誤差限界式を応用した多項式の実根、および複素根を求める数値計算における誤差改善法

田口　功

要　約

多項式の根をコンピュータを使って計算した場合、種々の誤差が生じ、それによって実根が複素根に変わってしまうことがある。ここではこれに対処する方法を考察し、多項式の相異なる実根、重複した実根、および複素根として計算された根の精度改善法を、誤差限界式を応用して考察した。ここでは、一度求められた根を用い、根の平行移動を繰り返して誤差限界値をほぼ0に近づけ、それによって根の計算精度を改善する方法を提案した。最後に、シミュレーションを行ない本研究の有用性を示した。

1．はじめに

多項式の根を計算機を用いて計算する場合誤差が生じる。文献［2］における誤差限界式を応用し、個々の根に対し、一度数値計算を行ない、その求められた根の値を用い、平行移動によって、任意の一つの根の精度を向上させる方法を提案する。ここでは、反復誤差改善計算とでも言うべき方法について述べる。ここで、任意の一つの根の精度を改善し計算するとき他の根の精度は保証されないが、任意の根の精度は反復平行移動計算によって著しく向上することを述べる。本論文においては相異なる実根、さらに重根を含む場合、および複素根をもつ多項式の単根、重根を含む場合の誤差改善について述べる。

2．1　多項式の根と誤差限界式の関係

本稿では計算機の演算桁数が十分大きく、根の計算過程における丸め誤差は無視できるとし、相異なる実根、複素根、および重複根が存在する場合の誤差限界式を求める。最初に、多項式の係数の変化が根に与える影響について考える。多項式

$$f(z) = z^n + a_n z^{n-1} + a_{n-1} z^{n-2} + \cdots + a_2 z + a_1 \qquad (2\text{-}1)$$

の各係数がわずかに変化した多項式を

$$\bar{f}(z) = z^n + \bar{a}_n z^{n-1} + \bar{a}_{n-1} z^{n-2} + \cdots + \bar{a}_2 z + \bar{a}_1 \qquad (2\text{-}2)$$

とする。$f(z)$ の任意の根を ξ、これに対応する $\bar{f}(z)$ の根を $\bar{\xi}$ とする。ここで、$\bar{a}_{n+1} =$

$a_{n+1}=1$ とし、$\Delta a_{n+1}=0$ とする

$$\Delta a_{k+1}=\bar{a}_{k+1}-a_{k+1} \tag{2-3}$$

$$\Delta\xi=\bar{\xi}-\xi \tag{2-4}$$

とおく（k＝0～n）。仮定から、

$$\begin{aligned}\bar{f}(\bar{\xi})&=0\\&=\bar{f}(\bar{\xi})-f(\xi)\\&=\sum_{k=0}^{n}\{(a_{k+1}+\Delta a_{k+1})(\xi+\Delta\xi)^k-a_{k+1}\xi^k\}\\&=\sum_{k=0}^{n}\{a_{k+1}(\xi+\Delta\xi)^k+\Delta a_{k+1}(\xi+\Delta\xi)^k-a_{k+1}\xi^k\}\\&=f(\xi+\Delta\xi)-f(\xi)+\sum_{k=0}^{n}\Delta a_{k+1}\xi^k\\&=\sum_{k=0}^{n}f^{(k)}(\xi)\Delta\xi^k/k!+\sum_{k=0}^{n}\Delta a_{k+1}\xi^k\end{aligned} \tag{2-5}$$

となる。（2-5）式の右辺第２項は、係数の摂動による多項式 f(x) の ξ における摂動と考えられる。そこで、

$$\Delta f(\xi)\triangleq\sum_{k=0}^{n}\Delta a_{k+1}\xi^k \tag{2-6}$$

と定義すると、多項式の根 ξ が受ける全体の摂動 $\Delta\xi$ は、

$$\sum_{k=0}^{n}f^{(k)}(\xi)\Delta\xi^k/k!+\Delta f(\xi)=0 \tag{2-7}$$

を満足しなければならない。多項式の各係数が２進法 t 桁に丸められ、入力されると考えれば、

$$|\Delta a_{k+1}|\leqq 2^{-t}|a_{k+1}| \tag{2-8}$$

となるから、関数値の誤差限界は、

$$|\Delta f(\xi)|\leqq 2^{-t}\sum_{k=0}^{n}|a_{k+1}||\xi|^k \tag{2-9}$$

となる。また（2-7）式において $\Delta\xi$ の２次以上の項を無視すると、

$$f'(\xi)\Delta\xi/1!=-\Delta f(\xi) \tag{2-10}$$

となり、（2-9）式を用いると、$\Delta\xi=-\Delta f(\xi)/f'(\xi)$ であるから相異なる単根の場合には、

$$|\Delta\xi|\leqq 2^{-t}\sum_{k=0}^{n}|a_{k+1}||\xi|^k/|f'(\xi)| \quad \text{（単根の場合）} \tag{2-11-1}$$

の関係が成立し、m重根を持つ根に対しては $f'(\xi)=0$, $f^{(2)}(\xi)=0$, …, $f^{(m-1)}(\xi)=0$ に注意し整理すると、

$$|\Delta\xi|\leqq 2^{-t/m}\{\sum_{k=0}^{n}|a_{k+1}||\xi|^k/|f^{(m)}(\xi)|\}^{1/m} \tag{2-11-2}$$

