数理解析研究所講究録
941 巻 1996 年 185-193

23.

近似計算による代数的数の符号判定について

関川 浩 (NTT CS 研)

23.1 はじめに

数式処理のアルゴリズムに対して, 入力を単純に近似しただけでは得られた出力が正確な答の近似になっているとは限らない. しかし, 白柳-Sweedler の代数的アルゴリズムの安定化理論 ([8]) により, 区間演算やゼロ書き換えなどの工夫をすれば, 近似計算でもある有限精度から先では常に妥当な近似を出力するようにできる. ただし, その精度 (lucky precision) がどれ位であるかを決定する問題は一般に難しい. そこで, 少なくとも出力結果が妥当か否かを判定したい, というのがこの研究の出発点である.

数式処理の多くのアルゴリズムにおいて, 近似計算を用いた場合の不安定性の原因はゼロ判定であった. 本稿では, もう少し一般に, 符号判定を扱う. ここでいう符号判定とは, 入力された実数を四則演算して得られた数の符号判定 (正か負かゼロかの判定) のことである. ゼロ判定は数式処理のほとんどのアルゴリズムで使われており, 符号判定も Sturm のアルゴリズム, 計算幾何学の凸包や Voronoi 図の構成などで本質的な部分である.

もし, 入力される数がすべて有理数ならば, 正確演算を用いれば符号判定には何の問題もない. しかし, 代数的数の符号判定を行おうとするといろいろな問題が生じる. 以下, 本稿では代数的数を対象とする.

代数的数の符号は代数的な情報のみからは決まらず, 代数的数の実数体への埋め込み方によって決まる. すなわち, 何らかの数値的な情報が必要である. 通常の符号判定法では, 数値的な情報の計算に比べて代数的な情報の計算 (最小多項式の計算など) に非常に負荷がかかるのに対し, 本稿で提案する

手法では, 代数的な情報の計算の負荷を軽くしてその分を数値的な情報の計算で補っている.

なお, 白柳-Sweedler の安定化理論によっても代数的数の符号判定は可能である. この場合, どんな近似精度でも出力が得られ, ある有限精度から先では常に正しく符号判定を行うのに対し, 本稿で述べる符号判定法は, 精度の低いうちは出力が得られず, 出力が得られるようになればそれは必ず正解であるという, ある意味で相補的なものである.

本稿の方法は, 「有限の精度で表現された数値に有限回の演算を施して得られる値の符号を判定することは, やはりある有限の精度の計算で厳密にできる」という一般的な原理 ([9] 3.6 節) によっているといってよい. この原理を生かした種々の例については, [9] および, そこに挙げられている文献を参照されたい.

23.2 従来の符号判定法

代数的数を扱う場合, まず問題になるのはその表現方法である ([4]). 固定した代数体 $K = \mathbf{Q}(\theta)$ の中で話を進める場合, 代数的数$\alpha \in K$を表現する方法はいくつかあるが, 符号判定を行う立場からは, $\alpha = \sum_{i=0}^{d-1} a_i\theta^i$ ($a_i \in \mathbf{Q}$, $d = [K : \mathbf{Q}]$), なる表現が便利である.

Kの原始元である実代数的数θが, 最小多項式 $f(x)$ と, $f(x) = 0$ の根のうちθのみを含む区間 $I = (r, t]$ で与えられているとき, $\alpha = \sum_{i=0}^{d-1} a_i\theta^i \in K$の符号は以下のようにして求めることができる ([6]).

1. $A(x) = \sum_{i=0}^{d-1} a_i x^i$とおく.
2. $A = 0$ のときは符号は 0. $\deg A = 0$ のときは, $A(\in \mathbf{Q})$ の符号.
3. $\deg A > 0$ のときは Iの部分区間 $I^* = (r^*, t^*]$ で, θを含み, $A(x) = 0$ の根を含まないものを求める (二分法などによる). $A(t^*)$ の符号がαの符号である.

