多倍長演算のための平方根の高速計算法†

小沢 一文††

本論文では，平方根を近似する高次収束法のアルゴリズム群を提案し，それらの多倍長演算における計算方法と計算効率について考察している．ここで提案されたアルゴリズム群は，収束の次数を任意に高く取ることができ，また初期値に無関係に収束するという特徴を持っている．このアルゴリズム群は，特別な場合として収束の次数が2の場合はニュートン法になり，3の場合は Bailey 法となる．反復関数はどの場合も有理関数となり，この有理関数を多倍長演算でより高速に計算するため，いくつかの計算方法を提案している．ここで提案された計算法の時間計算量を詳細に検討した結果，常に一定の桁数で計算する「固定長演算」では，多倍長数の平方根を計算するときは，5次収束法を2次因子に分解する方法が最も高速であり，単長数（単精度数）の平方根を計算するときは，平方根の逆数に2次収束する方法が最も高速であることが判明した．一方，計算桁数を反復値の精度に応じて変更していく「可変長演算」でも，平方根の逆数に2次収束する方法が最も高速であることが判明した．また「可変長演算」では，いかなる次数の解法あるいはいかなる計算法を用いたとしても，「固定長演算」で計算した場合の高々反復2回分で計算が完了するという結論も得られている．以上の結論は，これまでに知られているどのような乗算法にも妥当するものである．

1. はじめに

平方根は，あらゆる科学技術計算のなかで最も基本的な演算であり，また使用頻度の最も高い初等関数でもある．したがって，平方根をより高速かつ高精度で計算するアルゴリズムを導出することは，科学技術計算全体に寄与するところ大であろう．

通常の単精度，倍精度そして4倍精度のライブラリプログラムでは，平方根はニュートン法によって計算されている．良く知られているように，ニュートン法は2次収束性を示すため，少なくとも「要求桁数の半分の精度」を持った初期値を与えておけば，反復はたった1回で済むことになる．このように，与えられた初期値の精度が十分に良い場合，すなわち2次収束法による反復が1回で済む場合，反復式がより複雑になる3次以上の高次解法を使うことは，明らかに無駄なことである．

一方，何百桁，何千桁といった超高精度が要求されるとき，「要求桁数の半分の精度」を持つ初期値を簡単な手続きによって求めることは不可能であろう．このような場合，例えば倍精度あるいは4倍精度の組込み関数を用いて数十桁程度の精度の初期値を与え，ニュートン法あるいはそれ以上の高次解法を多数回反復することになる．一般に，解法の次数が高くなればなるほど，反復1回当たりのコストは大きくなるが，その一方で反復回数が少なくなるため，多数回の反復が必要な超高精度計算の場合には，高次解法が必ずしも不利だとはいえなくなる．

平方根の超高精度計算のアルゴリズムに関しては，これまでに Dutka[1] は平方根の連分数近似を拡張した2次収束法を提案し，それを用いて $\sqrt{2}$ を実際に100万桁計算した．また，小沢，海野[9] は Dutka[1] のアルゴリズムをより高次のものへと拡張してきた．しかし，これらは整数の平方根のみに適用可能なアルゴリズムである．一方，金田[10]は $\sqrt{2}$ を1600万桁計算した実績があるが，このときに用いたアルゴリズムは平方根の逆数に収束するニュートン法である．このアルゴリズムは任意の実数に適用可能であるが，収束の次数は2次である．これに対して，本論文で提案するアルゴリズム群は，任意の実数に対して適用可能なものであり，また収束の次数も任意に高く取れるという特徴がある．

本論文では，この高次収束法のアルゴリズム群の多倍長浮動小数点演算における効率的な計算法を考察し，最高速な解法を導出する．

2. 1次収束法

はじめに，$a>0$ の平方根に収束する1次収束列を導出する．まず次の差分方程式を考える：

$$U_m=2uU_{m-1}-(u^2-av^2)U_{m-2},$$
$$V_m=2uV_{m-1}-(u^2-av^2)V_{m-2},$$
$$m=2,3,\cdots \qquad (2.1)$$

方程式(2.1)の特性根は，

$$\lambda_1=u+\sqrt{a}v,\quad \lambda_2=u-\sqrt{a}v \qquad (2.2)$$

である．これを用いて，U_m, V_m の一般解を表すと

† Fast Multiple-Precision Calculation of Square Root by KAZUFUMI OZAWA (Department of Information Engineering, Sendai National College of Technology).

