

# Durand-Kerner 法と Aberth 法を用いた超高次方程式 の数値計算†

小野 令美††

高次方程式の数値解法で最近注目されてきたものに全根同時反復型解法 Durand-Kerner 法と Aberth 法がある (DKA 法と略す)。例として Chebyshev の数値積分公式の分点を与える高次方程式をとりあげ、1000 次以上におよぶ超高次方程式をこの解法で解いた。さらに低次のものについては system に備えられている subroutine library と比較してみた。

その結果次数が高くなるにつれ従来の解法では得られた解の精度が正しく評価できなかったが、DKA 法では Gerschgorin circle 半径の範囲内で正しく求められた。超高次方程式については DKA 法でも桁落ちのため必要な桁数の解が得られなくなるので、一部多倍長演算が必要になる。このようにして最高 1024 次のもので Gerschgorin circle 半径が完全に分離した解を得た。その結果解は  $n \rightarrow \infty$  の極限で根が並ぶと予想されている曲線に近づくことが数値的に確かめられた。これらの結果を述べる。

さらにこの数値実験を通して得られたこの種の大規模計算を行う際に注意すべきことがらを述べる。このうち

- 3.1 問題の特殊性の利用
  - 3.2 多倍長演算の効果的使用 (計算の各段階での必要な計算桁数の解析)
  - 3.3 計算法の手間の検討
- などが特に重要な知見である。

## 1. ま え が き

複素係数  $n$  次代数方程式

$$P(z) = z^n + a_1 z^{n-1} + \dots + a_{n-1} z + a_n = 0 \quad (1)$$

の数値解法は、昔からいろいろと研究されている。そのおもなものには Newton-Raphson 法, Bairstow 法, Jarratt 法<sup>6)</sup>, Lehmer 法<sup>9)</sup>, 商差法<sup>7)</sup>, 大中の方法<sup>8)</sup>などがあるが、一根  $\alpha$  を求め  $P(z)$  を  $z - \alpha$  で割って次数を下げていく方法では、次第に丸め誤差が累積して、最後に得られた解はおよそ真の解とは離れてしまうといったことも起り得る。

全根同時反復型解法で最近注目されてきた Durand-Kerner 法<sup>1), 3), 4)</sup> と Aberth 法<sup>3), 4)</sup> は (以下 DKA 法と略す)、次数を下げるという操作がないから、係数は入力した精度が保たれ前述のような心配がない。また得られた解の精度が正しく評価できるから、計算時間が少し多いことを除けば、最も優れた解法と思われる。

この解法を用いて、一部多倍長計算により解くことができた 1000 次以上におよぶ超高次方程式 (Chebyshev の数値積分公式の分点を与えるための方

式) の数値計算例と、この種の計算を行う際に留意すべきことがらについて述べる。

## 2. ある超高次方程式の数値計算

### 2.1 問題提起

Chebyshev の数値積分公式

$$\int_{-1}^1 f(x) dx \approx \frac{2}{n} \sum_{i=1}^n f(x_i) \quad (2)$$

の分点  $x_i$  を零点にもつ多項式を  $F_n(z)$  とおくと

$$F_n(z) = a_0 z^n + a_2 z^{n-2} + a_4 z^{n-4} + \dots + \begin{cases} a_{n-1} z & (n: \text{奇数}) \\ a_n & (n: \text{偶数}) \end{cases} \quad (3)$$

$$a_0 = 1, a_{2k} = -\frac{n}{2k} \sum_{j=1}^k \frac{1}{2j+1} a_{2(k-j)} \quad (k=1, 2, 3, \dots) \quad (4)$$

である。よく知られているように  $F_n(z) = 0$  の根は、 $n=1, 2, 3, 4, 5, 6, 7, 9$  では実根だけであるが、このほかの  $n$  では虚根も含む。  $n \rightarrow \infty$  のとき根は曲線

$$|(z+1)^{(z+1)/2} (z-1)^{-(z-1)/2}| = 2 \quad (5)$$

の上に密に並ぶという予想が森口・伊理によってたてられている<sup>5)</sup>。そこで、DKA 法の適用例としてこの方程式をとりあげ、同時にこの予想を数値実験的に確かめる。虚根を含む最低次 ( $n=8$ ) の数値解と曲線 (5) を図 1 に示す。

### 2.2 DKA 反復法

複素係数の方程式 (1) の根  $\alpha_k (k=1, 2, \dots, n)$  を求

† On Numerical Computation of a High Degree Polynomial Equation by the Methods of Durand-Kerner and Aberth by HARUMI ONO (Tokyo Metropolitan Nogeji Agricultural Upper Secondary School).