この方法を適用するためには, 扱う代数的数をすべて含む拡大体 Kの原始元とその最小多項式を求めておく必要がある. これは, $[K : \mathbf{Q}]$ は大きいが, 実際に符号判定する代数的数の次数がそれほど大きくない場合 (たとえば, 二次元の凸包の構成で, 各入力点の座標は低次の代数的数だが, すべての入力点の座標を含む代数体の拡大次数が非常に大きい場合など), 現実的ではない.

代数体を固定しない場合の代数的数の表現方法として, 代数的数αをその最小多項式 fと, αを含む区間 Iとの対 (f, I) で表現する方法がある. ただし, 区間 I内で $f(x) = 0$ の根はαのみとなるようにとっておく. fは最小多項式であるから, αが 0 であることと $f = x$ であることは同値である. αが 0 でない場合, 必要なら二分法などを用いることにより, Iの部分区間でαを含み 0 を含まないものを求めればαの符号が決まる.

この場合, 代数的数の間で演算を行うごとに計算結果の最小多項式を求める必要がある. たとえば, α, βがそれぞれ, (f, I), (g, J) で表されているとき, $\alpha + \beta$の最小多項式は次のようにして求められる ([6], [4]). まず, $f(x - y)$ と $g(y)$ を, yの多項式と見たときの終結式 $h(x)$ を計算する. もし, $h(x)$ が

$\mathbf{Q}$ 上既約ならば $h(x)$ が $\alpha+\beta$ の最小多項式であり, 可約ならば $h(x)$ の適当な既約因子が $\alpha+\beta$ の最小多項式である. ただし, I, J を十分に小さくとっておかないと, $h(x)=0$ の根のうちどれが $\alpha+\beta$ に対応するのかがわからず, I, J の部分区間をとる必要が生じる場合がある. さらに問題となるのは, 最小多項式の次数がすぐに巨大となることである.

23.3 代数的情報つき区間演算

本稿では, 代数体を固定しない場合にも使え, しかも, 最小多項式の計算はしない符号判定法を与える. 以下, 話を簡単にするため, 加減乗のみを考え除法は除外する. 23.2節の代数体を固定しない場合の方法は, 根の近似値である「数値的情報」と最小多項式という「代数的情報」の対の間で計算を行うものであった. ここで提案する手法は, 「数値的情報」と「代数的情報」の対の間で計算を行う, という点は同じだか, 「代数的情報」として, 最小多項式の代わりに, より計算の簡単な最小多項式の次数と measure を使うものである.

本稿の方法を適用するために measure m に要請される条件は以下の三つである.

1. 任意の代数的数αに対して, $m(\alpha)>0$.
2. ある計算可能な函数 Bが存在して, 任意の代数的数αに対して, $m(\alpha)\leq M$ ならば, $\alpha=0$ または $0<B(M)\leq|\alpha|$.
3. ある計算可能な函数 $B_+, B_-, B_\times$が存在して, α, βがそれぞれ $m(\alpha)\leq M$, $m(\beta)\leq N$ なる高々d 次, e 次の代数的数のとき,

$$\begin{aligned} m(\alpha+\beta) &\leq B_+(M,N,d,e),\\ m(\alpha-\beta) &\leq B_-(M,N,d,e),\\ m(\alpha\beta) &\leq B_\times(M,N,d,e). \end{aligned}$$

このような measure として, αの最小多項式 (整数係数) の種々のノルムを使うことが可能である. たとえば, $P(x)=\sum_{i=0}^{d}a_ix^i\in\mathbf{C}[x]$ に対して,

$$\begin{aligned} \|P\| &= \left(\sum_{i=0}^{d}|a_i|^2\right)^{1/2},\\ L(P) &= \sum_{i=0}^{d}|a_i|,\\ H(P) &= \max_{0\leq i\leq d}\{|a_i|\}, \end{aligned}$$

などがそうである.

23.3.1 Mahler の measure

ここでは, measure として Mahler によるもの ([7]) を使う.