†† 仙台電波工業高等専門学校情報工学科

$$U_m=c_{11}\lambda_1^m+c_{12}\lambda_2^m,$$
$$V_m=c_{21}\lambda_1^m+c_{22}\lambda_2^m, \quad (2.3)$$
$$m=0,1,2,\cdots$$

である．ここで，

$$uv>0 \quad (2.4)$$

を仮定すれば，特性根の間に，

$$|\lambda_1|>|\lambda_2| \quad (2.5)$$

という関係が成立する．このとき，(2.3)式に現れる係数が，例えば

$$c_{11}=c_{12}=1/2,$$
$$c_{21}=-c_{22}=1/(2\sqrt{a}) \quad (2.6)$$

を満たせば，数列 $\{U_m/V_m\}$ は u，v に無関係に $\sqrt{a}$ に収束することが直ちに示される．条件(2.6)と矛盾しないU，Vの初期値は，

$$U_0=1, \quad U_1=u, \quad (2.7)$$
$$V_0=0, \quad V_1=v$$

である．以下，条件(2.4)が成り立ち，初期値を上式のように定めた場合のみを考察する．なお，aが非平方な整数で，しかも

$$u^2-av^2=\pm 1$$

となるように u，v を定めたとき，漸化式(2.1)は $\sqrt{a}$ の連分数近似を与えている[2),9)]．

次に $W_m=U_m/V_m$ の1次収束性を簡単に示す．W_m の持つ誤差 e_m は，(2.3)，(2.6)式より

$$e_m=W_m-\sqrt{a}$$
$$=\frac{2\mu^m}{1-\mu^m}\sqrt{a}, \quad m=1,2,3,\cdots \quad (2.8)$$

として評価される．ここで，$\mu=\lambda_2/\lambda_1$ と置いた．この式より，

$$\lim_{m\to\infty} e_m/e_{m-1}=\mu \quad (2.9)$$

となり，W_m は $\sqrt{a}$ に漸近的に1次収束することが示される．

3. 高次収束法の導出

本章では，1次収束列(2.1)を加速し $\sqrt{a}$ に高次収束するような数列を生成するアルゴリズムを導出する．

(2.3)式に(2.2)，(2.7)式を代入すると，U_m，V_m は

$$U_m=\sum_i\binom{m}{2i}a^i u^{m-2i}v^{2i}\equiv A_m(u,v), \quad (3.1)$$

$$V_m=\sum_i\binom{m}{2i+1}a^i u^{m-2i-1}v^{2i+1}\equiv B_m(u,v) \quad (3.2)$$

となる．ここで二項係数は，

$$\binom{i}{j}=0, \quad i<j, \text{ または } j<0$$

と約束し，Σは二項係数が0にならない範囲でとるものとする．これより，$W_m=U_m/V_m$，$m=1,2,\cdots$ を $W_1=u/v$ によって表すと，

$$W_m=G_m(W_1) \quad (3.3)$$

となる．ここで，

$$G_m(x)=A_m(x)/B_m(x), \quad m=1,2,\cdots \quad (3.4)$$
$$A_m(x)=A_m(x,1) \quad (3.5)$$
$$B_m(x)=B_m(x,1) \quad (3.6)$$

である．

ところで，恒等式

$$\sum_i\binom{m}{2i}=\sum_i\binom{m}{2i+1}=2^{m-1} \quad (3.7)$$

を用いれば，有理関数 $G_m(x)$ が，任意の $m\geqq 1$ に対して，等式

$$G_m(\sqrt{a})=\sqrt{a} \quad (3.8)$$

を満たしていること，すなわち，$\sqrt{a}$ が $G_m(x)$ の不動点になっていることがわかる．さらに $G_m(x)$ は次の定理を満たしている：

定理1

任意の整数 $r>1$ に対して，有理関数 $G_r(x)$ によって不動点反復公式

$$x_{k+1}=G_r(x_k), \quad x_0>0, \quad (3.9)$$
$$k=0,1,2,\cdots$$

を定義すると，数列 $\{x_k\}$ は $\sqrt{a}$ に漸近的に r 次収束する．

[証明]

数列 $\{x_k\}$ の r 次収束性を証明するには，(3.8)式がすでに証明されているから，

$$G_r^{(j)}(\sqrt{a})=0, \quad j=1,2,\cdots,r-1, \quad (3.10\text{-}1)$$
$$G_r^{(r)}(\sqrt{a})\neq 0 \quad (3.10\text{-}2)$$