†† 東京都立農芸高等学校

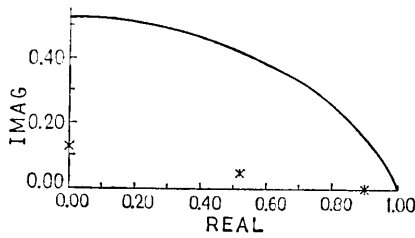


図1  $F_n(z)=0$  の数値解と曲線 (5) の第1象限の部分  
Fig. 1 Numerical solutions of  $F_n(z)=0$  and the curve (5) in the first quadrant.

めるのに、初期値  $z_k^{(0)} (k=1, 2, \dots, n)$  から出発して、第  $\nu+1$  回目の反復による近似値を次式で求める。

$$z_k^{(\nu+1)} = z_k^{(\nu)} + \delta_k^{(\nu)} \quad (6)$$

ここで

$$\delta_k^{(\nu)} = -P(z_k^{(\nu)}) / \prod_{\substack{j=1 \\ j \neq k}}^n (z_k^{(\nu)} - z_j^{(\nu)}) \quad (2 \text{ 次法 Jacobi 型}) \quad (7)$$

または

$$\delta_k^{(\nu)} = -\frac{P(z_k^{(\nu)})}{P'(z_k^{(\nu)})} / \left( 1 - \frac{P(z_k^{(\nu)})}{P'(z_k^{(\nu)})} \sum_{\substack{j=1 \\ j \neq k}}^n \frac{1}{z_k^{(\nu)} - z_j^{(\nu)}} \right) \quad (3 \text{ 次法 Jacobi 型}) \quad (8)$$

近似値  $z_k^{(\nu)}$  の精度は Smith の定理<sup>2)</sup>により

$$|z_k^{(\nu)} - \alpha_k| \leq n |\delta_k^{(\nu)}| \equiv \gamma_k^{(\nu)} \quad (\text{Gerschgorin circle 半径}) \quad (9)$$

である。初期値は

$$P(z) = (z + a_1/n)^n + c_2(z + a_1/n)^{n-2} + \dots + c_n$$

に対し

$$S(w) = w^n - \{|c_2|w^{n-2} + \dots + |c_n|\} = 0$$

とおくと  $S(w)=0$  は唯一つの正根をもつので、 $S(r_0) \geq 0$  となる任意の正数  $r_0^*$  を用いて

$$z_k^{(0)} = -\frac{a_1}{n} + r_0 \exp \left[ \left( \frac{2(k-1)\pi}{n} + \frac{\pi}{2n} \right) i \right], \quad (k=1, 2, \dots, n)$$

とする (Aberth の初期値)。

この解法は解の精度の限界が方程式の性質と計算桁数だけで定まり、大域的収束性をもつ<sup>5)</sup>。

### 2.3 低次の $F_n(z)$ の零点の計算 (他の解法との比較)

2次法 Jacobi 型の DKA 法と system の library (FACOM 230-75) の CNWTND (Newton 法) および JARATD (Jarratt modify 法) とを用いて  $F_n(z)$

表 1 結果の比較

Table 1 Comparison of results.

methods	8	40	60
DKA	(0.1 sec)	6.9 sec	23.6 sec
CNWTND	(0.1 sec)	/	/
JARATD	(0.1 sec)	0.9 sec	2.7 sec

○: good, /: overflow, ?: inaccuracy,  $\Delta$ :  $\max_k \gamma_k = .125_{10} - 4$

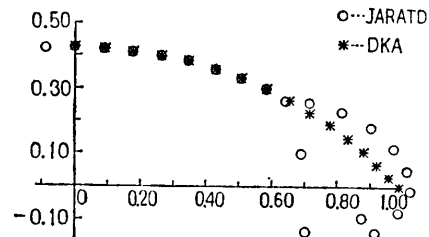


図2  $F_n(z)=0$  の JARATD と DKA 法による数値解 ( $\epsilon=10^{-8}$ )

Fig. 2 Numerical solutions of  $F_n(z)=0$  ( $\epsilon=10^{-8}$ ).