の関係が成立する。ここで、（2-11）式は各根に対する誤差限界を示す。したがって、各根の有効桁数は、単根の場合には、

$$\sum_{k=0}^{n}|a_{k+1}||\xi|^k/|f'(\xi)| \tag{2-12-1}$$

重根の場合には、

$$\{\sum_{k=0}^{n}|a_{k+1}||\xi|^k/|f^{(m)}(\xi)|\}^{1/m} \tag{2-12-2}$$

によって大きく変化する事となる（k＝0～n）。すなわち、個々の根の誤差限界は多項式の係数の絶対値、根自身の絶対値、多項式の微分関数の各根における絶対値によって決定される。

２．２　単根（実根および複素根）に対する反復平行移動計算による誤差改善法

ここで、相異なる実根および複素根を持つn次多項式を考える。n次多項式は一般に（2-1）式で書くことができる。

ここで、（2-12）式を応用し一度計算機を用いて計算された個々の多項式の根（一般的には誤差を含む）を用い、根の平行移動による反復再計算によって誤差を改善する方法を述べる。（2-12）式に対する多項式の誤差限界は、真

の根を z_k（$k=1\sim n$，$i=0\sim n$）とし、その誤差を Δz_k とすると、

$$|\Delta z_k| \leqq 2^{-t} \sum_{k=0}^{n} |a_{i+1}|\,|z_k|^i \Big/ |f'(z_k)| \qquad (2\text{-}13)$$

となる。(2-13) 式を詳しく書き表わすと、任意の z_k に対してその誤差は、

$$|\Delta z_k| \leqq 2^{-t} \{|z_k|^n + |a_n|\,|z_k|^{n-1} + |a_{n-1}|\,|z_k|^{n-2} \cdots + |a_2|\,|z_k| + |a_1|\} \Big/ |f'(z_k)| \qquad (2\text{-}14)$$

となる。(2-14) 式から各根の誤差限界は多項式の根自身の絶対値の大きさ、全根から決定される多項式の係数（$|a_{i+1}|\,(i=0\sim n-1)$）の絶対値、および $|f'(z_k)|\,(k=1\sim n)$ によって決定されることになる。

ここで、一度根計算が行なわれ誤差を含む近似根 $\underline{z_k}$ が計算されたと仮定する。(2-14) 式に対し全根の平行移動 $(-\underline{z_k})$ を考える。ここで $\underline{z_k}$ は任意の一つの根を選択する。平行移動により (2-14) 式は、平行移動後の誤差限界を $|\underline{\Delta z_k}|$ とすると

$$|\underline{\Delta z_k}| \leqq 2^{-t} \{|z_k-\underline{z_k}|^n + |a_n|\,|z_k-\underline{z_k}|^{n-1} + |a_{n-1}|\,|z_k-\underline{z_k}|^{n-2} \cdots + |a_2|\,|z_k-\underline{z_k}| + |a_1|\} \Big/ |f'(z_k)| \qquad (2\text{-}15)$$

となる。ここで、平行移動によって $|f'(z_k)|$ の値は変化しないことに注意する（2．4節で述べる）。$|z_k-\underline{z_k}|=|\varepsilon|$ とおくと

$$|\underline{\Delta z_k}| \leqq 2^{-t} \{|\varepsilon|^n + |a_n|\,|\varepsilon|^{n-1} + |a_{n-1}|\,|\varepsilon|^{n-2} \cdots + |a_2|\,|\varepsilon| + |a_1|\} \Big/ |f'(z_k)| \qquad (2\text{-}16)$$

となる。ε は、真の根と一度計算された根の差であるから非常に小さな値となる。二度目に計算された根に対する誤差限界式の値は一度目に計算された値より真の根に近づくから誤差限界式の値は減少する。誤差限界式の値が減少すれば、計算される根の精度は向上するから精度の向上した根に対する平行移動によって次々にこの操作を繰り返せば根の精度は向上することになる。この反復計算によって $|\varepsilon| \fallingdotseq 0$ となり、(2-16) 式は近似的に

$$|\underline{\Delta z_k}| \leqq 2^{-t} |a_1|^n \Big/ |f'(z_k)| \qquad (2\text{-}17)$$

となる。(2-17) 式の右辺の値は (2-14) 式の右辺と比較すれば明らかに減少するから（特に多項式の係数の絶対値が大きいとき）(2-14) 式および (2-17) 式から

$$|\Delta z_k| \geqq |\underline{\Delta z_k}| \qquad (2\text{-}18)$$

が成立する。したがって、根の反復移動計算により任意の根の精度は改善されることとなる。

2．3　重根を含む場合の反復平行移動計算による誤差改善法

2．2で述べた単根に対する平行移動と重根を含む場合の改善法は本質的にはあまり変わらない。誤差限界式が次の（$\sum_{k=0}^{n}$ 11-2）式によってもとめられているので、

$$|\Delta \xi| \leqq 2^{-t/m} \left\{ |a_{k+1}| \Big/ |\xi|^k / |f^{(m)}(\xi)| \right\}^{1/m} \qquad (2\text{-}11\text{-}2)$$

