お話：数値解析 第7回
収束の加速法(中編の２)

長田直樹

1　はじめに

リチャードソン補外は、数列 $\{s_\nu\}$ が既知の定数のべき λ_j^ν $(1 > |\lambda_1| > |\lambda_2| > \ldots > 0)$ による漸近展開

$$s_\nu \sim s + \sum_{j=1}^{\infty} c_j \lambda_j^\nu$$

を持つとき、$\lambda_1, \lambda_2, \ldots$ の項を順に消去することによる加速法であった (2008 年 9 月号)。

今回は、既知の漸近列によって漸近展開される数列を加速する一般化リチャードソン補外の話をする。この加速法は、前回話した e 変換の拡張になっており、2 つの行列式の比で表される。

収束の加速法は当初 2 ないし 3 回で話すつもりだったが、予定外のライプニッツ級数を巡る加速 (交代級数の加速) の話を加えたため、長くなってしまった。今回の話は前回の続きである。

2　行列式についての諸定理

2 つの行列式の比を再帰的に計算する方法を与えるシルヴェスターの定理とハンケル行列式から始める。これらの話題は、線形代数のテキストでは通常扱われない。

2.1　シルヴェスターの定理

定理 1 (シルヴェスターの定理) $A = (a_{ij})$ を $n \times n$ 行列とし、A の最初の r 行と r 列よりなる行列を

$$A_r = \begin{pmatrix} a_{11} & \cdots & a_{1r} \\ & \cdots & \\ a_{r1} & \cdots & a_{rr} \end{pmatrix}$$

とする。A_r が正則なとき、$r+1$ 次の小行列式 B_{ij} $(i, j = r+1, \ldots, n)$ と、B_{ij} を成分とする行列式 Δ を

$$B_{ij} = \begin{vmatrix} & & & a_{1j} \\ & A_r & & \vdots \\ & & & a_{rj} \\ a_{i1} & \cdots & a_{ir} & a_{ij} \end{vmatrix},$$

$$\Delta = \begin{vmatrix} B_{r+1\,r+1} & \cdots & B_{r+1\,n} \\ \vdots & & \vdots \\ B_{n\,r+1} & \cdots & B_{n\,n} \end{vmatrix}$$

により定義すると、次の式が成り立つ。

$$\Delta = (\det A_r)^{n-r-1} \det A$$

証明　証明の概略を見ることの出来る入手可能なテキストには [4, p.33][9, p.278] がある。　□

系 1 $A = (a_{ij})$ を $n \times n$ 行列とし、r 次の小行列式を

$$D\begin{pmatrix} i_1 & \cdots & i_r \\ j_1 & \cdots & j_r \end{pmatrix} = \begin{vmatrix} a_{i_1 j_1} & \cdots & a_{i_1 j_r} \\ \vdots & & \vdots \\ a_{i_r j_1} & \cdots & a_{i_r j_r} \end{vmatrix}$$

により表す。このとき、次の恒等式が成立する。

$$\begin{aligned} & D\begin{pmatrix} 1 & \cdots & n \\ 1 & \cdots & n \end{pmatrix} D\begin{pmatrix} 2 & \cdots & n-1 \\ 2 & \cdots & n-1 \end{pmatrix} \\ = \; & D\begin{pmatrix} 1 & \cdots & n-1 \\ 1 & \cdots & n-1 \end{pmatrix} D\begin{pmatrix} 2 & \cdots & n \\ 2 & \cdots & n \end{pmatrix} \\ & -D\begin{pmatrix} 1 & \cdots & n-1 \\ 2 & \cdots & n \end{pmatrix} D\begin{pmatrix} 2 & \cdots & n \\ 1 & \cdots & n-1 \end{pmatrix} \end{aligned}$$

証明 行列

$$C = \begin{pmatrix} a_{22} & \cdots & a_{2\,n-1} & a_{21} & a_{2n} \\ \vdots & & \vdots & \vdots & \vdots \\ a_{n-1\,2} & \cdots & a_{n-1\,n-1} & a_{n-1\,1} & a_{n-1\,n} \\ a_{12} & \cdots & a_{1\,n-1} & a_{11} & a_{1n} \\ a_{n2} & \cdots & a_{n\,n-1} & a_{n1} & a_{nn} \end{pmatrix}$$