$=0$  ( $n=8, 40, 60$ ) を要求精度  $\epsilon=10^{-8}$  で解いた結果の比較を表1に示す。計算はすべて倍精度である。JARATD の  $n=60$  は要求精度をみたす解が求められたときの指標が示されているのに答がおかしいものである (→図2)。これは収束判定を  $|F(z_i)| < |a_n| \epsilon$  (この例では  $|a_{60}| \approx 0.14_{10} - 10$ ) とし、求められた一次因数で割り減次を行う (全根同時反復型ではない) ので誤差が累積するためと考えられる。したがって必要な精度が  $10^{-8}$  でも、最初の方で得られる解は非常に精度よく求めなければならないと同時に、 $|F(z_i)|$  が十分小さくても  $|z_i - \alpha_i|$  は小さいとはいえない例を示している。この例では  $\epsilon=10^{-15}$  にすれば overflow の error は出るが有効桁数6桁の解が得られた。これらとその他の解法の比較については山本ら<sup>4)</sup>に詳しく述べられており、 $F_n(z)=0$  の例でも同じ傾向を示したといえる。

### 2.4 $F_n(z)=0, n=2^m (m=5(1)10)$ の計算結果

得られた数値解 (Gerschgorin circle は完全に分離) の並ぶ様子と曲線 (5) を図3に、実軸上および虚軸上の根の様子を図4に、計算桁数や計算時間 (FACOM 230-75) などを表2に示す。紙面の都合で係数、数値解などの資料は他の印刷物にする予定である。

\*  $r_0$  は例えば<sup>5)</sup>  $r_0 = \max_m \sqrt[n-1]{|c_m|} \quad (m=2, 3, \dots, n)$ .

表 2  
Table 2

$n$	256	512	1024
digits of multiple-precision arithmetic for computing the roots of $F_n(z)=0$ (63 kw)	64	112	216
digits of multiple-precision arithmetic for computing the coefficients of $F_n(z)$ (100 kw)	120	224	432
$a_{2k}$ for $\max_k  a_{2k} $	$a_{54} = -.437_{10}14$	$a_{110} = -.146_{10}29$	$a_{220} = -.228_{10}58$
$a_{2k}$ for $\min_k  a_{2k} $	$a_{198} = -.587_{10}-38$	$a_{312} = -.551_{10}-72$	$a_{1024} = .139_{10}-140$
C. P. U. time for computing the coefficients of $F_n(z)$	17.3 sec	129 sec	1702.1 sec
C. P. U. time for one iteration	27.5 sec	153.3 sec	993.4 sec

