は，A の特性行列 $A - xE \in M_n(R[x])$ のスミス標準形を計算することである．ここで E は $M_n(R)$ の単位行列を表す．

スミス標準形は $M_n(R[x])$ での行と列の基本変形によって計算できる．基本変形とは次の3種類の操作をいう．

- 行または列の入換え．
- 行または列ベクトルのスカラ倍．
- 行（列）ベクトルを多項式倍したものを別な行（列）ベクトルに加えること．

$A(x), B(x) \in M_n(R[x])$ とするとき，$A(x)$ が有限回の基本変形で $B(x)$ に移るとき $A(x)$ と $B(x)$ は対等であるという．

定理3（スミス標準形） 任意の $A(x) \in M_n(R[x])$ は，次の対角行列と対等である．

$$\begin{bmatrix} e_1(x) & & & & & & \\ & e_2(x) & & & & & \\ & & \ddots & & & & \\ & & & e_r(x) & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{bmatrix} \tag{1}$$

ここに，$e_i(x)$ $(i = 1, \ldots, r)$ は以下を満足する．

(1) $e_{i-1}(x)$ は $e_i(x)$ を割り切る $(i = 2, \ldots, r)$．

(2) $e_i(x)$ $(i = 1, \ldots, r)$ の主係数は1である．

定理に現れる多項式 $e_1(x), \ldots, e_r(x)$ を $A(x)$ の単因子と呼ぶ．単因子は，$A(x)$ から一意に決まる．式(1)の行列を $A(x)$ のスミス標準形と呼ぶ．

定理3はよく知られた定理であるが，ここではその詳しい証明を載せる．その理由は，その証明が構成的で，そのままでスミス標準形を計算するアルゴリズムを表現していること，そのアルゴリズムの不安定性の原因を説明するのに簡便であること，実際にそのアルゴリズムを実装してそれを安定化し，計算機実験を行ったことの3つである．なお，証明には文献10)を参考にした．

証明：

証明は，いくつかのステップからなる．ここで $A(x) = [a_{ij}]_{1 \leq i,j \leq n}$ とする．

(I) 「$A(x) = 0$ ならば $A(x)$ はスミス標準形である．」

以下では $A(x) \neq 0$ の場合を考える．

(II) 「任意の (i, j) に対して，$A(x)$ と対等な $B(x)$ が存在して，$b_{11} = a_{ij}$ である．ここに，b_{11} は $B(x)$ の $(1,1)$ 成分である．」

これは明らかである．$i \neq 1$，$j \neq 1$ のときは $A(x)$ の i 行と1行を交換し，次に j 列と1列を交換すればよい．

(III) 「a_{11} が $A(x)$ の第1行（第1列）の成分をすべて割り切るならば，$A(x)$ と対等な $B(x)$ が存在して，$b_{11} = a_{11}$ であり，$(1,1)$ 成分以外の第1行（第1列）の成分が0である．」

もし，$a_{1j} = q_j a_{11}$ $(j \neq 1)$ ならば，j 列から1列の q_j 倍を引けば，第1列の $(1,1)$ 成分以外を0にできる．

(IV) 「第1行（第1列）に a_{11} で割り切れない成分があるとき，$A(x)$ と対等な $B(x)$ が存在して，$deg(b_{ij}) < deg(a_{11})$ なる b_{ij} を持つ．」

たとえば，a_{i1} が a_{11} で割り切れないとき，

$$a_{i1} = qa_{11} + r,\ 0 < \deg r < \deg a_{11}$$

となる2つの多項式 q，r が存在する．第1行を q 倍してそれを i 行から引けば，$deg(b_{i1}) < deg(a_{11})$ を満たす $B(x)$ が得られる．

(V) 「a_{ij} を $A(x)$ の0でない最小次数の成分とする．$d(A(x)) = \deg a_{ij}$ とおく．a_{ij} で割り切れない成分 a_{kl} が存在するならば，$A(x)$ と対等な $B(x)$ が存在して，$d(B(x)) < d(A(x))$ となる．」

行，列を入れ換えても $d(A(x))$ は変化しないので，(II) より $d(A(x)) = \deg a_{11}$ と仮定してよい．第1行（第1列）に a_{11} で割り切れない成分があるならば，(IV) から (V) がいえる．したがって，第1行と第1列のすべての成分が a_{11} で割り切れると仮定する．$a_{i1} = q_i a_{11}$ とせよ．各 $i = 2, \ldots, n$ に対し，第 i 行から第1行の q_i 倍を引いて得られる行列を $C(x)$ とする．すると $c_{11} = a_{11}$，$c_{i1} = 0$ $(i = 2, \ldots, n)$ となる．$C(x)$ の (k, l) 成分は