を証明すれば十分である[3)]．そこで，(3.10-1)式を数学的帰納法によって証明する．まず，$G_r'(\sqrt{a})=0$ は以下のよう証明される．$\omega=\sqrt{a}$ と置き，式

$$G_r'(x)=[A_r'(x)B_r(x)-A_r(x)B_r'(x)]/B_r^2(x)$$

にωを代入すると，

$$G_r'(\omega)=2^{r-1}\omega^{2(r-1)}\sum_{i=0}\left\{(r-2i)\binom{r}{2i}\right.$$
$$\left.-(r-2i-1)\binom{r}{2i+1}\right\}\Big/B_r^2(\omega)$$
$$=\frac{2^{r-1}\omega^{2(r-1)}r-1}{B_r^2(\omega)}\sum_{j=0}(-1)^{j+1}\binom{r-1}{j}$$
$$=0 \quad (3.11)$$

が得られる．ここで，$k<r$ を満たす k に対して

$$G_r'(\omega)=G_r''(\omega)=\cdots=G_r^{(k-1)}(\omega)=0 \qquad (3.12)$$

を仮定し，$G_r^{(k)}(\omega)=0$ を証明する．帰納法の仮定(3.12)より $A_r^{(k)}(\omega)$ を計算すると，

$$A_r^{(k)}(\omega)=\sum_{i=0}^{k}\binom{k}{i}G_r^{(i)}(\omega)B_r^{(k-i)}(\omega)$$
$$=\omega B_r^{(k)}(\omega)+G_r^{(k)}(\omega)B_r(\omega) \qquad (3.13)$$

を得る．これより，

$$G_r^{(k)}(\omega)=\{A_r^{(k)}(\omega)-\omega B_r^{(k)}(\omega)\}/B_r(\omega) \qquad (3.14)$$

となる．上式に，

$$A_r^{(k)}(\omega)=k!\omega^{r-k}\binom{r}{k}\sum_i\binom{r-k}{2i} \qquad (3.15)$$

$$B_r^{(k)}(\omega)=k!\omega^{r-k-1}\binom{r}{k}\sum_i\binom{r-k}{2i+1} \qquad (3.16)$$

を代入すると，

$$G_r^{(k)}(\omega)=\frac{k!\omega^{r-k}}{B_r(\omega)}\binom{r}{k}\sum_{j=0}^{r-k}(-1)^j\binom{r-k}{j}$$
$$=0 \qquad (3.17)$$

が示される．よって，(3.10-1)式が証明された．

次に(3.10-2)式を証明する．(3.14)式を導いたのと同様にして，

$$G_r^{(r)}(\omega)=\{A_r^{(r)}(\omega)-\omega B_r^{(r)}(\omega)\}/B_r(\omega) \qquad (3.18)$$

が示される．ここで，A_r は r 次の多項式で最高次の係数が1であるから，

$$A_r^{(r)}(\omega)=r! \qquad (3.19)$$

となり，また B_r は $r-1$ 次の多項式であるから，

$$B_r^{(r)}(\omega)=0 \qquad (3.20)$$

となる．よって，

$$G_r^{(r)}(\omega)=r!/(2^{r-1}\omega^{r-1})\neq 0 \qquad (3.21)$$

が得られる． (証明終)

この定理より，反復公式(3.9)の誤差 $e_k(=x_k-\sqrt{a})$ は，

$$e_{k+1}=\frac{1}{2^{r-1}\omega^{r-1}}e_k{}^r+O(e_k{}^{r+1}), \qquad k\to\infty \qquad (3.22)$$

となることがわかる．

次の定理は，r 次収束列 $\{x_k\}$ と(2.1)式で定められる1次収束列 $\{W_m\}$ の関係を示している：

定理2

適当な u，v に対して初期値を

$$x_0=U_1/V_1=u/v>0 \qquad (3.23)$$

と定めたとき，(3.9)式によって定義される数列 $\{x_k\}$ は，条件

$$x_k=W_{r^k}=U_{r^k}/V_{r^k} \qquad k=0,1,2,\cdots \qquad (3.24)$$

を満たす．ただし，$\{U_k\}$，$\{V_k\}$ は(2.1)式を初期値(2.7)によって計算した値である．

[証明]