この式を詳しく書き表わすと

$$|\Delta z_k| \leqq 2^{-t/m} \{ |z_k|^n + |a_n| |z_k|^{n-1} + |a_{n-1}| |z_k|^{n-2} \cdots + |a_2| |z_k| + |a_1| \}^{1/m} / |f^{(m)}(z_k)|^{1/m} \quad (2\text{-}19)$$

となる。ここで単根の時と同様に、一度根計算が行なわれ誤差を含む近似根 $\underline{z_k}$ が計算されたと仮定する。(2-19) 式に対し全根の平行移動 $(-\underline{z_k})$ を考える。ここで $\underline{z_k}$ は任意の重根とする。平行移動により (2-19) 式は、平行移動後の誤差限界を $|\underline{\Delta z_k}|$ とすると

$$|\underline{\Delta z_k}| \leqq 2^{-t/m} \{ |z_k - \underline{z_k}|^n + |a_n| |z_k - \underline{z_k}|^{n-1} + |a_{n-1}| |z_k - \underline{z_k}|^{n-2} \cdots + |a_2| |z_k - \underline{z_k}| + |a_1| \}^{1/m} / |f^{(m)}(z_k)|^{1/m} \quad (2\text{-}20)$$

となる。ここで、平行移動によって $|f^{(m)}(z_k)|^{1/m}$ の値は変化しないことに注意する (2.4節で述べる)。

$|z_k - \underline{z_k}| = \varepsilon$ とおくと

$$|\underline{\Delta z_k}| \leqq 2^{-t/m} \{ |\varepsilon|^n + |a_n| |\varepsilon|^{n-1} + |a_{n-1}| |\varepsilon|^{n-2} \cdots + |a_2| |\varepsilon| + |a_1| \}^{1/m} / |f^{(m)}(z_k)|^{1/m} \quad (2\text{-}21)$$

となる。ε は、真の根と一度計算された根の差であるから非常に小さな値となる。二度目に計算された根に対する誤差限界式の値は一度目に計算された値より真の根に近づくから誤差限界式の値は減少する。誤差限界式の値が減少すれば、計算される根の精度は向上するから精度の向上した根に対する平行移動によって次々にこの操作を繰り返せば根の精度は向上することになる。この反復計算によって $|\varepsilon| \fallingdotseq 0$ となり、(2-21) 式は近似的に

$$|\underline{\Delta z_k}| \leqq 2^{-t/m} |a_1|^{1/m} / |f^{(m)}(z_k)|^{1/m} \quad (2\text{-}22)$$

となる。(2-22) 式の右辺の値は (2-19) 式の右辺と比較すれば明らかに減少するから (特に多項式の係数の絶対値が大きいとき) (2-22) 式および (2-19) 式から

$$|\Delta z_k| \geqq |\underline{\Delta z_k}| \quad (2\text{-}23)$$

の関係が成立し、重根に対しても、誤差限界の値は減少することになる。したがって、この操作を繰り返すことによって重根に対する精度を向上させることができる。

2．4　平行移動による $|f'(z)|$ および $|f^{(m)}(z)|$ について

2．2および2．3で全根の平行移動によって $|f'(z)|$ と $|f^{(m)}(z)|$ の値は変化しないということを述べた。複素数での不変性をここでは述べる。(2-1) 式に対して最初に単根の場合を考える。根異なる複素根を $\alpha_1, \alpha_2, \alpha_3, \cdots, \alpha_n$ とすると、$f(z)$ は複素数の範囲で、

$$f(z) = (z - \alpha_1)(z - \alpha_2) \cdots (z - \alpha_n) \quad (2\text{-}24)$$

と因数分解することができる (文献 (5))。(2-24) 式に対し z に関する微分を行なうと、

$$f'(z) = (z - \alpha_2) \cdots (z - \alpha_n) + (z - \alpha_1) \{ (z - \alpha_2) \cdots (z - \alpha_n) \}' \quad (2\text{-}25)$$

となる。したがって $z = \alpha_1$ に関する微分値は

$$f'(\alpha_1) = (\alpha_1 - \alpha_2)(\alpha_1 - \alpha_3) \cdots (\alpha_1 - \alpha_n) \quad (2\text{-}26)$$

となり、任意の α_k (k= 1 , n) に対し

$$f'(\alpha_k)=(\alpha_k-\alpha_1)(\alpha_k-\alpha_2)(\alpha_k-\alpha_3)\cdots(\alpha_k-\alpha_{k-1})(\alpha_k-\alpha_{k+1})\cdots(\alpha_k-\alpha_n) \quad (2\text{-}27)$$

が成立する。従って、(2-27) 式に対しすべての根の z_p の平行移動を行なうと (2-27) 式は

$$f'(\alpha_k)=(\alpha_k+z_p-\alpha_1-z_p)(\alpha_k+z_p-\alpha_2-z_p)(\alpha_k+z_p-\alpha_3-z_p)\cdots(\alpha_k+z_p-\alpha_{k-1}-z_p)(\alpha_k+z_p-\alpha_{k+1}-z_p)\cdots(\alpha_k+z_p-\alpha_n-z_p) \quad (2\text{-}28)$$