に $r = n-2$ として定理 1 を適用する。 □

2.2 ハンケル行列式

数列 $u = \{u_\nu\}$ に対し、行列式

$$\begin{aligned} H_0^{(\nu)}(u) &= 1 \\ H_k^{(\nu)}(u) &= \begin{vmatrix} u_\nu & u_{\nu+1} & \dots & u_{\nu+k-1} \\ u_{\nu+1} & u_{\nu+2} & \dots & u_{\nu+k} \\ & & \cdots & \\ u_{\nu+k-1} & u_{\nu+k} & \dots & u_{\nu+2k-2} \end{vmatrix} \end{aligned}$$

を**ハンケル行列式**という。下付きの添字 k は行列式の次数、上付きの添字 ν は $(1,1)$ 要素に表れる数列 u_ν の添字である。右上から左下への斜めに等しい要素が並んでいる。

ハンケル行列式に関する次の 2 つの漸化式を次節で利用する。

補題 1

$$\begin{aligned} & H_k^{(\nu)}(\Delta u) H_k^{(\nu+1)}(\Delta u) \\ = \; & H_k^{(\nu)}(\Delta^2 u) H_k^{(\nu+1)}(u) - H_{k-1}^{(\nu+1)}(\Delta^2 u) H_{k+1}^{(\nu)}(u) \end{aligned}$$

証明 $k+2$ 次の行列

$$A = \begin{pmatrix} 1 & 1 & \dots & 1 & 0 \\ u_\nu & u_{\nu+1} & \dots & u_{\nu+k} & 1 \\ & & \cdots & & \\ u_{\nu+k} & u_{\nu+k+1} & \dots & u_{\nu+2k} & 1 \end{pmatrix}$$

に系 1 を適用する。 □

補題 2

$$\begin{aligned} & H_k^{(\nu+1)}(\Delta u) H_{k+1}^{(\nu)}(\Delta u) \\ = \; & H_k^{(\nu)}(\Delta^2 u) H_{k+1}^{(\nu+1)}(u) - H_k^{(\nu+1)}(\Delta^2 u) H_{k+1}^{(\nu)}(u) \end{aligned}$$

証明 $k+3$ 次の行列

$$A = \begin{pmatrix} 0 & 1 & \dots & 1 & 0 \\ 1 & u_\nu & \dots & u_{\nu+k} & 1 \\ & & \cdots & & \\ 0 & u_{\nu+k+1} & \dots & u_{\nu+2k+1} & 1 \end{pmatrix}$$

に系 1 を適用する。 □

3 ϵ 算法の導出

前回やり残した ϵ 算法の導出を行う。シャンクス e 変換 $e_k(s_\nu)$ は、ハンケル行列式を用いて表すことができる。

$$\begin{aligned} & e_k(s_\nu) \\ = \; & \frac{\begin{vmatrix} s_\nu & s_{\nu+1} & \cdots & s_{\nu+k} \\ \Delta s_\nu & \Delta s_{\nu+1} & \cdots & \Delta s_{\nu+k} \\ & & \cdots & \\ \Delta s_{\nu+k-1} & \Delta s_{\nu+k} & \cdots & \Delta s_{\nu+2k-1} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ \Delta s_\nu & \Delta s_{\nu+1} & \cdots & \Delta s_{\nu+k} \\ & & \cdots & \\ \Delta s_{\nu+k-1} & \Delta s_{\nu+k} & \cdots & \Delta s_{\nu+2k-1} \end{vmatrix}} \\ = \; & \frac{\begin{vmatrix} s_\nu & s_{\nu+1} & \cdots & s_{\nu+k} \\ s_{\nu+1} & s_{\nu+2} & \cdots & s_{\nu+k+1} \\ & & \cdots & \\ s_{\nu+k} & s_{\nu+k+1} & \cdots & s_{\nu+2k} \end{vmatrix}}{\begin{vmatrix} 1 & 0 & \cdots & 0 \\ \Delta s_\nu & \Delta^2 s_\nu & \cdots & \Delta^2 s_{\nu+k-1} \\ & & \cdots & \\ \Delta s_{\nu+k-1} & \Delta^2 s_{\nu+k-1} & \cdots & \Delta^2 s_{\nu+2k-2} \end{vmatrix}} \\ = \; & \frac{H_{k+1}^{(\nu)}(s)}{H_k^{(\nu)}(\Delta^2 s)} \end{aligned} \tag{1}$$