$$c_{kl} = a_{kl} - q_k a_{1l}$$

となる．仮定より，a_{kl} は a_{11} で割り切れず，かつ a_{1l} は a_{11} で割り切れるので，c_{kl} は a_{11} で割り切れない．$C(x)$ の第1行に第 k 行を加えてできる行列を $D(x)$ とする．$D(x)$ は $A(x)$ と対等であり，$d_{1l} = c_{1l} + c_{kl} = a_{1l} + c_{kl}$ は $d_{11} = c_{11} = a_{11}$ で割り切れない．よって，(IV) により，$D(x)$ と対等で，$\deg b_{ij} < \deg d_{11} = \deg a_{11} = d(A(x))$ となるような成分 b_{ij} を持つ $B(x)$ が存在することがいえた．$B(x)$ は $A(x)$ と対等であり，$d(B(x)) \leq \deg b_{ij} < d(A(x))$ であるから，(V) が証明された．

(VI)「$A(x)$ と対等な行列 $B(x)$ が存在して，$B(x)$ のすべての成分 b_{kl} を割り切る b_{ij} を持つ．」

a_{ij} を $d(A(x)) = \deg a_{ij}$ なる $A(x)$ の成分とせよ．a_{ij} が $A(x)$ のすべての成分を割り切るならば，$B(x) = A(x)$ とすればよい．それ以外の場合は，(V) より，$d(B(x)) < d(A(x))$ となる $A(x)$ と対等な $B(x)$ が存在する．$B(x)$ がそのすべての成分を割り切る成分を持っていれば，証明は終わり．そうでない場合は，再び (V) により $d(C(x)) < d(B(x))$ となる $B(x)$ と対等な $C(x)$ が存在する．この議論を繰り返す．$d(A(x))$ が自然数であるので，この議論は有限ステップで終了する．これで (VI) が証明された．

(VII)「$A(x)$ と対等な $B(x)$ が存在して，次を満足する．

(*1) $b_{11} \neq 0$

(*2) $b_{i1} = b_{1j} = 0$ $(i > 1, j > 1)$

(*3) b_{11} は b_{ij} $(i > 1, j > 1)$ を割り切る．

(*4) b_{11} の主係数は1である．」

(VI) より，$A(x)$ と対等な $C(x)$ が存在して，そのすべての成分を割り切る成分 c_{ij} を持つ．$C(x)$ の行や列の入換えを行い，$d_{11} = c_{ij}$ となるように行列 $D(x)$ を作る．$D(x)$ は $A(x)$ と対等であり，d_{11} が $D(x)$ のすべての成分を割り切る．(III) より，$D(x)$ と対等であり，(VII*1) と (VII*2) を満たす $B(x)$ が存在する．このとき，$B(x)$ は (VII*3) も満足する．なぜならば，たとえば (III) において d_{1j} $(j > 1)$ を消去するとき，結果として得られる成分は，

$$d_{ij} - q_j d_{i1} \ (i > 1, j > 1)$$

となる．ここに，$d_{1j} = q_j d_{11}$ で，各成分は d_{11} で割り切れる（d_{i1} $(i > 1)$ を消去する場合も同様）．したがって，b_{ij} は $b_{11} = d_{11}$ で割り切れる．さらに，$B(x)$ の第1行を b_{11} の主係数で割れば，(VII*4) も満たされる．

(VIII) (VII*1)–(VII*4) を満たす $B(x)$ を半標準形と呼ぼう．$B(x)$ は以下のように書ける．

$$B(x) = \begin{bmatrix} b_{11} & 0 \\ 0 & B_1(x) \end{bmatrix}.$$

$B_1(x)$ を (VII) における $A(x)$ に置き換えて考えると，$B_1(x)$ は以下の半標準形と対等である．

$$C_1(x) = \begin{bmatrix} b_{22} & 0 \\ 0 & B_2(x) \end{bmatrix}.$$

b_{11} は $B_1(x)$ のすべての成分を割り切るので，b_{11} が $C_1(x)$ のすべての成分をも割り切る．$B_1(x)$ から $C_1(x)$ への基本変形を $B(x)$ にも行うと以下の行列を得る．

$$C(x) = \begin{bmatrix} b_{11} & & \\ & b_{22} & \\ & & B_2(x) \end{bmatrix}.$$

この手続きを繰り返すことにより，定理のスミス標準形が有限ステップで得られる．

証明終

定理3の証明でのステップ (I)–(VIII) から成るアルゴリズムを，以下 *smith* で表す．

3.2 不安定性

アルゴリズム *smith* の不安定性の原因となるいくつかの側面について議論する．まず，*smith* において，多項式 f が多項式 g を割り切るかどうかの判定が頻繁に起きる．ステップ (III)，(IV)，(VI) がそうである．この判定は g を f で割った余りが，0多項式（すべての係数が0である多項式）であるかどうかに帰着される．多項式が0多項式であるかどうかという述語は不連続点0を持ち，これがアルゴリズム *smith* に不安定性を生じさせる可能性がある．次に，多項式