となるから、平行移動しても α_k に対する微分値の絶対値は変化しない。次にm重根の場合を考える。

$$f(z)=(z-\alpha_1)^m \quad (2\text{-}29)$$

とし、微分を行なうと

$$f'(z)=m(z-\alpha_1)^{m-1} \quad (2\text{-}30)$$

$$f^{(2)}(z)=m(m-1)(z-\alpha_1)^{m-2} \quad (2\text{-}31)$$

$$f^{(3)}(z)=m(m-1)(m-2)(z-\alpha_1)^{m-3} \quad (2\text{-}32)$$

となるので、$|f'(\alpha_1)|=0$、$|f^{(2)}(\alpha_1)|=0$、$|f^{(m-1)}(\alpha_1)|=0$の関係が成立し、

$$f^{(m)}(\alpha_1)=m(m-1)(m-2)(z-\alpha_1)^{m-m}=m(m-1)(m-2) \quad (2\text{-}33)$$

となるから $|f^{(m)}(z)|\neq 0$ の値は平行移動により変化しない。最後に、重根、単根両方を含む場合を考える。

ここで、

$$f(z)=(z-\alpha_1)^m g(z) \quad (2\text{-}34)$$

とおく。(2-34) 式の $(z-\alpha_1)^m$ はm乗根を表わし $g(z)$ は $n-m$ 次単根を持つ多項式を表わすものとする。(2-34) 式を z に関して微分すると、

$$f'(z)=m(z-\alpha_1)^{m-1}g(z)+(z-\alpha_1)^m g'(z) \quad (2\text{-}35)$$

となるから、さらに微分操作を続けると

$$f^{(2)}(z)=m(m-1)(z-\alpha_1)^{m-2}g(z)+m(z-\alpha_1)^{m-1}g'(z)+m(z-\alpha_1)^{m-1}g'(z)+(z-\alpha_1)^m g^{(2)}(z) \quad (2\text{-}36)$$

$$f^{(3)}(z)=m(m-1)(m-2)(z-\alpha_1)^{m-3}g(z)+m(m-1)(z-\alpha_1)^{m-2}g'(z)+m(m-1)(z-\alpha_1)^{m-2}g'(z)+m(z-\alpha_1)^{m-1}g^{(2)}(z)+m(m-1)(z-\alpha_1)^{m-2}g'(z)+m(z-\alpha_1)^{m-1}g^{(2)}(z)+m(z-\alpha_1)^{m-1}g^{(2)}(z)+(z-\alpha_1)^m g^{(3)}(z) \quad (2\text{-}37)$$

となる。ここで (2-37) 式に対して

$$z=\alpha_1 \quad (2\text{-}38)$$

の関係を代入し、一般的な関係を求めると、

$$f^{(h)}(\alpha_1)=m(m-1)(m-2)\cdots(m-(h-1))(z-\alpha_1)^{m-h}$$

$$g(\alpha_1) \qquad (2\text{-}39)$$

となり、m＝hの関係（重根に対しその微分係数がはじめて0でなくなる条件）を代入すると

$$f^{(h)}(\alpha_1) = m(m-1)(m-2)\cdots(m-(h-1))\ g(\alpha_1) \qquad (2\text{-}40)$$

となり（2-40）式の$g(\alpha_1)$が平行移動によって変化しない。これは単根をz_1、z_2、…、z_{n-m}とすると、$g(\alpha_1)$は

$$g(\alpha_1) = (\alpha_1 - z_1)(\alpha_1 - z_2)(\alpha_1 - z_3)\cdots(\alpha_1 - z_{n-m}) \qquad (2\text{-}41)$$

となりすべての根の複素平行移動量を$+z_p$とすると、

$$g(\alpha_1) = (\alpha_1 + z_p - z_1 - z_p)(\alpha_1 + z_p - z_2 - z_p)(\alpha_1 + z_p - z_3 - z_p)\cdots(\alpha_1 + z_p - z_{n-m} - z_p) \qquad (2\text{-}42)$$