$e_0(s_\nu) = s_\nu$ なので、(1) は $k=0$ のときも成立する。

定理 2 (前号の定理 3 の再掲) $\nu = 1, 2, \dots$ に対し、

$$\begin{aligned} \epsilon_{-1}^{(\nu)} &= 0, \\ \epsilon_{2k}^{(\nu)} &= e_k(s_\nu), \quad k = 0, 1, \dots \\ \epsilon_{2k+1}^{(\nu)} &= \frac{1}{e_k(\Delta s_\nu)}, \quad k = 0, 1, \dots \end{aligned}$$

とおいたとき、漸化式

$$\epsilon_{l+1}^{(\nu)} = \epsilon_{l-1}^{(\nu+1)} + \frac{1}{\epsilon_l^{(\nu+1)} - \epsilon_l^{(\nu)}}, \quad l = 0, 1, \dots$$

が成立する。ただし、分母は0にならないものとする。

証明 (1) より、$k=0,1,\ldots$ に対し、

$$\epsilon_{2k}^{(\nu)}=\frac{H_{k+1}^{(\nu)}(s)}{H_k^{(\nu)}(\Delta^2 s)},\quad \epsilon_{2k+1}^{(\nu)}=\frac{H_k^{(\nu)}(\Delta^3 s)}{H_{k+1}^{(\nu)}(\Delta s)}$$

である。

$l=0$ のとき、

$$\epsilon_1^{(\nu)}=\frac{1}{e_0(\Delta s_\nu)}=\frac{1}{s_{\nu+1}-s_\nu}=\frac{1}{\epsilon_0^{(\nu+1)}-\epsilon_0^{(\nu)}}$$

$l=2k$ のとき

$$\begin{aligned}&\epsilon_{2k-1}^{(\nu+1)}+\frac{1}{\epsilon_{2k}^{(\nu+1)}-\epsilon_{2k}^{(\nu)}}\\ =\;&\frac{H_{k-1}^{(\nu+1)}(\Delta^3 s)}{H_k^{(\nu+1)}(\Delta s)}+\frac{1}{\dfrac{H_{k+1}^{(\nu+1)}(s)}{H_k^{(\nu+1)}(\Delta^2 s)}-\dfrac{H_{k+1}^{(\nu)}(s)}{H_k^{(\nu)}(\Delta^2 s)}}\end{aligned}$$

(補題 2 より)

$$=\frac{H_{k-1}^{(\nu+1)}(\Delta^3 s)}{H_k^{(\nu+1)}(\Delta s)}+\frac{H_k^{(\nu+1)}(\Delta^2 s)H_k^{(\nu)}(\Delta^2 s)}{H_k^{(\nu+1)}(\Delta s)H_{k+1}^{(\nu)}(\Delta s)}$$

(補題 1 より)

$$=\frac{H_k^{(\nu)}(\Delta^3 s)}{H_{k+1}^{(\nu)}(\Delta s)}=\frac{1}{e_k(\Delta s_\nu)}=\epsilon_{2k+1}^{(\nu)}$$

$l=2k+1$ のときも同様である。 □

4 一般化リチャードソン補外

数列 $\{s_\nu\}$ が

$$\begin{aligned}s_\nu&=s+\sum_{j=1}^{\infty}c_j\lambda_j^\nu,\qquad(2)\\ 1&>|\lambda_1|>|\lambda_2|>\ldots>0\end{aligned}$$