定義 1 複素数係数の一変数多項式 $P(x) = \sum_{i=0}^{d} a_i x^i = a_d \prod_{i=1}^{d}(x - \alpha_i)$ に対して, Pの *measure* $M(P)$ を次の式で定義する.

$$M(P) = |a_d| \prod_{i=1}^{d} \max\{1, |\alpha_i|\}.$$

代数的数αに対して, αの *measure* $M(\alpha)$ を次の式で定義する.

$$M(\alpha) = M(P).$$

ただし, Pはαの整数係数の原始的な最小多項式である.

命題 1 *Mahler の measure* Mは以下の性質を満たす.

1. 任意の代数的数αに対して, $M(\alpha) \geq 1$.
2. $M(\alpha) \leq M$ならば, $\alpha = 0$ または$\frac{1}{M} \leq |\alpha| \leq M$.
3. α, βがそれぞれ高々d 次, e 次の代数的数のとき,

$$\begin{aligned} M(\alpha \pm \beta) &\leq 2^{de} M(\alpha)^e M(\beta)^d, \\ M(\alpha\beta) &\leq M(\alpha)^e M(\beta)^d. \end{aligned}$$

証明は [2] を参照されたい. この命題より Mahler の measure は前述の要請を満たすことがわかる.

注意 1 $M(P)$ はそれほど大きくない. 次の不等式 *(Landau の不等式)* が成り立つ *([5])*.

$$M(P) \leq \|P\|.$$

なお, $H(P) \leq \|P\| \leq L(P)$ である. Hを使うと, $H(\alpha \pm \beta)$, $H(\alpha\beta)$ を $H(\alpha)$, $H(\beta)$ を使って上から評価する式が複雑で値の増加も激しい.

23.3.2 代数的情報つき区間

数値的情報と代数的情報の対である代数的情報つき区間を以下のように定義する.

定義 2

1. αを高々m 次の代数的数とする. Iをαを含む閉区間 *(αの他の共役を含んでいてもよい)*, $M \geq M(\alpha)$ としたとき, 三つ組 (I, m, M) をαに対する代数的情報つき区間と定義する.

2. 代数的情報つき区間の間の演算は以下のように定義する.

$$(I, m, M) \pm (J, n, N) = (I \pm J, mn, 2^{mn} M^n N^m),$$
$$(I, m, M) \times (J, n, N) = (I \times J, mn, M^n N^m).$$

ただし, $I \pm J$, $I \times J$は区間演算 *([1])* による.

数値的情報を表す区間に浮動小数を用いた場合, 代数的情報つき区間の精度がμとは, 浮動小数の仮数部分をμ桁にとることと定義する. このとき, 以下の定理が成り立つのは明らかである.

定理 1 $\alpha_1, \alpha_2, \ldots, \alpha_k$を代数的数とする. これらの数の間の加減乗算により得られた代数的数をαとする. また, $\alpha_1, \alpha_2, \ldots, \alpha_k$をある精度で代数的情報つき区間に変換し, αを得たのと同じ計算過程で定義 *2*の計算により得られた代数的情報つき区間を $([a, b], m, M)$ とする. このとき,

1. $a \leq 0 \leq b$ かつ $\max\{-a, b\} < \frac{1}{M}$ ならば, $\alpha = 0$ である.
2. 逆に, $\alpha = 0$ ならば,

$$a \leq 0 \leq b$$

は精度によらず成り立ち, さらに, ある有限精度から先でつねに,

$$\max\{-a, b\} < \frac{1}{M}$$

が成り立つ.

注意 2 数値的情報を表す区間に浮動小数以外のものを用いても, 精度をうまく定義すれば定理 *1*と同様のことが成り立つ.

23.3.3 実際の計算

数値的情報を表す区間に浮動小数を用いた場合の実際の計算は以下のようになる. ただし, 入力される代数的数は, その最小多項式が与えられていて, 必要に応じていくらでも精度の高い根の近似値が計算できるものとする.