となる。従って（2-40）式および（2-42）式から$|f^{(h)}(\alpha_1)|$の値はすべての根の平行移動によって変化しない。

3　例題

本例題に対するシミュレーションは、Macintosh MATLABを使用して行なった。

例題1

文献［2］における例題を少しかえ、複素根はないが重根（19）が存在する場合を取り上げる（18は含まない）。ここでは、多項式

$$f(x) = (x-20)(x-19)(x-19)(x-17)(x-16)\cdots(x-3)(x-2)(x-1) = 0$$

の個々の根の誤差を少なくするために2回の反復平行移動計算を行なって単根はもちろん重根に対しても精度の改善ができることを示す。対角行列Aから多項式P（プログラムではPは多項式の係数を持つ横ベクトル）をつくりその多項式からroots(P)命令を使用し平行移動を行なわないで根を求めた結果がx0であり、これを図1に示す。次に、根x0を基とし平行移動を2回行なうプログラムと1回の平行移動によって根 2.000010533542072 e ＋ 01（正確には20であるが誤差が生じている。プログラムではHEIRYOとしてエリアをとっている）の値が 2.000000000000000 e ＋ 01 となり、誤差が減少している様子を図2に示す。他の根の精度は悪化しているものも存在する。さらに、1回の平行移動で求められた値を平行移動量として（HEIRYO2）2回目の再計算を行なった場合の多項式の係数を図3に示し、その結果を図4に示す。次に同様な方法によって、重根（19）を計算した結果を図5および図6に示す。図5、図6をみると一度計算された値が複素根になっているためか再計算を行なった結果がすべて複素根になっているが一度計算された値、1回平行移動を行なって再計算を行なった結果と2回目の平行移動計算を行なった結果を比較すると実部はもちろん虚部も値が小さくなり誤差改善が行なわれていることがわかる。

例題2

例題1と全く同様な方法で10次元の全てが複素根であり、重根を含む場合の2回反復平行移動計算を行なった例を示す。

$$f(Z) = (1 + 20\,i - Z)(1 + 20\,i - Z)(3 + 18\,i - Z)(4 + 17\,i - Z)$$

$$(5+16i-Z)(6+15i-Z)$$
$$(7+14i-Z)(8+13i-Z)$$
$$(9+12i-Z)(10+11i-Z)$$
$$=0$$

の複素根をもつ多項式を考える。最初に計算された結果を図7に示す。図8には1回目の平行移動を行なった場合の重根に対する結果および2回目の平行移動を行なった場合の結果を示す。2回目の平行移動で重根1＋20iが正確に計算されていることがわかる。単根10＋11iに対する結果を図9に示す。この場合には1回目の平行移動で精度が非常によく2回目の平行移動を必要としないと考えられる。ここで他の根の誤差は少し変化している。

4．おわりに

多項式の根を計算機を用いて計算する場合、多項式の係数のわずかな変動で根の誤差が増加してしまう場合がある。本稿では、複素根および重根が多項式の根となる場合に対する誤差改善反復法を提案し、その有用性を具体的にシミュレーションを行ない示した。本方法の使用によって個々の根の精度を向上させることが可能となる。時間はかかっても精度が欲しい場合には本反復改善法は有効になると考えられる。

5．謝　辞

本研究にあたり、非常に細かな部分に注意をしていただきました東京工業大学美多勉教授に感謝いたします。

6．参考文献

(1) Wilkinson J. H:The Algebraic Eigenvalue Problem、OXFORD、p89－p90、1988年

(2) 牧之内三郎：数値解析、オーム社、p77－83、p150－152、1975年

(3) 町田東一　他：マトリクスの固有値と対角化、東海大学出版会、p36－39、1990年

(4) 有本　卓：数値解析(1)、コロナ社、p49－51、1981年

(5) 小林善一：複素数、共立出版、p90－91、1966年

(6) 田口　功：行列の一次摂動に関する固有値の誤差に対する一考察、千葉敬愛短期大学紀要、第16号、1994年

```
A =

  Columns 1 through 12

   20    0    0    0    0    0    0    0    0    0    0    0
    0   19    0    0    0    0    0    0    0    0    0    0
    0    0   19    0    0    0    0    0    0    0    0    0
    0    0    0   17    0    0    0    0    0    0    0    0
    0    0    0    0   16    0    0    0    0    0    0    0
    0    0    0    0    0   15    0    0    0    0    0    0
    0    0    0    0    0    0   14    0    0    0    0    0
    0    0    0    0    0    0    0   13    0    0    0    0
    0    0    0    0    0    0    0    0   12    0    0    0
    0    0    0    0    0    0    0    0    0   11    0    0
    0    0    0    0    0    0    0    0    0    0   10    0
    0    0    0    0    0    0    0    0    0    0    0    9
    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0    0    0    0    0

  Columns 13 through 20

    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    0    0    0    0    0    0    0    0
    8    0    0    0    0    0    0    0
    0    7    0    0    0    0    0    0
    0    0    6    0    0    0    0    0
    0    0    0    5    0    0    0    0
    0    0    0    0    4    0    0    0
    0    0    0    0    0    3    0    0
    0    0    0    0    0    0    2    0
    0    0    0    0    0    0    0    1
```

図1－1 対角行列A
Fig.1−1 A diagonal matrix A.

```
»A(3,3)=19;
»P=poly(A);
»X0=roots(P)

X0 =

      2.000010533542072e+01
      1.900020710635934e+01 + 2.254364619855954e-02i
      1.900020710635934e+01 - 2.254364619855954e-02i
      1.699666111128294e+01
      1.600892782173941e+01
      1.498458467586702e+01
      1.401903294759144e+01
      1.298207451117433e+01
      1.201311601995228e+01
      1.099277771079844e+01
      1.000309251674159e+01
      8.999015130003038e+00
      8.000233560535481e+00
      6.999959988220620e+00
      6.000004841132842e+00
      4.999999593788566e+00
      4.000000024046072e+00
      2.999999998959655e+00
      2.000000000027119e+00
      9.999999999998042e-01
```