を満たしているものとし、$s,c_1,c_2,\ldots$ は未知の定数で、$\lambda_1,\lambda_2,\ldots$ は既知の定数とする。$T_k^{(\nu)},c_1,\ldots,c_k$ を未知数と考えると連立 1 次方程式

$$\begin{cases}s_\nu=T_k^{(\nu)}+c_1\lambda_1^\nu+\cdots+c_k\lambda_k^\nu\\ \quad\cdots\\ s_{\nu+k}=T_k^{(\nu)}+c_1\lambda_1^{\nu+k}+\cdots+c_k\lambda_k^{\nu+k}\end{cases}\qquad(3)$$

は唯一の解を持つ。一方、$T_k^{(\nu)}$ は

$$\begin{aligned}T_1^{(\nu)}&=\frac{s_{\nu+1}-\lambda_1 s_\nu}{1-\lambda_1}\\ T_l^{(\nu)}&=\frac{T_{l-1}^{(\nu+1)}-\lambda_l T_{l-1}^{(\nu)}}{1-\lambda_l},\quad l=2,\ldots,k\end{aligned}$$

により与えられるので、(2) にリチャードソン補外を適用して得られる数列である。

(2)(3) を一般化し、数列 $\{s_\nu\}$ は既知の関数列または漸近列 $\{g_j(\nu)\}$ によって

$$s_\nu=s+\sum_{j=1}^{\infty}c_j g_j(\nu)$$

または

$$s_\nu\sim s+\sum_{j=1}^{\infty}c_j g_j(\nu)$$

と表され、$T_k^{(\nu)}$ は

$$\begin{cases}s_\nu=T_k^{(\nu)}+c_1g_1(\nu)+\cdots+c_kg_k(\nu)\\ \quad\cdots\\ s_{\nu+k}=T_k^{(\nu)}+c_1g_1(\nu+k)+\cdots+c_kg_k(\nu+k)\end{cases}$$

を満たすものとする。$T_k^{(\nu)}$ はクラメールの公式により

$$T_k^{(\nu)}=\frac{\begin{vmatrix}s_\nu & s_{\nu+1} & \cdots & s_{\nu+k}\\ g_1(\nu) & g_1(\nu+1) & \cdots & g_1(\nu+k)\\ & & \cdots & \\ g_k(\nu) & g_k(\nu+1) & \cdots & g_k(\nu+k)\end{vmatrix}}{\begin{vmatrix}1 & 1 & \cdots & 1\\ g_1(\nu) & g_1(\nu+1) & \cdots & g_1(\nu+k)\\ & & \cdots & \\ g_k(\nu) & g_k(\nu+1) & \cdots & g_k(\nu+k)\end{vmatrix}}\qquad(4)$$

となる。(行と列を入れ替えてある。)

数列 $\{s_\nu\}$ に数列 $\{T_k^{(\nu)}\}$ を対応させる変換は**一般化リチャードソン補外**あるいは一般補外算法と呼ばれる。一般化リチャードソン補外の範疇に属す数列変換には次のものがある。

例 1 $g_j(\nu)=\lambda_j^\nu$ で λ_j $(j=1,2,\ldots)$ が既知の場合は、$T_k^{(\nu)}$ はリチャードソン補外である。

例 2 $g_j(\nu)=\Delta s_{\nu+j-1}$ のときは $T_k^{(\nu)}$ はシャンクス e 変換 $e_k(s_\nu)$ である。したがって、ϵ 算法で実装化できる。

例 1 や例 2 のように $g_j(\nu)$ が特別な形をしているときは、リチャードソン補外や ϵ 算法のような効率のよい算法が知られている。一般の関数 $g_j(\nu)$ に対しても、(4) の分母と分子の規則性を用いた漸化式を用いた算法 (再帰的な算法) が考えられている。一つは次節で取り上げる E 算法、もう一つは FS 算法である。