から主項または主係数を選ぶという操作に，暗に不連続性が含まれている．この操作は，多項式どうしの除算を行い，剰余を計算するときにつねに必要となる．実際，(III)，(IV)，(VI)，(VII) において現れる．この操作の持つ不連続性について説明しよう．多項式の主項とは，その多項式を構成する「係数が0でない」単項式のうち，次数が最大のものをいった．したがって，主項を選ぶためには係数が0であるかどうかの判定が必要となる．前述のように，この述語は不連続点0を持つ．主係数についても同様である．詳細な説明は文献1) にある．

結論として，*smith* は不連続点0の代数的アルゴリズムであることが分かる．

次にアルゴリズム *smith* の不安定性の現象が具体的に起きる例を示す．以下，$A \in M_n(R)$ に対し，A の特性行列 $A - xE \in M_n(R[x])$ を $A(x)$ と書く．よく知られているように，A が体 R に多重固有値 α を持つとき，$smith(A(x))$ は不安定となりやすい．より詳しくいえば，$A(x)$ の最後尾の単因子 $e_n(x)$ が $(x-\alpha)^k$ ($k \geq 2$) という多重因子を持つときである．この現象は，A のジョルダン標準形の不安定性とも深く関連している．実際，任意の行列 $A \in M_n(R)$ に対し，ある「摂動方向行列」$F \in M_n(R)$ (F の成分は0か±1) が存在して，$0 < |\epsilon| \leq 1$ なる任意の ϵ に対し，$A + \epsilon F$ が n 個の相異なる固有値を持つ（実は，ほとんどの摂動方向行列がこの性質を持つ）．詳細は，文献11) を参照されたい．以降に示す例では，いずれの場合も行列 A がすべて体 R に多重固有値を持つという上記の性質を満たしている．

例1 次のような $M_3(\mathbf{Q})$ の行列 A を考える．ただし，$\mathbf{Q}$ は有理数全体の集合である．

$$A = \begin{bmatrix} 2 & 0 & 1 \\ -1 & 1 & -1 \\ -1 & 0 & 0 \end{bmatrix}.$$

このとき，$smith(A(x))$ ($A - xE$ のスミス標準形) は以下のようになる．

$$smith(A(x)) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & x-1 & 0 \\ 0 & 0 & x^2-2x+1 \end{bmatrix}.$$

ここで，摂動方向行列として次の F を考える．

$$F = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & -1 & -1 \end{bmatrix}.$$

摂動の大きさ ϵ を 0.000001 とする．摂動された行列 $A + \epsilon F$ を $\tilde{A}_\epsilon$ で表す．すると，$\tilde{A}_\epsilon$ は次のスミス標準形を持つ．

$$smith(\tilde{A}(x)) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1.0000 & 0 \\ 0 & 0 & p(x) \end{bmatrix}.$$

ここに，$p(x) = 1.0x^3 - 3.0x^2 + 3.0x - 0.999997$ である．

どんなに小さな ϵ に対しても，スミス標準形の「形」は正確なものと一致しない．言い換えると，$\epsilon \to 0$ であっても $smith(\tilde{A}_\epsilon(x)) \to smith(A(x))$ とはならない．

4. スミス標準形計算の安定化

4.1 $Int(smith)_R$

2章で説明したアルゴリズムの安定化手法を古典的なアルゴリズム *smith* に適用する☆．

$Int(smith)$ について説明しよう．まず $Int(smith)$ のデータ領域は，区間係数多項式を成分とする行列の集合である．ここで区間係数多項式とは，1変数で係数が区間である多項式をいう．区間演算が行われるのは，区間係数多項式の行列に基本変形を行うとき，および区間係数多項式どうしで除算を行い剰余を求めるときである．ゼロ書換えが行われるのは，3.2節で説明したように，剰余多項式が0かどうかを判定する述語を評価する直前，および，多項式の係数が0かどうかを判定する述語を評価する直前である．しかし，3.2節で述べたように，アルゴリズムの記述に述語が陽に現れていない場合があるため，実装上は，行列の基本変形や多項式の除算を行った直後でゼロ書換えを行うことにする☆☆．さらに $Int(smith)_R$ は，$Int(smith)$ の出力行列の各成分の各係数にゼロ書換えを施し，その結果の各区間係数の第1要素を取り出すことによって，実係数多項式行列に戻すアルゴリズムである（2章の $Int(\mathcal{A})_R$ の定義を見よ）．