図１－２　多項式ｆ（ｘ）の根ｘ0

Fig.1−2　Computing roots of a polynomial f(x)

```
»for i=1:5
 HEIRYO=X0(i,1)
 PHi=poly(A-eye(20)*HEIRYO);
 Xi=roots(PHi)+HEIRYO
  HEIRYO2=Xi(20,1)
PHi2=poly(A-eye(20)*HEIRYO2)
Xi2=roots(PHi2)+HEIRYO2
  end

HEIRYO =

      2.000010533542072e+01


Xi =

      9.999013776966486e-01
      2.000854898592515e+00
      2.996651216971028e+00
      4.008021886553404e+00
      4.987273913033711e+00
      6.014611754047586e+00
      6.987569998649432e+00
      8.007916011270856e+00
      8.996053035017093e+00
      1.000150991102519e+01
      1.099954903938499e+01
      1.200010195272044e+01
      1.299998329969355e+01
      1.400000181346155e+01
      1.499999988950425e+01
      1.600000000234255e+01
      1.700000000003550e+01
      1.899999986818776e+01
      1.900000013181216e+01
      2.000000000000000e+01
```

図２　一度計算された根ｘ0を用い平行移動を行なうプログラムと結果

Fig.2　A recomputation program using x0 by parallel translation and results.

```
HEIRYO2 =

    2C


PHi2 =

  Columns 1 through 3

    1.000000000000000e+00    1.890000000000000e+02    1.662700000000000e+04

  Columns 4 through 6

    9.041110000000000e+05    3.402927400000000e+07    9.408002980000000e+0

  Columns 7 through 9

    1.978527493400000e+10    3.233748956620000e+11    4.160729477957000e+12

  Columns 10 through 12

    4.242441555337700e+13    3.433538721473510e+14    2.199073066675043e+15

  Columns 13 through 15

    1.105934961206136e+16    4.309584112833390e+16    1.274989576359639e+17

  Columns 16 through 18

    2.778166086034192e+17    4.258676458788814e+17    4.269905143238822e+17

  Columns 19 through 21

    2.461938485110272e+17    6.082255020441600e+16                    C


Xi2 =

    2.000000000000000e+01
    9.999742037662784e-01
    2.000231213095038e+0C
    2.999031823364181e+0C
    4.002518723460726e+0C
    4.995527937226120e+0C
    6.005766707976703e+0C
    6.994516186094025e+0C
    8.003873207349384e+0C
    8.997948747819501e+0C
    1.000079891961485e+01
    1.099977132788789e+01
    1.200004740109502e+01
    1.299999288143188e+01
    1.400000077613111e+01
    1.499999994128488e+01
    1.600000000241327e+01
    1.699999999998925e+01
    1.899999977560154e+01
    1.900000022439822e+01
```

図3　2回の平行移動による多項式の係数

Fig.3　Coefficients of a polynomial f(x) by two times parallel translations.

図4　2回の平行移動による根20の計算結果

Fig.4　A result by two times parallel translation for a root 20.

```
HEIRYO =

      1.900020710635934e+01 + 2.254364619855954e-02i


Xi =

      1.000007597957801e+00 + 4.107262019589775e-07i
      1.999930069040161e+00 - 3.828560327066854e-06i
      3.000295039682070e+00 + 1.643863612854971e-05i
      3.999247138548172e+00 - 4.284914565758427e-05i
      5.001298533976083e+00 + 7.609757336289255e-05i
      5.998409273345263e+00 - 9.645600613732500e-05i
      7.001424659689519e+00 + 9.007812639789600e-05i
      7.999055326720056e+00 - 6.283432244060319e-05i
      9.000462491263438e+00 + 3.258387922582331e-05i
      9.999833933469185e+00 - 1.252621429307862e-05i
      1.100004271923355e+01 + 3.483112064897598e-06i
      2.000000000000000e+01 + 3.122502256758253e-17i
      1.199999238684385e+01 - 6.792187363484359e-07i
      1.300000089138082e+01 + 8.816537728703611e-08i
      1.399999993638276e+01 - 7.055944630657285e-09i
      1.500000000251663e+01 + 3.110446136767031e-10i
      1.599999999995017e+01 - 6.286103582109348e-12i
      1.700000000000037e+01 + 1.874889132835733e-14i
      1.899999999999766e+01 - 2.732959067386354e-10i
      1.900000000000234e+01 + 2.732958859219536e-10i
```

図5　重根（19）に対する1回目の平行移動による結果

Fig.5　A result for multiple roots x＝19 by parallel translation.

```
Xi2 =

      1.000000370661514e+00 + 3.501665347528755e-09i
      2.000008378102628e+00 - 3.335779659336736e-08i
      2.999923194726478e+00 + 1.436858194348041e-07i
      4.000279962082855e+00 - 3.723428007389080e-07i
      4.999412980189598e+00 + 6.483992995743192e-07i
      6.000800229675971e+00 - 8.075338368395409e-07i
      6.999253569120278e+00 + 7.423090316550840e-07i
      8.000486117193374e+00 - 5.128113692940637e-07i
      8.999778923786682e+00 + 2.690788975655656e-07i
      1.000006845292573e+01 - 1.065879991812177e-07i
      2.000000000000000e+01 + 2.261069754570508e-17i
      1.099998635953692e+01 + 3.147787515288426e-08i
      1.200000151723843e+01 - 6.712364003650370e-09i
      1.299999995013791e+01 + 9.772161100103223e-10i
      1.399999999427483e+01 - 8.733498118061525e-11i
      1.500000000033086e+01 + 3.693818062936589e-12i
      1.600000000001670e+01 + 6.414064963191285e-15i
      1.699999999999925e+01 - 3.695169531461832e-15i
      1.900000000000000e+01 - 6.540586896820974e-18i
      1.900000000000000e+01 + 6.540586741724609e-18i
```