5 E 算法

5.1 E 算法の導出

2 次元配列 $E_k^{(\nu)}$ と 3 次元の補助配列 $g_{k,j}^{(\nu)}$ を次のように定義する。

$$
\begin{aligned}
E_0^{(\nu)} &= s_\nu, \quad \nu = 1, 2, \ldots, && (5)\\
g_{0,j}^{(\nu)} &= g_j(\nu), \quad j = 1, 2, \ldots; \nu = 1, 2, \ldots, && (6)\\
E_k^{(\nu)} &= \frac{E_{k-1}^{(\nu)} g_{k-1,k}^{(\nu+1)} - E_{k-1}^{(\nu+1)} g_{k-1,k}^{(\nu)}}{g_{k-1,k}^{(\nu+1)} - g_{k-1,k}^{(\nu)}}, && (7)\\
& \quad k = 1, 2, \ldots; \nu = 1, 2, \ldots,\\
g_{k,j}^{(\nu)} &= \frac{g_{k-1,j}^{(\nu)} g_{k-1,k}^{(\nu+1)} - g_{k-1,j}^{(\nu+1)} g_{k-1,k}^{(\nu)}}{g_{k-1,k}^{(\nu+1)} - g_{k-1,k}^{(\nu)}}, && (8)\\
& \quad j = k+1, k+2, \ldots.
\end{aligned}
$$

次の定理が基本的である。

定理 3 (ブレザンスキー) $G_{k,j}^{(\nu)}$, $N_k^{(\nu)}$, と $D_k^{(\nu)}$ を次のように定義する。

$$
\begin{aligned}
G_{k,j}^{(\nu)} &= \begin{vmatrix} g_j(\nu) & \cdots & g_j(\nu+k) \\ g_1(\nu) & \cdots & g_1(\nu+k) \\ & \cdots & \\ g_k(\nu) & \cdots & g_k(\nu+k) \end{vmatrix},\\
N_k^{(\nu)} &= \begin{vmatrix} s_\nu & \cdots & s_{\nu+k} \\ g_1(\nu) & \cdots & g_1(\nu+k) \\ & \cdots & \\ g_k(\nu) & \cdots & g_k(\nu+k) \end{vmatrix},\\
D_k^{(\nu)} &= \begin{vmatrix} 1 & \cdots & 1 \\ g_1(\nu) & \cdots & g_1(\nu+k) \\ & \cdots & \\ g_k(\nu) & \cdots & g_k(\nu+k) \end{vmatrix}
\end{aligned}
$$

そのとき、$\nu = 1, 2, \ldots; k = 1, 2, \ldots$ に対し

$$
\begin{aligned}
g_{k,j}^{(\nu)} &= G_{k,j}^{(\nu)} / D_k^{(\nu)}, \quad j > k,\\
E_k^{(\nu)} &= N_k^{(\nu)} / D_k^{(\nu)} = T_k^{(\nu)},\\
& \quad \nu = 1, 2, \ldots; k = 1, 2, \ldots.
\end{aligned}
$$

証明 $g_{k,j}^{(\nu)} = G_{k,j}^{(\nu)} / D_k^{(\nu)}$ を k についての数学的帰納法により証明する。(6)(8) により、

$$
\begin{aligned}
g_{1,j}^{(\nu)} &= \frac{g_{0,j}^{(\nu)} g_{0,1}^{(\nu+1)} - g_{0,j}^{(\nu+1)} g_{0,1}^{(\nu)}}{g_{0,1}^{(\nu+1)} - g_{0,1}^{(\nu)}}\\
&= \frac{g_j(\nu) g_1(\nu+1) - g_j(\nu+1) g_1(\nu)}{g_1(\nu+1) - g_1(\nu)} = \frac{G_{1,j}^{(\nu)}}{D_1^{(\nu)}}
\end{aligned}
$$