定理2より以下が成り立つ．

定理4（*smith* の安定化） 行列 $A(x) = [a_{kl}] \in M_n(R[x])$ に対して，係数多項式の行列の列 $Int(A(x))_j = [Int(a_{kl})_j]$ で，成分ごとに $A(x)$ に近

☆ スミス標準形を計算するアルゴリズムとして，より効率的で実用的なものが多く存在する[12)~14)]．効率性を追求するならば，これらについても安定化を行い，その有用性を示すことは重要であろう．文献15) は，古典的なアルゴリズムの安定性について議論し，また有限精度の p 進近似計算でスミス標準形を計算する方法を提案している．これは，我々の研究に密接に関係する．

☆☆ 実装上の議論は文献2) に詳しい．

似的に収束するものを考える．すなわち各 (k,l) について，$Int(a_{kl})_j$ が a_{kl} に近似的に収束する行列の列を考える．このとき，

(1) $Int(smith)_R(Int(A(x))_j) \to smith(A(x))$ が行列成分ごとに成り立つ．

(2) ある N が存在して，$j \geq N$ ならば
$$Supp(Int(smith)_R(Int(A(x))_j) = Supp(smith(A(x)))$$
が成り立つ．ただし，行列の台（Supp）は，各成分の台の行列である．

4.2 例

$Int(smith)_R$ の効果を見るために，具体的な実行例をあげる．実験は，$smith$ と $Int(smith)_R$ を Maple V Release 3[16]で実装し，計算機は HP9000/735 を使って行った．Maple は linear algebra というパッケージに "smith" という組込み関数を持っているが，これは浮動小数点や代数的数の係数を受けつけない．

例 2 例 1 を再掲する．

$$A = \begin{bmatrix} 2 & 0 & 1 \\ -1 & 1 & -1 \\ -1 & 0 & 0 \end{bmatrix}.$$

$$F = \begin{bmatrix} -1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & -1 & -1 \end{bmatrix}.$$

これより

$$A(x) = \begin{bmatrix} 2-x & 0 & 1 \\ -1 & 1-x & -1 \\ -1 & 0 & -x \end{bmatrix}$$

となる．$A(x)$ に近似的に収束する区間係数多項式の行列の列として，たとえば "$[A+10^{-j}F, 10^{-j}]-x[E,0]$" を考える．すなわち，$Int(A(x))_j$ として，

$$\begin{bmatrix} [2-e_j,e_j]-x & [0,e_j] & [1,e_j] \\ [-1+e_j,e_j] & [1,e_j]-x & [-1+e_j,e_j] \\ [-1,e_j] & [-e_j,e_j] & [-e_j,e_j]-x \end{bmatrix}$$

を考える．ここに $e_j = 10^{-j}$ である．x の係数は $-[1,0]$ であるが，スペースの都合で -1 としている．$Int(smith)$ に $\{Int(A(x))_j\}_j$ を入力すると，次のようになる．$j=2$ に対しては，

$$Int(smith)_R(Int(A(x))_2) = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & 1.0 & 0 \\ 0 & 0 & 1.0x^3 - 3.0x^2 + 3.0x - 1.0 \end{bmatrix}$$

である．この結果は $smith(A(x))$ と台が一致しない．しかし $j=3$ に対しては，

$$Int(smith)_R(Int(A(x))_3) = \begin{bmatrix} 1.00 & 0 & 0 \\ 0 & 1.00x - 1.00 & 0 \\ 0 & 0 & p(x) \end{bmatrix}$$

となる．ここに，$p(x) = 1.00x^2 - 2.00x + 0.997$ である．この結果は，$smith(A(x))$ と台が一致している．また $j = 4, \ldots, 10$ で各係数の収束性も実験で観測した．

次に摂動方向行列 F をあらかじめ指定せず，A を単に浮動小数点近似する場合について考える．

例 3

$$A = \begin{bmatrix} \frac{107}{153} & \frac{202}{153} & \frac{8}{153} & -\frac{61}{153} \\ -\frac{29}{153} & \frac{421}{306} & \frac{25}{153} & -\frac{37}{306} \\ -\frac{71}{306} & \frac{577}{612} & \frac{325}{306} & -\frac{175}{612} \\ -\frac{22}{51} & \frac{19}{51} & \frac{26}{51} & \frac{44}{51} \end{bmatrix}$$

のとき，

$$smith(A(x)) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & (x-1)^2 & 0 \\ 0 & 0 & 0 & (x-1)^2 \end{bmatrix}$$

となる．

A の 5 桁の浮動小数点近似 $\tilde{A}$ を考える．

$$\tilde{A} = \begin{bmatrix} 0.69935 & 1.3203 & 0.052288 & -0.39869 \\ -0.18954 & 1.3758 & 0.16340 & -0.12092 \\ -0.23203 & 0.94281 & 1.0621 & -0.28595 \\ -0.43137 & 0.37255 & 0.50980 & 0.86275 \end{bmatrix}.$$