図6－1　重根に対する2回目の平行移動による結果

Fig.6－1　A result for multiple roots x＝19 by two times parallel translations.

```
HEIRYO =

      1.900020710635934e+01 - 2.254364619855954e-02i


Xi =

      1.000007597957801e+00 - 4.107262019589775e-07i
      1.999930069040161e+00 + 3.828560327066854e-06i
      3.000295039682070e+00 - 1.643863612854971e-05i
      3.999247138548172e+00 + 4.284914565758427e-05i
      5.001298533976083e+00 - 7.609757336289255e-05i
      5.998409273345263e+00 + 9.645600613732500e-05i
      7.001424659689519e+00 - 9.007812639789600e-05i
      7.999055326720056e+00 + 6.283432244060319e-05i
      9.000462491263438e+00 - 3.258387922582331e-05i
      9.999833933469185e+00 + 1.252621429307862e-05i
      1.100004271923355e+01 - 3.483112064897598e-06i
      2.000000000000000e+01 - 3.122502256758253e-17i
      1.199999238684385e+01 + 6.792187363484359e-07i
      1.300000089138082e+01 - 8.816537728703611e-08i
      1.399999993638276e+01 + 7.055944630657285e-09i
      1.500000000251663e+01 - 3.110446136767031e-10i
      1.599999999995017e+01 + 6.286103582109348e-12i
      1.700000000000037e+01 - 1.874889132835733e-14i
      1.899999999999766e+01 + 2.732959067386354e-10i
      1.900000000000234e+01 - 2.732958859219536e-10i


HEIRYO2 =

      1.900000000000234e+01 - 2.732958859219536e-10i


Xi2 =

      1.000000370661514e+00 - 3.501665347528755e-09i
      2.000008378102628e+00 + 3.335779659336736e-08i
      2.999923194726478e+00 - 1.436858194348041e-07i
      4.000279962082855e+00 + 3.723428007389080e-07i
      4.999412980189598e+00 - 6.483992995743192e-07i
      6.000800229675971e+00 + 8.075338368395409e-07i
      6.999253569120278e+00 - 7.423090316550840e-07i
      8.000486117193374e+00 + 5.128113692940637e-07i
      8.999778923786682e+00 - 2.690788975655656e-07i
      1.000006845292573e+01 + 1.065879991812177e-07i
      2.000000000000000e+01 - 2.261069754570508e-17i
      1.099998635953692e+01 - 3.147787515288426e-08i
      1.200000151723843e+01 + 6.712364003650370e-09i
      1.299999995013791e+01 - 9.772161100103223e-10i
      1.399999999427483e+01 + 8.733498118061525e-11i
      1.500000000033086e+01 - 3.693818062936589e-12i
      1.600000000001670e+01 - 6.414064963191285e-15i
      1.699999999999925e+01 + 3.695169531461832e-15i
      1.900000000000000e+01 + 6.540586896820974e-18i
      1.900000000000000e+01 - 6.540586741724609e-18i
```

図６－２　重根に対する１回目、２回目の平行移動による結果

Fig. 2－2　A result for another multiple roots x＝19 by parallel translation.

```
»A(2,2)=A(1,1);
»X0=roots(poly(A))

X0 =

      9.999498212826085e-01 + 2.000022746585358e+01i
      1.000050150050617e+00 + 1.999977248043963e+01i
      9.999999997748182e+00 + 1.100000000476692e+01i
      9.000000033018308e+00 + 1.199999996157251e+01i
      7.999999810314311e+00 + 1.300000012569354e+01i
      3.000000299237538e+00 + 1.800000028211130e+01i
      7.000000589208924e+00 + 1.399999981388673e+01i
      3.999999133820269e+00 + 1.699999952051131e+01i
      5.999998900657267e+00 + 1.500000003095681e+01i
      5.000001264661974e+00 + 1.600000031420765e+01i

»for k=1:10
  HEIRYO=X0(k,1);
  PHi=poly(A-eye(10)*HEIRYO);
  Xi=roots(PHi)+HEIRYO
 HEIRYO2=Xi(10,1)
 PHi2=poly(A-eye(10)*HEIRYO2);
 Xi2=roots(PHi2)+HEIRYO2
end

HEIRYO =

      9.999498212826085e-01 + 2.000022746585358e+01i

Xi =

      9.999999999980140e+00 + 1.100000000004683e+01i
      9.000000000099027e+00 + 1.199999999979332e+01i
      7.999999999804031e+00 + 1.300000000037079e+01i
      7.000000000199482e+00 + 1.399999999965367e+01i
      5.999999999887855e+00 + 1.500000000017887e+01i
      5.000000000034235e+00 + 1.599999999995036e+01i
      3.999999999994968e+00 + 1.700000000000646e+01i
      3.000000000000244e+00 + 1.799999999999974e+01i
      1.000000000003888e+00 + 1.999999999999991e+01i
      9.999999999961121e-01 + 2.000000000000009e+01i


HEIRYO2 =

      9.999999999961121e-01 + 2.000000000000009e+01i


Xi2 =

      1.000000000002592e+01 + 1.100000000000595e+01i
      8.999999999888429e+00 + 1.199999999997820e+01i
      8.000000000187141e+00 + 1.300000000003855e+01i
      6.999999999845215e+00 + 1.399999999995749e+01i
      6.000000000065643e+00 + 1.500000000002895e+01i
      4.999999999986478e+00 + 1.599999999998907e+01i
      4.000000000001225e+00 + 1.700000000000189e+01i
      2.999999999999948e+00 + 1.799999999999990e+01i
      1.000000000000000e+00 + 2.000000000000000e+01i
      1.000000000000000e+00 + 2.000000000000000e+01i
```