が成立する。

$g_{k-1,j}^{(\nu)} = G_{k-1,j}^{(\nu)} / D_{k-1}^{(\nu)}$ が成り立つとすると (8) より

$$
\begin{aligned}
g_{k,j}^{(\nu)} &= \frac{g_{k-1,j}^{(\nu)} g_{k-1,k}^{(\nu+1)} - g_{k-1,j}^{(\nu+1)} g_{k-1,k}^{(\nu)}}{g_{k-1,k}^{(\nu+1)} - g_{k-1,k}^{(\nu)}}\\
&= \frac{\dfrac{G_{k-1,j}^{(\nu)}}{D_{k-1}^{(\nu)}} \dfrac{G_{k-1,k}^{(\nu+1)}}{D_{k-1}^{(\nu+1)}} - \dfrac{G_{k-1,j}^{(\nu+1)}}{D_{k-1}^{(\nu+1)}} \dfrac{G_{k-1,k}^{(\nu)}}{D_{k-1}^{(\nu)}}}{\dfrac{G_{k-1,k}^{(\nu+1)}}{D_{k-1}^{(\nu+1)}} - \dfrac{G_{k-1,k}^{(\nu)}}{D_{k-1}^{(\nu)}}}\\
&= \frac{G_{k-1,j}^{(\nu)} G_{k-1,k}^{(\nu+1)} - G_{k-1,j}^{(\nu+1)} G_{k-1,k}^{(\nu)}}{G_{k-1,k}^{(\nu+1)} D_{k-1}^{(\nu)} - G_{k-1,k}^{(\nu)} D_{k-1}^{(\nu+1)}},
\end{aligned}
$$

系 1 より、

$$
= \frac{(-1)^{2k-3} G_{k,j}^{(\nu)} G_{k-2,k-1}^{(\nu+1)}}{(-1)^{2k-3} D_k^{(\nu)} G_{k-2,k-1}^{(\nu+1)}} = \frac{G_{k,j}^{(\nu)}}{D_k^{(\nu)}}
$$

$E_{k-1}^{(\nu)}$ についても同様である。 □

一般化リチャードソン補外の $T_k^{(\nu)}$ を (5)-(8) で計算する算法を ***E* 算法**という。(7)(8) は、

$$
\left.
\begin{aligned}
c_k^{(\nu)} &= \frac{g_{k-1,k}^{(\nu)}}{g_{k-1,k}^{(\nu+1)} - g_{k-1,k}^{(\nu)}},\\
E_k^{(\nu)} &= E_{k-1}^{(\nu)} - c_k^{(\nu)} (E_{k-1}^{(\nu+1)} - E_{k-1}^{(\nu)}),\\
g_{k,j}^{(\nu)} &= g_{k-1,j}^{(\nu)} - c_k^{(\nu)} (g_{k-1,j}^{(\nu+1)} - g_{k-1,j}^{(\nu)})
\end{aligned}
\right\} \quad (9)
$$

と変形することにより、演算回数を 4 割程度削減できる。

5.2 E 算法小史

E 算法は 1975 年に C. シュナイダー [7]、1979 年に T. ホビエ [3]、次いで 1980 年に C. ブレザンスキー [1] が独立に同じ算法を導いた。シュナイダーとホビエは連続量に対するリチャードソン補外の拡張として扱い、ブレザンスキーは数列に対する汎用的補外法として扱った。E 算法の命名者はブレザンスキーで、E は e 変換 (シャンクス)、ϵ 算法 (ウィン) の名称を踏まえている。

1987 年に W.F. フォードと A. シディ [2] は、E 算法よりも演算回数が (式 (9) による E 算法より $1 \sim 3$ 割程度) 少ない算法を提案した。今日では彼らの名前を取って FS 算法と呼ばれている。筆者 [6] は、E 算法と FS 算法は一方から他方が導けるという意味で数学的に同値であることを示した。

5.3 ライプニッツ級数への適用

前回の主題であったライプニッツ級数の 4 倍

$$s_\nu = \sum_{i=1}^{\nu} \frac{4(-1)^{i-1}}{2i-1}$$

は、前回の補題 1 より

$$s_\nu \sim \pi + (-1)^\nu \sum_{j=1}^{\infty} c_j \nu^{1-2j}$$

を満たすので、

$$g_j(\nu+l) = (-1)^{\nu+l-1}(\nu+l)^{1-2j}$$

とおくと E 算法が適用できる。結果は表 1 のようである。

表 1: ライプニッツ級数に E 算法を適用

ν	$E_0^{(\nu)}$	$E_1^{(\nu-1)}$	$E_2^{(\nu-2)}$
1	*4.0000000000*		
2	*2.6666666667*	3.1*111111111*	
3	3.*4666666667*	3.14*66666667*	3.14*31111111*
4	*2.8952380952*	3.14*01360544*	3.1414*421769*
5	3.*3396825397*	3.14*21516755*	3.1416*181287*
10	3.*0418396189*	3.1415*626106*	3.141592*5347*