単に $smith$ に $\tilde{A}(x)$ を入力すると，正確な形の行列を返さない．

$$smith(\tilde{A}(x)) = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & p(x) \end{bmatrix}.$$

ここに，$p(x) = 1.0000x^4 - 3.9598x^3 + 5.9237x^2 - 3.9677x + 1.0040$ である．

具体的には以下のようにして，正確でない計算結果が導かれる．定理3で説明したアルゴリズムの(VIII)により以下の半標準形が導かれる．

$$\tilde{B}(x) = \begin{bmatrix} 1.3203 & 0 & 0 & 0 \\ 0 & -0.0012504 & 0 & 0 \\ 0 & 0 & p_1(x) & p_2(x) \\ 0 & 0 & p_3(x) & p_4(x) \end{bmatrix}.$$

ここで，$p_1(x) = 241.49 - 483.00\,x + 241.50\,x^2$，$p_2(x) = -173.21 + 346.42\,x - 173.21\,x^2$，$p_3(x) = 799.78 - 1599.5\,x + 799.76\,x^2$，$p_4(x) = -571.12 + 1142.2\,x - 571.10\,x^2$ である．

この行列が計算されるまでは，厳密に定理3のアルゴリズムを実行したときと同じ形の行列が計算過程の途中においても計算される．次に定理3の(VIII)により，$\tilde{B}$ の部分行列

$$\tilde{B}_1 = \begin{bmatrix} p_1(x) & p_2(x) \\ p_3(x) & p_4(x) \end{bmatrix}$$

に対して，(I)，(II)，(III) ... の操作が順に実行される．(I)，(II) は $\tilde{B}_1$ を変化させない．(III) においては $p_1(x) = \tilde{B}_{3,3}(x)$ が $p_2(x) = \tilde{B}_{3,4}(x)$，$p_3(x) = \tilde{B}_{4,3}(x)$ を割り切るかの判定が行われる．つまり除算の剰余の0判定が行われる．浮動小数点近似を行っていないアルゴリズムにおいては，割り切る，という判定を行う．そして定理3の(III)により，正確なスミス標準形を導く．しかし，浮動小数点近似においては，割り切れない行列要素がある，という判定になり，定理3の(IV)の操作に入ってしまう．

桁数を高めて $j = 1000$ まで計算したが，正しい形の行列は得られなかった．$Int(smith)_R$ はどうか？ $A(x)$ に近似的に収束する区間係数多項式の列 $\{Int(A(x))_j\}_j$ として "$[float_j(A(x)), error_j(A(x))]$" を考える．すなわち行列成分は，それに対応する $A(x)$ の行列成分の j 桁での区間（浮動小数点近似とその誤差の上界）である．すると，

$$Int(smith)_R(Int(A(x))_2) = \begin{bmatrix} 1.0 & 0 & 0 & 0 \\ 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0x^2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

となる．まだ，精度桁が足りない．

$$Int(smith)_R(Int(A(x))_3) = \begin{bmatrix} 1.00 & 0 & 0 & 0 \\ 0 & 1.00 & 0 & 0 \\ 0 & 0 & p_1(x) & 0 \\ 0 & 0 & 0 & p_2(x) \end{bmatrix},$$

$p_1(x) = 1.00x^2 - 2.00x + 1.01$,
$p_2(x) = 1.00x^2 - 2.00x + 0.989$.

$j = 3$ とすると正確な形の行列を得る．

$j=4,\ldots,10$ に対して，$\{Int(smith)_R(Int(A(x))_j)\}_j$ が $smith(A(x))$ に成分ごとに台収束することを実験により観測した．

最後に無理数を含む場合を考える．

例4 A として $M_4(\mathbf{Q}(\sqrt{2}))$ の要素を考える．ただし，$\mathbf{Q}(\sqrt{2})$ は集合 $\{a + b\sqrt{2} \mid a, b \in \mathbf{Q}\}$ である．

$$A = (3608\sqrt{2} - 8545)^{-1} \times \left(\begin{bmatrix} 6322 & -186 & -408 & 959 \\ -1788 & 6844 & -816 & 1918 \\ -9834 & -2046 & 2728 & 10549 \\ -5364 & -1116 & -2448 & 12970 \end{bmatrix} + \sqrt{2} \begin{bmatrix} -8545 & 252 & -24 & 40 \\ 0 & -8041 & -48 & -80 \\ 0 & 2772 & -8809 & -440 \\ 0 & 1512 & -144 & -8785 \end{bmatrix} \right).$$

このとき，

$$smith(A(x)) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & x - \sqrt{2} & 0 & 0 \\ 0 & 0 & x - \sqrt{2} & 0 \\ 0 & 0 & 0 & \left(x - \sqrt{2}\right)^2 \end{bmatrix}$$