1. 適当な精度μを設定する.
2. 入力された代数的数を精度μで代数的情報つき区間に変換する.
3. 代数的情報つき区間の間で計算を行う.
4. 計算結果の数値的情報部分の区間が 0 を含まなければ, 代数的情報によらず符号が決まり (正か負), 終了.
5. 計算結果の数値的情報部分の区間が 0 を含めば, 定理1を適用する. 0 と判定できれば, 終了. 判定できなければ精度μを上げて 2 に戻る.

注意 3 代数的情報は数値的情報と独立に決まるから, 上記の手続きは有限ステップで終了する.

23.4 計算例

ここで計算例を二つ示す. 実装は HP9000/735 上の Maple V Release 3 ([3]) による. なお, 区間演算には, Maple の Share Library にある区間演算パッケージ intpak (by Connell, A. E. and Corless, R. M.) を使用した. 例は以下の通りである.

例 1　$\sqrt{2}\cdot\sqrt{3}-\frac{211462}{86329}$ (>0). ただし, $\frac{211462}{86329}$は$\sqrt{6}$の連分数展開を 10 段で打ち切ったもの.

$$\frac{211462}{86329}=2+\underbrace{\frac{1}{2}+\frac{1}{4}+\frac{1}{2}+\frac{1}{4}+\cdots+\frac{1}{4}}_{10}.$$

数値的情報部分を 10 進 11 桁の浮動小数で計算する. $\sqrt{2}$, $\sqrt{3}$, $\frac{211462}{86329}$は, それぞれ,

$$([1.4142135623, 1.4142135624], 2, 2),$$
$$([1.7320508075, 1.7320508076], 2, 3),$$
$$([2.4494897427, 2.4494897428], 1, 211462),$$

に変換される. $\sqrt{2}\cdot\sqrt{3}$に対応する計算結果は,

$$([2.4494897425, 2.4494897430], 4, 36),$$

$(\sqrt{2}\cdot\sqrt{3})-\frac{211462}{86329}$に対応する計算結果は,

$$([-0.30000000001\times10^{-9}, 0.30000000001\times10^{-9}], 4, 11517330385170975589263 36),$$

となる. 区間 $[-0.30000000001\times10^{-9}, 0.30000000001\times10^{-9}]$ は 0 を含むが,

$$\begin{aligned} 0.30000000001\times10^{-9} &> \frac{1}{11517330385170975589263 36} \\ &= 0.868\cdots\times10^{-24} \end{aligned}$$

であり, 符号判定ができない.

よって精度を上げ, 数値的情報部分を 10 進 12 桁の浮動小数で計算する. $\sqrt{2}$, $\sqrt{3}$, $\frac{211462}{86329}$は, それぞれ,

$$([1.41421356237, 1.41421356238], 2, 2),$$
$$([1.73205080756, 1.73205080757], 2, 3),$$
$$([2.44948974272, 2.44948974273], 1, 211462),$$

に変換される. $\sqrt{2}\cdot\sqrt{3}$に対応する計算結果は,

$$([2.44948974276, 2.44948974281], 4, 36),$$

$(\sqrt{2}\cdot\sqrt{3}) - \frac{211462}{86329}$に対応する計算結果は,

$$([0.299999999999 \times 10^{-10}, 0.900000000001 \times 10^{-10}], 4, 1151733038517097558926336),$$

である. 区間 $[0.299999999999 \times 10^{-10}, 0.900000000001 \times 10^{-10}]$ は正の領域に入るので, 計算結果は正と判定できる.

例 2 $\sqrt{2}\cdot\sqrt{3} - \sqrt{6}\ (= 0)$.