ここで，2つの等式

$$A_r(U_k,V_k)\pm\omega B_r(U_k,V_k)=(U_k\pm\omega V_k)^r \qquad (3.25)$$

が成り立っていることに着目すると，

$$A_r(U_k,V_k)$$
$$=\{(U_k+\omega V_k)^r+(U_k-\omega V_k)^r\}/2$$
$$=(\lambda_1{}^{r^k}+\lambda_2{}^{r^k})/2$$
$$=U_{r^k} \qquad (3.26)$$

$$B_r(U_k,V_k)$$
$$=\{(U_k+\omega V_k)^r-(U_k-\omega V_k)^r\}/(2\omega)$$
$$=(\lambda_1{}^{r^k}-\lambda_2{}^{r^k})/2$$
$$=V_{r^k} \qquad (3.27)$$

という結果が得られる．これより，

$$W_{r^k}=U_{r^k}/V_{r^k}=\frac{A_r(U_k,V_k)}{B_r(U_k,V_k)}$$
$$=\frac{A_r(W_k,1)}{B_r(W_k,1)}$$
$$=\frac{A_r(W_k)}{B_r(W_k)}$$
$$=G_r(W_k) \qquad (3.28)$$

が得られる．したがって，

$$x_1=G_r(x_0)=G_r(W_1)=W_r$$
$$x_2=G_r(x_1)=G_r(W_r)=W_{r^2}$$
$$x_k=G_r(x_{k-1})=G_r(W_{r^{k-1}})=W_{r^k}$$

となり定理が証明された． (証明終)

以上の2つの定理より，反復法(3.9)は $x_0>0$ を満たす任意の初期値から出発しても収束し，また漸近的に r 次収束性を示すことが得られた．

最後に，r 次収束法の例をいくつか示す：

(1) $r=2$ とすると，

$$x_{k+1}=(x_k{}^2+a)/(2x_k)$$

となり，ニュートン法となる．

(2) $r=3$ とすると，

$$x_{k+1}=(x_k{}^3+3ax_k)/(3x_k{}^2+a)$$

となり，$\sqrt{a}$ を近似する Bailey 法[3] となる．

4. 高次収束法とその計算効率 (1)

本章では，r 次収束法(3.9)の時間計算量を求め，何次の解法が最も高速かを考察する．ここでは，(3.9)式の r 次収束法のほかに，$1/\sqrt{a}$ に収束する2次収束法，

$$x_{k+1}=x_k+\frac{x_k(1-ax_k{}^2)}{2} \qquad (4.1)$$

を用いて $\sqrt{a}$ の近似を求める方法—この方法をここ

では「逆平方根法」とよぶことにするーも考察の対象とする．以下，$\sqrt{a}$ の t 桁近似を初期値 x_0 とし，これをもとに r 次収束法を p 回反復し，$\omega=\sqrt{a}$ の s 桁 $(s=r^p t)$ 近似を得るのに要する時間計算量 $T_r(t;s)$ を求める．演算は多倍長浮動小数点演算とする．

多倍長演算による各種アルゴリズムの時間計算量を考察するにあたって，それらアルゴリズムに含まれる多倍長数同士の乗除算だけに注目し，他の演算，例えば多倍長数の加減算，多倍長数と単長数の乗除算等の時間計算量は微小であると考え無視する．ここでは，演算桁数は N 桁 $(t<s\leqq N)$ に固定されているものとするーこれをここでは「固定長演算」とよぶことにする．

乗算（以下特に断らないかぎり「多倍長数×多倍長数」という演算を乗算とよぶことにする）1回当たりの時間計算量を $M(N)$，反復1回当たりの乗算の実質的な回数（乗算1回を1と数え，除算，二乗演算はそれらの乗算に対する時間比のウエイトをつけて回数を合計したもの）を c_r で表せば，

$$T_r(t;s)=pc_rM(N)$$
$$=\log(s/t)M(N)(c_r/\log r) \qquad (4.2)$$

となる．この式より，r に依存する部分，すなわち $c_r/\log r$ が全時間を決定することになる．以下，多項式 A_r，B_r の計算方法をいくつか提案し，それらの実質的な乗算回数 c_r の値を求め，時間計算量を比較検討する．これ以降，多項式 A_r，B_r の係数を a_j，b_j とし簡略化して表す．

（1） Horner の方法

多項式の計算は，因数分解などの前処理を許さないとすれば，Horner の方法が最適であることが知られている[4]．Horner の方法の実質的な乗算回数を考察する．例えば，$r=2m$ のとき $A_r(x)$，$B_r(x)$ を Horner の方法にて計算すると，

$$A_r(x)=(((X+a_1)X+a_2)\cdots+a_{m-1})X+a_m, \qquad (4.3)$$

表1 Horner の方法の演算量
Table 1 Total number of operations in Horner method.