である．

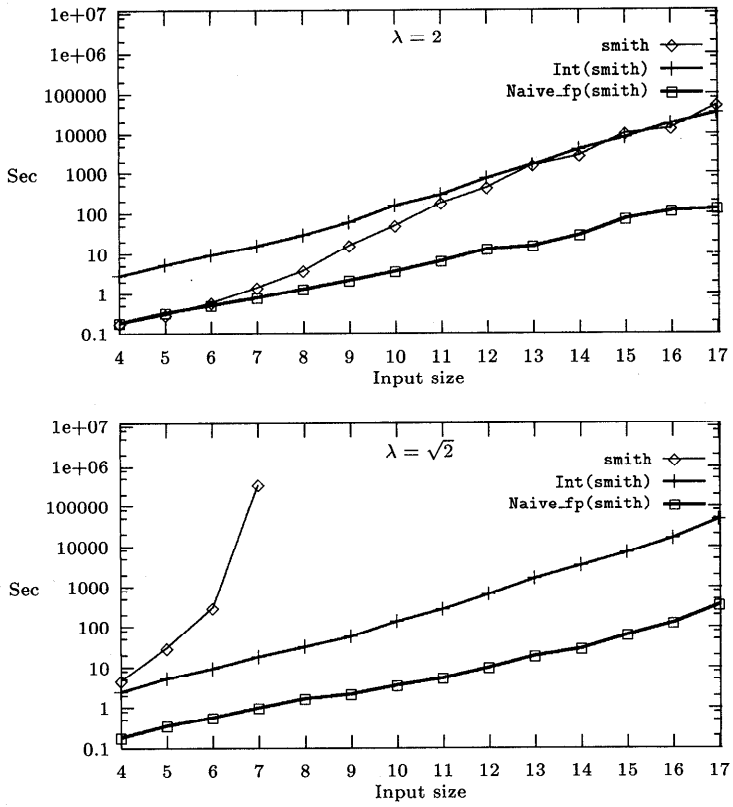

図1 計算時間
Fig. 1 Timings.

同様に，単に $A(x)$ の浮動小数点5桁による近似 $\tilde{A}(x)$ を *smith* に入力すると，

$$smith(\tilde{A}(x)) = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & p(x) \end{bmatrix}.$$

ここに，$p(x) = 1.0000x^3 - 7.8284x^2 + 16.141x - 9.9985$ である．

これは不安定である．この場合の不安定性の原因も，例3と同様の形の行列が導かれた後の定理3の(III)において，割り切るかどうかの判定が正しく行われていないことによる．

$j = 1000$ でも安定な結果が得られなかった．ところが，$Int(smith)_R$ によれば，$j = 5$（かつ5より大きいいくつかの j）に対して，以下のような正確な結果が得られた．

$$Int(smith)_R([Int(A(x))_5]) = \begin{bmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & p_1(x) & 0 & 0 \\ 0 & 0 & p_2(x) & 0 \\ 0 & 0 & 0 & p_3(x) \end{bmatrix}.$$

ここに，$p_1(x) = 1.0000x - 1.4144$，$p_2(x) = 1.0000x - 1.4143$，$p_3(x) = 1.0000x^2 - 2.8282x + 1.999$ である．

4.3 実験結果

前節では，2〜3の例で $Int(smith)_R$ の安定性を観察してきた．この節では，より大きな行列の大量な例について，その実験結果の一部を次の項目ごとにグラフで示す．

- 計算時間（素朴な浮動小数点計算と厳密計算との比較）
- 提案手法に対する厳密計算の計算時間の比
- $Int(smith)_R$ の1回の実行中，ゼロ書換えによって実際に0に置き換わった区間の個数
- 正確な台を与える最小の桁数（定理4の N）

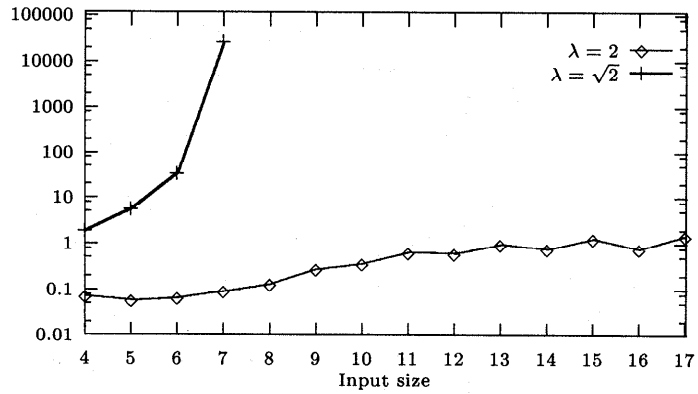

図2 （smith の計算時間）÷（Int(smith) の計算時間）の比
Fig. 2 Ratio of time(smith)/time(Int(smith)).