図7　平行移動のないときの計算結果
Fig.7　A computed roots without parallel translation.

図8．1　重根（1+20i）に対する改善結果
Fig.8−1　Results Xi and Xi2, for multiple roots x=1+20i by parallel translation.

```
HEIRYO =

      1.000050150050617e+00 + 1.999977248043963e+01i


Xi =

      1.000000000001081e+01 + 1.100000000000752e+01i
      8.999999999955261e+00 + 1.199999999996620e+01i
      8.000000000071640e+00 + 1.300000000006215e+01i
      6.999999999944241e+00 + 1.399999999993898e+01i
      6.000000000021296e+00 + 1.500000000003484e+01i
      4.999999999996727e+00 + 1.599999999998841e+01i
      3.999999999999981e+00 + 1.700000000000205e+01i
      3.000000000000027e+00 + 1.799999999999986e+01i
      1.000000000000645e+00 + 1.999999999999383e+01i
      9.999999999993551e-01 + 2.000000000000617e+01i


HEIRYO2 =

      9.999999999993551e-01 + 2.000000000000617e+01i


Xi2 =

      9.999999999961295e+00 + 1.100000000001301e+01i
      9.000000000164176e+00 + 1.199999999994434e+01i
      7.999999999720657e+00 + 1.300000000009840e+01i
      7.000000000243383e+00 + 1.399999999990799e+01i
      5.999999999885414e+00 + 1.500000000004855e+01i
      5.000000000028099e+00 + 1.599999999998568e+01i
      3.999999999996886e+00 + 1.700000000000216e+01i
      3.000000000000101e+00 + 1.799999999999988e+01i
      1.000000000000000e+00 + 2.000000000000000e+01i
      1.000000000000000e+00 + 2.000000000000000e+01i
```

図８．２　重根（1+20i）に対する改善結果
Fig.8−2　A result for another multiple root x=1+20i by parallel translation

```
HEIRYO =

      9.999999997748182e+00 + 1.100000000476692e+01i


Xi =

      9.999942617518052e-01 + 2.000000695670385e+01i
      1.000005738279759e+00 + 1.999999304324848e+01i
      2.999999999916064e+00 + 1.800000000013110e+01i
      4.000000000076984e+00 + 1.699999999987523e+01i
      4.999999999972487e+00 + 1.600000000004591e+01i
      6.000000000002473e+00 + 1.499999999999710e+01i
      7.000000000000537e+00 + 1.399999999999792e+01i
      7.999999999999934e+00 + 1.300000000000039e+01i
      9.000000000000000e+00 + 1.199999999999999e+01i
      1.000000000000000e+01 + 1.100000000000000e+01i


HEIRYO2 =

      1.000000000000000e+01 + 1.100000000000000e+01i


Xi2 =

      1.000000000000000e+01 + 1.100000000000000e+01i
      9.999945602126115e-01 + 2.000000530872603e+01i
      1.000005439821475e+00 + 1.999999469124226e+01i
      2.999999999883081e+00 + 1.800000000010374e+01i
      4.000000000146187e+00 + 1.699999999987728e+01i
      4.999999999913916e+00 + 1.600000000006686e+01i
      6.000000000026491e+00 + 1.499999999998157e+01i
      6.999999999996021e+00 + 1.400000000000236e+01i
      8.000000000000195e+00 + 1.299999999999992e+01i
      9.000000000000005e+00 + 1.199999999999999e+01i
```

図９　10+11iに対する改善結果
Fig.9　A result for x=10+11i by parallel translation.

ABSTRACT

A Method of Improvement of Errors in the Numerical Computation of Polynomials with Real and Complex Roots Using the Limit of Errors.

Isao Taguchi

Various errors occur in the numerical computation of roots of an integral equation $f(z)=0$ by computer-system, and roots change frequently from real to complex. In this paper, the author discusses the improvement of individual roots and the case of real and complex roots, and their multiple roots. Here, I use the notion of the limit of error and the iteration of parallel translation. Then the computation of individual roots is very improved. The results are verified by simulation.