( ) : memory requirements

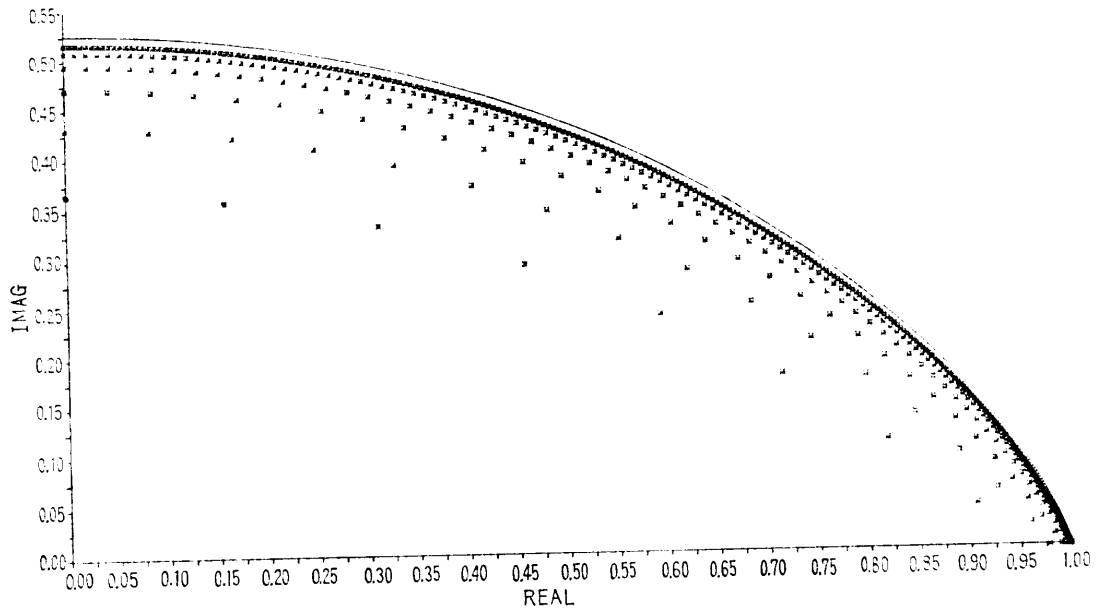


図 3  $F_n(z)=0$ ,  $n=2^m(m=5(1)10)$  の数値解と曲線(5)の第1象限の部分

Fig. 3 Numerical solutions  $F_n(z)=0$  for  $n=2^m(m=5(1)10)$  and the curve (5) in the first quadrant.

### 3. 数値実験で得られた知見およびこの種の計算を行う際の留意点

方程式の次数  $n$  と解の精度の限界との関係は、係数や根の大きさのばらつき、根の重複度などいろいろな要素が関連するので、一般的に  $n$  を用いた関係式で表わすことはできない。

ここでとりあげた  $F_n(z)$  については実験的に次のような関係があることがわかった。  $\max r_k (k=1, 2, \dots, n)$  をもつ  $z_k$  の有効桁数を解の精度の限界の尺度とすると、安全側で見積って (計算桁数)  $-n/5$  桁となると、したがって2進 62 bit で有効桁数3桁とすると  $n=80$  ぐらい、2進 131 bit でも  $n=180$  が限度で、

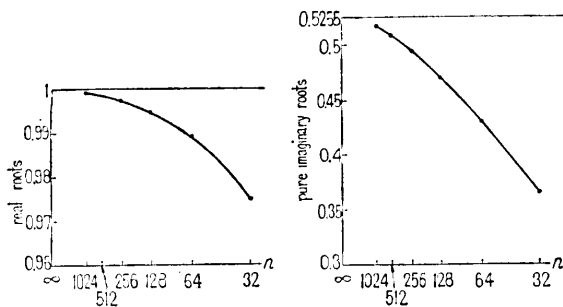


図 4  $F_n(z)=0$  の実軸と虚軸上の根

Fig. 4 Real and pure imaginary roots for  $F_n(z)=0$ .

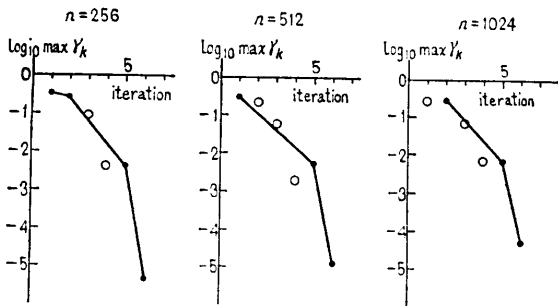


図5 反復(6),(7)による  $\log_{10} \max r_k$  の値  
 ・… $z_k (k=1, 2, \dots, n/4+1)$  に対する反復  
 ○…ある閾値を超えた  $z_k$  に対する反復

Fig. 5  $\log_{10} \max r_k$  by iteration of (6) and (7) for  
 ・ $z_k (k=1, 2, \dots, n/4+1)$ ,  
 ○ $z_k (k; r_k > a \text{ threshold value})$

これより大きい  $n$  については多倍長計算が必要である。また  $F_n(z)$  の係数は漸化式(4)で与えられるが、この計算で大きな桁落ち(ほぼ  $n/5$  桁)が起こることがわかった。したがって倍精度および4倍精度の計算に用いる係数を精度いっぱい入力するには、それぞれ4倍精度および多倍長計算をしなければならない。

多倍長計算のルーチンは自分で用意しなければならないが、計算時間がかかるのでこのような大規模な問題を解こうとする場合、メモリやプログラムの簡単さを犠牲にしても少しでも速いルーチン<sup>10)</sup>を用意する必要がある。また解法もできる限り時間を速める細かい検討が必要である。以下に具体的な留意点を述べる。

### 3.1 問題の特殊性の利用

#### 3.1.1 非常によい初期値を選び収束を速めること

DKA法は大域的収束性をもつが、初期値が根に近ければ収束は速い。この問題の根は曲線(5)に近づくことが予想されており  $n \leq 180$  までの根の様子からこのことがうかがえるので初期値をつぎのようにとる。