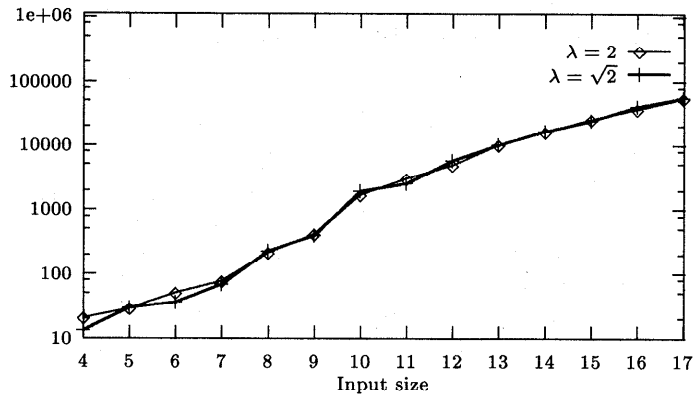

図3 0に書き換えられた区間係数の個数
Fig. 3 Number of interval-coefficients rewritten to zero.

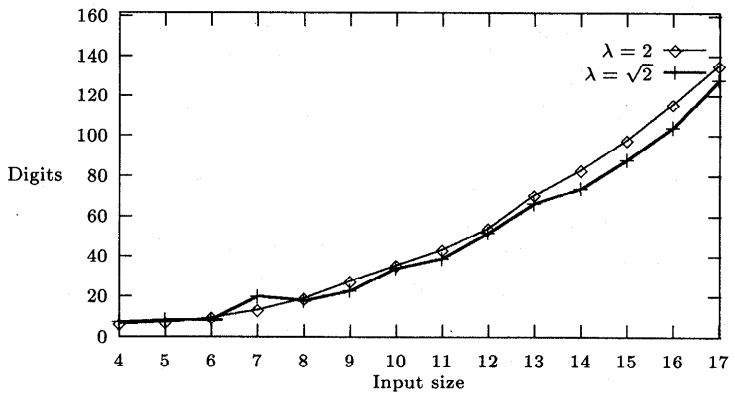

図4 正確な台を与える最小精度桁
Fig. 4 Minimum precision giving the correct support.

入力のサンプルは，多重固有値を持つようにランダムに生成した．より詳しくいうと，J を同一の固有値に対するジョルダンブロックが複数個あるようなジョルダン標準形，P をランダムな正則整数行列として $A = P^{-1}JP$ を入力とした．P の生成には，Maple のランダム関数を使った．簡単のため，次の2つの場合について実験報告をする．(1) A のすべての固有値が2の場合，(2) A のすべての固有値が $\sqrt{2}$ の場合．(2)

の場合には，P も $\sqrt{2}$ をランダムに含むようにした．

図について説明する．すべての図で，横軸は，$n \times n$ の入力行列のサイズ n を表している．縦軸は，最後の図を除き，底10の対数目盛で書いてある．

図1 では，縦軸は計算CPU時間（秒）を表す．(1) 固有値 λ が2の場合，(2) 固有値 λ が $\sqrt{2}$ の場合の両方について記している．`smith`，`Int(smith)`，`Naive_fp(smith)` はそれぞれ，*smith* の厳密計算，$Int(smith)_R$ の計算，*smith* の（0の判定を行わない）素朴な浮動小数点計算を示す．`Int(smith)` の精度は，正確な台を与える最小の桁数を使った．`Naive_fp(smith)` に対してもそれと同じ桁数を使って計算した．桁数については，図4を見よ．

図2 は，図1から導かれるもので，`smith` の `Int(smith)` に対する計算時間の比を表す．

図3 は，図1での `Int(smith)` の1回の実行において，C_j は0ではないが $|C_j| \leq \gamma_j$ であるため $[C_j, \gamma_j]$ が $[0,0]$ に書き換えられたその $[C_j, \gamma_j]$ の個数を示す．