ν	$E_3^{(\nu-3)}$	$E_4^{(\nu-4)}$	$E_5^{(\nu-5)}$
4	3.1415*169053*		
5	3.1415*985104*	3.1415*964311*	
10	3.141592654	3.1415926535	3.1415926535

(表には載せてないが)10 項までの部分和から $E_7^{(3)} = 3.1415926535879$ が得られ、円周率と 12 桁一致している。

5.3 節で用いた C 言語による E 算法のプログラムは

`http://www.cis.twcu.ac.jp/~osada/rikei/`

においておく。

6 レヴィン変換

1979 年 D.A. スミスと W.F. フォード [8] は 11 の異なる加速法を 20 の異なる級数に適用し比較を行った。加速法には一般オイラー変換、ϵ 算法が含まれる。全域にわたって最良の方法はレヴィン u 変換というのがその結論であった。線型収束に対してはレヴィン t 変換も同等であるとした。

連載の第 1 回 (2008 年 5 月号) では、線型収束を

$$\lim_{\nu\to\infty} \frac{s_{\nu+1}-s}{s_\nu - s} = \rho, \quad 0 < |\rho| < 1$$

により定義したが、[8] では

$$\lim_{\nu\to\infty} \frac{s_{\nu+1}-s}{s_\nu - s} = \lim_{\nu\to\infty} \frac{\Delta s_{\nu+1}}{\Delta s_\nu} = \rho, \quad \rho \neq 1, 0$$

により定義している。そのため、ライプニッツ級数などの交代級数も線型収束となる。

レヴィン変換は 1973 年 D. レヴィン [5] によって提案された 3 通りの変換である。論文 [8] により、レヴィン変換は加速法の研究者と利用者の注目を集めることとなった。

数列 $\{s_\nu\}$ に対するレヴィン変換は、

$$T_k^{(\nu)} = \frac{\begin{vmatrix} s_\nu & s_{\nu+1} & \cdots & s_{\nu+k} \\ R_\nu & R_{\nu+1} & \cdots & R_{\nu+k} \\ & & \cdots & \\ R_\nu\nu^{1-k} & R_{\nu+1}(\nu+1)^{1-k} & \cdots & R_{\nu+k}(\nu+k)^{1-k} \end{vmatrix}}{\begin{vmatrix} 1 & 1 & \cdots & 1 \\ R_\nu & R_{\nu+1} & \cdots & R_{\nu+k} \\ & & \cdots & \\ R_\nu\nu^{1-k} & R_{\nu+1}(\nu+1)^{1-k} & \cdots & R_{\nu+k}(\nu+k)^{1-k} \end{vmatrix}}$$

により定義される。$R_\nu = \nu\Delta s_{\nu-1}$ とおくのが u 変換, $R_\nu = \Delta s_\nu \Delta s_{\nu-1}/\Delta^2 s_{\nu-1}$ が v 変換, $R_\nu = \Delta s_\nu$ が t 変換である。レヴィン u 変換では $T_k^{(k)}$ の値を求めるのに、E 算法同様、$s_1, \ldots, s_{\nu+k}$ が必要であるのに対し、レヴィン t 変換と v 変換では、$s_1, \ldots, s_{\nu+k+1}$ と 1 項先までが必要であるので、加速効率の比較には注意が必要である。

レヴィンは分母と分子の行列式をファンデルモンドの行列式に帰着させることにより

$$T_k^{(\nu)} = \frac{\displaystyle\sum_{j=0}^{k} (-1)^j \begin{pmatrix} k \\ j \end{pmatrix} (j+\nu)^{k-1} \frac{s_{j+\nu}}{R_{j+\nu}}}{\displaystyle\sum_{j=0}^{k} (-1)^j \begin{pmatrix} k \\ j \end{pmatrix} (j+\nu)^{k-1} \frac{1}{R_{j+\nu}}} \qquad (10)$$