演算	回数	時間比
乗算	$r-2$	1
二乗演算	1	0.5
除算	1	d

合計 $c_r=r-1.5+d$

$$B_r(x)=x((((rX+b_1)X+b_2)\cdots+b_{m-2})X+b_{m-1}), \qquad (4.4)$$

$$X=x^2$$

となるから，乗算は定数 r を単長数とすると，$2m-2=r-2$ 回となる．同様な考察を行うことによって $r=2m+1$ のときも $r-2$ 回の乗算が必要になることがわかる．その他，$X=x^2$ で二乗演算（後述）1回，$G_r=A_r/B_r$ で除算1回を要する．

（2） 2次因子分解法

多倍長演算においては，多倍長数を表す各要素(桁)間の対称性を利用することによって，二乗の演算を乗算の約半分の時間で計算することができる（このようにして二乗を計算する方法をここでは「二乗演算」と呼ぶことにする）．そこで，できるかぎり多く二乗演算を取り入れることによって，多項式の計算を高速化する方法を提案する．

まず2次式 $X^2+\alpha X+\beta$ の計算を考える．これを Horner の方法で計算すると1回の乗算を要する．これに対して，この2次式を

$$(X+\alpha/2)^2-\alpha^2/4+\beta$$

と変形し，$(X+\alpha/2)^2$ の計算に二乗演算を用いたとすると，定数 $\alpha/2$，$-\alpha^2/4$ の計算に要する時間が無視できるような状況においては，この方法を用いると二乗演算1回を要するから，Horner の方法の約半分の時間で計算が完了することになる．この定数の計算時間が無視できるような状況とは，例えば，多くの X について同じ2次式を何回も計算する場合がこれにあたる．

ここでは，多項式 A，B を2次因子に分解し，各2次因子の計算にいま述べた方法を適用し，時間計算量を減少させる方法ーこれをここでは「2次因子分解法」と名づけるーを解析する．

例えば，$r=4m+2$ のとき多項式 A，B を2次因子に分解すると

$$A_r(x)=X\prod_{i=1}^{m}(X^2+\alpha_iX+\beta_i)+a_{2m+1} \qquad (4.5)$$

$$B_r(x)=rx\prod_{i=1}^{m}(X^2+\gamma_iX+\delta_i) \qquad (4.6)$$

であるから，上で述べた方法で2次因子を計算すると，反復1回当たり，$X=x^2$ の計算を含めて二乗演算 $2m+1$ 回，乗算 $2m-1$ 回，除算1回を要することになる．

r が大きいとき，多項式 A，B を(4.5)，(4.6)式のように2次因子に分解することは，必ずしも容易なこ

表 2.1 2次因子分解法の演算量
Table 2.1 Total number of operations in Square decomposition method.

$r=4m$			$r=4m+1$		
演算	回数	時間比	演算	回数	時間比
乗算	$2m-1$	1	乗算	$2m-1$	1
二乗演算	$2m$	0.5	二乗演算	$2m+1$	0.5
除算	1	d	除算	1	d
合計	$c_r=3m-1+d$		合計	$c_r=3m-0.5+d$	
	$=3r/4-1+d$			$=3r/4-5/4+d$	

表 2.2 2次因子分解法の演算量
Table 2.2 Total number of operations in Square decomposition method.

$r=4m+2$			$r=4m+3$		
演算	回数	時間比	演算	回数	時間比
乗算	$2m$	1	乗算	$2m+1$	1
二乗演算	$2m+1$	0.5	二乗演算	$2m+1$	0.5
除算	1	d	除算	1	d
合計	$c_r=3m+0.5+d$		合計	$c_r=3m+1.5+d$	
	$=3r/4-1+d$			$=3r/4-3/4+d$	

とではないが，仮にこのような2次因子による分解が可能であったとして，時間計算量の解析を行った結果を**表2**に示す．

(3) Todd の4次式[4]