図4 は，正確な台が最初に出現する桁数，すなわち，定理4における N を示す．

考察：図1を見れば，予期していたとおり，素朴な浮動小数点計算 `Naive_fp(smith)` が最も速いことが分かる．しかし，$j = 1{,}000$ 桁の精度で計算を行っても正確な台は得ることはできなかった．ゼロ書換えを行わない素朴な区間解析法でも `Naive_fp(smith)` と同様であった．すなわちゼロ書換えなしでは1,000桁の精度で計算をしても正確な台を得ることはできない．一方，提案手法 `Int(smith)` では，図4で示すように比較的小さな桁数の計算で安定な結果が得られる．したがって，ゼロ書換えが実際に効力を発揮していることが分かる．このことは，図3によっても確認される．大まかにいえば，計算時間は，`smith`，`Int(smith)`，`Naive_fp(smith)` はどれも，対数目盛上で行列サイズとともに線形に増加しているように見える．すなわち通常の目盛上では指数関数的に増加している．図1と図2からいえることは，有理数係数の場合，`Int(smith)` は `smith` より必ずしも速くはない（少なくとも中くらいのサイズの行列に対しては）のに対し，係数が無理数を含むような複雑な場合には，`Int(smith)` は `smith` より，はるかに速いことである．さらに，`Int(smith)` は，`Naive_fp(smith)` より区間演算とゼロ書換えの計算コストの分だけ計算が遅くなるが，はるかに安定である．

5. おわりに

スミス標準形を，有限精度の浮動小数点計算を用いても安定に計算できるための手法を提案し，実際にその効果を実証した．本論は，数値計算で行われている便法を否定するものではない．場合に応じて，本手法をそれと比較しながら研究を進めるべきであろう．

今後の課題としては，行列が与えられたとき，本手法がスミス標準形の正確な台を与えるのにどれだけの桁数が必要であるかを理論的に見積もることがあげられる．また，安定化手法をより効率的で実用的なアルゴリズムに適用し，スミス標準形の高速で信頼性の高い計算方式を実現することも重要である．

参考文献

1) Shirayanagi, K. and Sweedler, M.: A Theory of Stabilizing Algebraic Algorithms, Technical Report 95-28, Mathematical Sciences Institute, Cornell University (1995).
2) 白柳 潔：アルゴリズムの安定化理論，数式処理，Vol.5, No.2, pp.2–21 (1997).
3) 白柳 潔：不安定なアルゴリズムを安定化する，情報処理，Vol.39, No.2, pp.111–115 (Feb.1998).
4) Alefeld, G. and Herzberger, J.: Introduction to Interval Computations, *Computer Science and Applied Mathematics*, Academic Press (1983).
5) Shirayanagi, K.: An Algorithm to Compute Floating Point Gröbner Bases, *Mathematical Computation with Maple V: Ideas and Applications*, Lee, T. (Ed.), pp.95–106, Birkhäuser (1993).
6) Shirayanagi, K.: Floating Point Gröbner Bases, *Mathematics and Computers in Simulation*, Vol.42, No.4-6, pp.509–528 (1996).
7) 白柳 潔，関川 浩：ゼロ書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，Vol.J80-A, No.5, pp.791–802 (1997).
8) Lang, S.: *Algebra*, 3rd edition, Addison-Wesley (1993).
9) Lang, S.: *Introduction to Linear Algebra*, Addison-Wesley (1965).
10) 杉浦光夫：Jordan標準形と単因子論II，岩波書店 (1977).
11) Kato, T.: *Perturbation Theory for Linear Operators*, 2nd edition, Springer-Verlag (1976).
12) Kannan, R.: Polynomial-time algorithms for solving systems of linear equations over polynomials, *Theoretical Computer Science*, Vol.39, pp.69–88 (1985).
13) Kaltofen, E., Krishnamoorthy, M.S. and Saunders, B.D.: Fast parallel computation of Hermite and Smith forms of polynomial matrices, *SIAM Journal on Algebraic and Discrete Methods*, Vol.8, pp.683–690 (1987).

14) Villard, G.: Computation of the Smith normal form of polynomial matrices, *Proc. ISSAC '93*, Kiev, Ukraine, pp.208–217 (1993).
15) Ramachandran, V.: Exact reduction of a polynomial matrix to the Smith normal form, *IEEE Trans. Automatic Control*, Vol.AC-24, No.4, pp.638–641 (1979).
16) Char, B.W., Geddes, K.O., Gonnet, G.H., Leong, B.L., Monagan, M.B., and Watt, S.M.: *First Leaves: A Tutorial Introduction to Maple V*, Springer-Verlag (1992).

(平成10年 3 月16日受付)
(平成10年11月 9 日採録)

白柳 潔（正会員）

1982年東京大学理学部数学科卒業．1984年同大学院修士課程修了．同年日本電信電話公社入社．博士（数理科学）．1992年から1993年まで米国コーネル大学客員研究員．コンピュータ囲碁の研究を経て，数式処理の研究に従事．現在，NTTコミュニケーション科学基礎研究所主任研究員．京都大学大学院情報学研究科客員助教授．情報処理学会，日本数式処理学会，日本数学会，ACM，AMS（米国数学会）各会員．

新妻 弘崇

1993年大阪大学工学部応用物理学科卒業．1995年同大学院修士課程修了．現在，奈良先端科学技術大学院大学情報科学研究科博士後期課程在学中．