を導いた。$g_j(\nu) = R_\nu\nu^{1-j}$ とおくとレヴィン変換は一般化リチャードソン補外の一つであるので E 算法で計算できる。

スミスとフォードによって、線型収束に対しては u 変換と同様の効果があると認定された t 変換を取り上げる。ライプニッツ級数の 4 倍にレヴィン t 変換を適用した結果は表 2 である。$T_1^{(\nu-1)}$ はエイトケン Δ^2 法なので、ϵ 算法の結果 (前号の表 3) と一致 ($\epsilon_2^{(\nu-2)} = T_1^{(\nu-1)}$) している。

表 2: ライプニッツ級数にレヴィン t 変換を適用

ν	$T_0^{(\nu)}$	$T_1^{(\nu-1)}$	$T_2^{(\nu-2)}$
1	*4.0000000000*		
2	*2.6666666667*	3.1*666666667*	
3	3.*4666666667*	3.1*333333333*	3.1*393939394*
4	*2.8952380952*	3.14*52380952*	3.141*9913420*
5	3.*3396825397*	3.1*396825397*	3.141*4843415*
9	3.*2523659347*	3.141*2548236*	3.1415*887771*
10	3.*0418396189*	3.141*8396189*	3.14159*48209*

ν	$T_3^{(\nu-4)}$	$T_4^{(\nu-5)}$	$T_5^{(\nu-6)}$
4	3.141*7027417*		
5	3.1415*817564*	3.14159*13847*	
9	3.1415926*366*	3.1415926*602*	3.141592654*7*
10	3.141592657*4*	3.141592651*3*	3.141592653*4*

(表には載せてないが)10 項までの部分和から $T_7^{(2)} = 3.14159265357\mathit{33}$ が得られ、円周率と 11 桁一致している。

7 まとめ

前回と今回の 2 回にわたりライプニッツ級数を巡る加速法の話をした。10 項までの部分和からそれぞれの加速法で円周率と何桁一致するかをまとめておく。計算は倍精度 (10 進 16 桁弱) によっている。

表 3: ライプニッツ級数に対する加速法の比較

方法	データ	一致桁数
シャンカラ	前号表 1	7.79
遅延オイラー変換	前号表 2	5.14
e 変換 (ϵ 算法)	前号表 3	7.33
一般化リチャードソン補外	本号表 1†	12.22
レヴィン t 変換	本号表 2†	11.28

† は表を説明する本文中

数列の漸近展開の漸近列を用いた一般化リチャードソン補外 (アルゴリズムは E 算法) が最も一致桁数が大きい。(シャンカラ以外の) 各加速法は、ライプニッツ級数に類似の漸近展開を持つ交代級数に対しても、同様の結果を与える。

参考文献

[1] C. Brezinski, A general extrapolation algorithm, Numer. Math. 35(1980), 175–187.

[2] W.F. Ford and A. Sidi, An algorithm for a generalization of the Richardson Extrapolation process, SIAM J. Numer. Anal. 24(1987), 1212–1232.

[3] T. Håvie, Generalized Neville type extrapolation schemes, BIT 19(1979), 204–213.

[4] 伊理正夫、一般線形代数、岩波書店、2003

[5] D. Levin, Development of non-linear transformations for improving convergence of sequences, Intern. J. Computer Math. Sec.B, 3(1973) 371–388.

[6] N. Osada, The E-algorithm and the Ford-Sidi algorithm, J. Comput. Appl. Math. 122(2000), 223-230.

[7] C. Schneider, Vereinfachte Rekursionen zur Richardson Extrapolation in Spezialfällen, Numer. Math. 24(1975), 177-184.

[8] D.A. Smith and W.F. Ford, Acceleration of linear and logarithmic convergence, SIAM J. Numer. Anal. 16(1979), 223–240.

[9] 高木貞治、代数学講義改訂新版、共立出版、1965

[10] P. Wynn, On a Device for computing the $e_m(S_n)$ transformations, MTAC 10(1956), 91–96.

(おさだ なおき/東京女子大学)