ある  $n$  に対する実軸上と虚軸上の根をそれぞれに得られた数値解を用いて  $1/\sqrt{n}$  の2次式で予測する。 $n=1024$  のときは  $n=480, 512, 640$  の数値解を用い実軸上の根を  $-0.71223087/n + 0.0070847165/\sqrt{n} + 0.99991391$  で(予測値  $0.99943977$ , 数値解  $0.9994413$ ) 虚軸上の根を  $-5.1247861/n - 0.2109110/\sqrt{n} + 0.52788351$  で(予測値  $0.51628787$ , 数値解  $0.5162541$ ) 予測しこれらの点から曲線(5)までの距離を  $L_R, L_I$  とする。 $n/4$  と  $n/2$  の一つおきの数値解を結ぶ直線と曲線(5)との交点を求めこの交点からこの直線上に  $L_R$  と  $L_I$  を用いて線形補間した点を取りこれを  $n$  に

対する三つおきの初期値とし残りは線形補間する。この初期値による収束の様子を図5に示す。この図からも非常によい初期値であることがわかる。図5の○で示される部分的反復については3.4節で述べる。

#### 3.1.2 $a_{2m-1}=0$ であることの利用

$n$  が奇数ならば一根本は0なので偶数のときに限る。 $w=z^2$  とおけば(3)は

$$G_{n/2}(w) = a_0 w^{n/2} + a_2 w^{n/2-1} + \dots + a_{n-2} w + a_n$$

という  $n/2$  次式となる。 $F_n(z)=0$  の根は  $G_{n/2}(w)=0$  の根を  $1/2$  乗して得られ、 $1/2$  乗の計算による精度の減少はないと考えられる。両方程式とも修正量を求める(7)式の分子の計算は同じものなので方程式の次数が減っても必要な計算桁数は変わらないから計算の手間は減らないが、分母の  $\prod_{j=1}^n (z_k^{(r)} - z_j^{(r)})$  と  $\prod_{j=1}^{n/2} (w_k^{(r)} - w_j^{(r)})$  では  $w$  とおいた方が掛け算の回数は半分になる。

#### 3.1.3 実係数であることの利用

係数はすべて実数なので、虚根は二つずつ互いに共役である。初期値をうまく選び  $w_1 \sim w_{n/4+1}$  の中に共役な根を含まないようにし、この  $n/4+1$  個の根の修正量だけ計算して共役なものもこの値で修正する。初期値を3.1.1項のように選べば、各反復で近似根は初期値の順序通りに並ぶので  $n/4+1$  個の中に共役なものは入ってこない。

このように  $n/4+1$  個の修正量だけ求めることにすれば、手間は約半分になる。念のため  $n=256$  までのものについて最後に全根に対して反復してみたが修正量も  $r_k$  も共役根では8桁目まで完全に一致した。

### 3.2 反復計算の計算桁数

$n$  により根の精度の限界が決まるので必要な計算桁数は決まるが、反復計算のすべてをこの桁数で行う必要はない。(7)式による場合はつぎのようになる。

$F_n(z)=0$  の根  $\alpha_k$  は  $1/2 < |\alpha_k| < 1$  ( $n \geq 320, k=1, 2, \dots, n$ ) で重根はないと予想されるから、 $G_{n/2}(w)=0$  の根  $\beta_k$  は  $1/4 < |\beta_k| < 1$  ( $n \geq 320, k=1, 2, \dots, n/2$ ) で重根はないと仮定できる。各反復での  $w$  の有効桁数は修正量で見積れる。 $r_k^{(r)} \geq 10^{-5}$  まででは(9)式から  $|\delta_k^{(r)}|$  は  $10^{-8}$  より大きく、有効数字は8桁もあれば十分である。

したがって

$$G_{n/2}(w) = (\dots((w + a_2)w + a_4)w + \dots)w + a_n$$

の計算で  $a_i (i=2, 4, \dots, n)$  は必要な計算桁数全部を精度よく入力しなければならないが、乗数  $w$  は有効数字

表 3  $\max r_k$  と CPU time の比較Table 3 Comparison of  $\max r_k$  and CPU time.

formula	(10)	(11)
$n$		
140	.2255 <sub>10</sub> -12	.1897 <sub>10</sub> -12
160	.1363 <sub>10</sub> -8 35 sec	.1165 <sub>10</sub> -8 72 sec
180	100 sec	more than 180 sec