数値的情報部分を 10 進 10 桁の浮動小数で計算する. $\sqrt{2}$, $\sqrt{3}$, $\sqrt{6}$はそれぞれ,

$$([1.414213562, 1.414213563], 2, 2),$$
$$([1.732050807, 1.732050808], 2, 3),$$
$$([2.449489742, 2.449489743], 2, 6),$$

に変換される. $\sqrt{2}\cdot\sqrt{3}$に対応する計算結果は,

$$([2.449489740, 2.449489745], 4, 36),$$

$(\sqrt{2}\cdot\sqrt{3}) - \sqrt{6}$に対応する計算結果は,

$$([-0.3000000001 \times 10^{-8}, 0.3000000001 \times 10^{-8}], 8, 429981696),$$

となる. 区間 $[-0.3000000001 \times 10^{-8}, 0.3000000001 \times 10^{-8}]$ は 0 を含が,

$$0.3000000001 \times 10^{-8} > \frac{1}{429981696} = 0.2325\cdots \times 10^{-8}$$

となり, 符号判定ができない.

よって精度を上げ, 数値的情報部分を 10 進 11 桁の浮動小数で計算する. $\sqrt{2}$, $\sqrt{3}$, $\sqrt{6}$はそれぞれ,

$$([1.4142135623, 1.4142135624], 2, 2),$$
$$([1.7320508075, 1.7320508076], 2, 3),$$
$$([2.4494897427, 2.4494897428], 2, 6),$$

に変換される. $\sqrt{2}\cdot\sqrt{3}$に対応する計算結果は,

$$([2.4494897425, 2.4494897430], 4, 36),$$

$(\sqrt{2}\cdot\sqrt{3}) - \sqrt{6}$に対応する計算結果は,

$$([-0.30000000001 \times 10^{-9}, 0.30000000001 \times 10^{-9}], 8, 429981696),$$

となる. 区間 $[-0.30000000001 \times 10^{-9}, 0.30000000001 \times 10^{-9}]$ は 0 を含むが,

$$0.30000000001 \times 10^{-9} < \frac{1}{429981696} = 0.2325\cdots \times 10^{-8}$$

となり, 計算結果は 0 と判定できる.

23.5 おわりに

本稿では, 最小多項式の計算をせずに代数的数の符号判定を行える代数的情報つき区間演算を提案した. 通常の区間演算のみでは, 真にゼロとなる場合の判定ができないが, 本提案手法によればゼロ判定も正確に行うことができる. ただし, 最小多項式を計算する必要がなくなっても, 演算を繰り返すうちに代数的情報部分に巨大な数が現れるのは防げない.

したがって, 今後の課題として, 本手法を実際のアルゴリズム (Sturm のアルゴリズム, 凸包の構成など) に適用して有効性を確認することが挙げられる. また, 符号判定に必要な精度の見積りが事前にできれば応用上便利であろう.

さらに, 白柳-Sweedler の代数的アルゴリズムの安定化理論との融合ができれば望ましいと考えている.

参考文献

[1]Alefeld, G. and Herzberger, J., *Introduction to Interval Computations, Computer Science and Applied Mathematics, Academic Press* (1983).

[2]Cerlienco, L., Mignotte, M., and Piras, F., Computing the Measure of a Polynomial, *J. Symbolic Computation* **4** (1987), 21–33.

[3]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).

[4]Cohen, H., *A Course in Computational Algebraic Number Theory, Springer-Verlag* (1993).

[5]Landau, E., Sur quelques théorèmes de M. Petrovic relatifs aux zéros des fonctions analytiques, *Bull. Soc. Math. France* **33** (1905), 251–261.

[6]Loos, R., Computing in Algebraic Extensions, *Computer Algebra Symbolic and Algebraic Computation (Ed. B. Buchberger, G. E. Collins, and R. Loos), Springer-Verlag* (1983), 173–187.

[7]Mahler, K., An Application of Jensen's Formulae to Polynomials, *Mathematica* 7 (1960), 98–100.

[8]Shirayanagi, K. and Sweedler, M., A Theory of Stabilizing Algebraic Algorithms, *Technical Report* 95-28, *Mathematical Sciences Institute, Cornell University* (1995).

[9]杉原厚吉, 計算幾何工学, 培風館 (1994).