8桁でよい。また  $\Pi(w_k - w_j)$  の計算でも8桁以上の桁落ちや積み残しはないから8桁演算でよい。

すなわち  $G_{n/2}(w)$  の計算だけを多倍長で、あとは全部単精度で計算すればよい。この効果を、i) すべての計算を同じ桁数の多倍長で、ii)  $w$  を有効数字8桁の多倍長にして全部多倍長で、iii)  $w$  について単精度と有効数字8桁の多倍長の変換を行い  $G_{n/2}(w)$  だけを多倍長であとは全部単精度で、計算した場合について比較すると、 $n=200$  のとき計算時間はii)はi)の約2/5に、iii)はii)の約3/5になる。この効果は  $n$  が増すほど大きい。

### 3.3 計算方法による手間の比較

#### 3.3.1 多項式の計算方法

一般に多項式を計算するときの二つの方法

$$P(z) = (\dots((z+a_1)z+a_2)z+\dots)z+a_n \quad (10)$$

$$P(z) = [(\dots((z+a_1)z+a_2)z+\dots+a_{m-1})z+a_m + (\dots((a_n/z+a_{n-1})/z+a_{n-2})/z + \dots + a_{m+1})/z] \times z^{n-m} \quad (11)$$

を比べると(11)式の方が計算途中での“overflow”，“桁落ちの度合”を少なくすることができる。

$F_n(z)$  については  $n=60 \sim 120$  まで  $r_k$  は4桁目まで全く一致した。4倍精度計算による  $n \geq 140$  についての結果を表3に示す。要求精度  $\epsilon=10^{-8}$  で反復回数は両方法とも同じである。

この表から  $F_n(z)$  に関しては(11)式を用いたとき、精度は反復を1回節約できるほどよくはならないが時間は約2倍になることがわかる。したがって overflow の心配がなく(多倍長ルーチンは指数部が大きい)要求精度  $10^{-5}$  ならば(10)式を用いる方がよい。

#### 3.3.2 2次法と3次法

2次法と3次法の手間を多倍長の乗法で見積ると、3次法の方が  $P'(w)$  の計算のため  $n/2-2$  回多くなる。これを2倍とみて  $r_k^{(v)}$  がある程度 ( $10^{-1}$  ぐらい) 小さくなった段階で要求精度をみたまでに必要な反復回数で時間を予測し、早い方を用いばよい。

表 4

Table 4

$n$	128	256	512	1024
size of array elements used by multiple-precision arithmetic for one iteration	5	8	14	27
estimated CPU time for one iteration	/	36.5 sec	192.5 sec	1182.6 sec
CPU time for one iteration	5.7 sec	27.5 sec	153.3 sec	993.4 sec
size of array elements used by multiple-precision arithmetic for computing the coefficients	9	15	28	54
estimated CPU time for computing the coefficients	/	18 sec	129 sec	1180 sec
CPU time for computing the coefficients	2.7 sec	17.3 sec	153 sec	1702 sec

### 3.4 部分的な反復

計算桁数によって定まる解の精度の限界まで達している根について反復計算を行わないのは当然であるが、そこまで達していなくても、半数以上の  $k$  についての  $r_k$  より  $10 \sim 10^2$  程度も大きい  $r_k$  をもつ近似根(この例では実軸に近い根)があれば2~3回これだけを反復し  $r_k$  が“ほぼ揃う”ようにしてから全体を反復する。

この方法による各反復段階での  $\max r_k$  が図5に示されている。計算桁数  $m$  のとき反復回数を増せば最悪のところでは  $m-n/5$  桁まで求められるが、そこまで必要でない場合には計算時間を節約するうえでこの部分的反復は非常に有効である。

### 3.5 計算時間の見積り

反復計算の手間を  $G_{n/2}(w)$  の計算での多倍長の乗法の回数で見積ると  $n(n/2-1)/4$  回では  $n^2$  に比例する。またここでの多倍長の乗法の手間は ( $w$  が8桁なので) 多倍長の配列の数に比例する。

係数の計算については、多倍長の除法の回数が  $n$  に比例し手間は(除数が4桁以内の整数なので)配列の数に比例する。多倍長の乗法は回数が  $n$  に比例し手間は  $(n/2)j$  ( $j=1, 2, \dots, n/2$ ) が整数になるものは少ないから配列の数の二乗に比例する。(配列の数)  $\ll n$  なので除法だけで考える。

$n/2$  の計算時間を用いた  $n$  に対する見積りと実際にかかった時間を表4に示す。

### 3.6 underflow の防止

$n$  が大きくなると  $\delta_k^{(v)}$  が  $10^{-3}$  程度の値でも(7)式の分子と分母は共に指数部が  $-76$  以下となり計算

できない。  $n=200$  で分子は  $10^{-28} \sim 10^{-31}$ , 分母は  $10^{-25} \sim 10^{-28}$  であるが,  $n=640$  では分子は  $10^{-87} \sim 10^{-90}$ , 分母は  $10^{-84} \sim 10^{-87}$  となり  $n=1024$  では分子は  $10^{-137} \sim 10^{-140}$  にもなる。

したがってこのままで指数部の大きい多倍長計算にするか, 適当な数を掛けて underflow を防がねばならない, 幸いにこの例では  $n$  が大きいとき  $1/4 < |\beta_k| < 1$  なので  $5/4$  ( $n=512$  まで) または  $25/16$  ( $n=1024$ ) を  $c$  として, おおのこの  $w_i - w_j$  に  $c$  を掛け分子には多倍長で  $c^{n/2-1}$  を掛けて underflow を防いだ。

#### 4. おわりに

この数値実験の結果から DKA 法は非常にすぐれた解法であることがわかる。また  $n$  を大きくしたとき曲線(5)に近づいていくことが数値的に確かめられた。

多倍長計算など大規模な計算を行う際には3章で述べたような工夫が必要で, 特に3.2節に述べた計算の各段階での, 与えられた問題に応じた計算桁数の検討は, むだな計算をしないためにも非常に重要である。

また, ここでは非常によい初期値を用いたので underflow に対する処置だけを考えれば済んだが, 一般的な DKA 法では多項式の計算を(11)式で行っても当然 overflow のおそれがありこの処置も必要になる。この例でも計算機の system に指数部の大きい数が備えられていれば便利であると痛感した。

謝辞 東京大学教授伊理正夫博士にはこの研究を行う機会をお与えくださりご懇篤なるご指導を賜わった。また電子技術総合研究所戸田英雄博士には本研究を進めるにあたり種々便宜をおはかりいただき絶大な

るご援助を賜わった。ここに深甚なる感謝の意を表す。なおこの一部は文部省昭和53年度科学研究費助金によった。

#### 参考文献

- 1) Kerner, I. O.: Ein Gesamtschrittenverfahren zur Berechnung der Nullstellen von Polynomen Numer. Math., Bd. 8 pp. 290-294 (1966).
- 2) Smith, B. T.: Error Bounds for Zeros of a Polynomial Based upon Gerschgorin's Theorems, J. ACM., Vol. 17, pp. 661-674 (1970).
- 3) 山本哲朗: ある代数方程式解法と解の事後評価法, 数理科学, Vol. 14, No. 7 pp. 52-57 (1976).
- 4) 山本哲朗, 古金卯太郎, 野倉久美: 代数方程式を解く Durand-Kerner 法と Aberth 法, 情報処理, Vol. 18, No. 6, pp. 566-571 (1977).
- 5) 伊理正夫, 山下 浩, 寺野隆雄, 小野令美: 大域的収束性をもつ代数方程式の解法, 京大数理解析研究所研究集会 (1977年10月31日-11月2日).
- 6) Graside, G. R., Jarratt, P. and Mack, C.: A New Method for Solving Polynomial Equations Comput. J., Vol. 11, pp. 87-90 (1968).
- 7) P. ヘンリッチ著/一松 信, 平本 巖, 本田勝 共訳: 数値解析の基礎, 培風館 (1973).
- 8) 大中幸三郎: 多項式の零点の決定論的評価, 情報処理, Vol. 19, No. 1, pp. 26-31 (1978).
- 9) Ralston, A.: A First Course in Numerical Analysis, McGraw Hill (first edition) (1965).
- 10) 戸田英雄, 小野令美: 高精度計算用の除算のアルゴリズムに関して, 電子技術総合研究所彙報, Vol. 42, No. 4, pp. 66-71 (1978).

(昭和54年1月9日受付)

(昭和54年4月19日採録)