$r=9$ の場合を考える．$r=9$ のとき，A_9 は $x\times$(X の4次式)，B_9 は X の4次式となる．この2つの X の4次式を Todd の4次式に変形する（このような変形は常に可能である）：

$$A_9(x)=x[\{(X-\alpha)^2+\beta\}\{(X-\alpha)^2+X+\gamma\}+\delta], \tag{4.7}$$

$$B_9(x)=9\{(X-\varepsilon)^2+\zeta\}\{(X-\varepsilon)^2+X+\eta\}+\theta, \tag{4.8}$$

$$X=x^2$$

ここで，a_j，b_j から上式の定数を求めると，

$$\begin{aligned}
&\alpha=1/4-9a,\\
&\beta=1/16-180a^2+3648a^3,\\
&\gamma=1/16+9a-180a^2-3648a^3,\\
&\delta=-1/64+45a^2-42624a^4+13307904a^6,\\
&\varepsilon=1/4-7a/3,\\
&\zeta=1/16-28a^2/3+1088a^3/27,\\
&\eta=1/16+7a/3-28a^2/3-1088a^3/27,\\
&\theta=-9/64+21a^2-2944a^4/3+1183744a^6/81
\end{aligned} \tag{4.9}$$

となる．このように変形することによって，乗算3回，二乗演算3回，除算1回となり，前の2次因子分解法に比べ，二乗演算の回数を2回減らすことができる．この方法は $r=8m+1$ のとき多項式 A は $x\times$(X の $4m$ 次式)，B は X の $4m$ 次式となり都合がよい．しかし，このような一般の r に対して，A，B を4次式の積に分解し，各4次式を Todd の4次式に変形することは容易なことではないが，仮にそれが可能であったとすれば，乗算 $4m-1$ 回，二乗演算 $2m+1$ 回，除算1回とすることができる．

表 3 Todd の4次式を用いた方法の演算量 ($r=8m+1$)
Table 3 Total number of operations in Todd's quartic polynomial method ($r=8m+1$).

演算	回数	時間比
乗算	$4m-1$	1
二乗演算	$2m+1$	0.5
除算	1	d
合計	$c_r=5m-0.5+d=5r/8-9/8+d$	

(4) Knuth の方法[4]

前処理を許すとき，多項式の効率的な計算方法として Knuth の方法というのがよく知られている．この方法は，n 次の多項式 $p_n(X)$ を奇数ベキからなる多項式 $R(X)$ と偶数ベキからなる多項式 $P(X)$ とに分け，すなわち，

$$p_n(X)=X\cdot R(Y)+P(Y) \tag{4.10}$$
$$Y=X^2$$

とし，$R(Y)$ を因数分解し $P(Y)$ から $R(Y)$ の因子をできるだけくくりだし，多項式 R，P をまとめて計算していく方法である．この方法を用いると，最高次の係数が1のとき（あるいは単長数で多倍長数との乗算の時間が無視できるようなときも同様である），次数 $n=2m$ あるいは $n=2m+1$ 次の多項式の計算に，乗算 m 回と二乗演算1回を要することが知られている[4]．この方法を多項式 A_r，B_r の計算に応用する．

例えば，$r=7$ の場合 $Y=X^2$ と置いて，

$$\begin{aligned}
A_7(x)&=x(X^3+21aX^2+35a^2X+7a^3)\\
&=x\{X(Y+35a^2)+(21aY+7a^3)\}\\
&=x\{(Y+35a^2)(X+21a)-728a^3\},
\end{aligned} \tag{4.11}$$

$$\begin{aligned}
B_7(x)&=7X^3+35aX^2+21a^2X+a^3\\
&=X(7Y+21a^2)+35aY+a^3\\
&=(7Y+21a^2)(X+5a)-104a^3
\end{aligned} \tag{4.12}$$

となり，乗算3回，二乗演算2回，除算1回となる．

次に一般の r について考える．$r=4m+1$ とすると，$A_r(x)$，$B_r(x)$ は

$$A_r(x)=x(X^{2m}+a_1X^{2m-1}+\cdots+a_{2m})$$
$$B_r(x)=(rX^{2m}+b_1X^{2m-1}+\cdots+b_{2m})$$

表 4.1 Knuth の方法の演算量
Table 4.1 Total number of operations in Knuth method.

$r=4m$

演算	回数	時間比
乗算	$2m$	1
二乗演算	2	0.5
除算	1	d
合計	$c_r=2m+1+d$ $=r/2+1+d$	

$q=4m+1$

演算	回数	時間比
乗算	$2m+1$	1
二乗演算	2	0.5
除算	1	d
合計	$c_r=2m+2+d$ $=r/2+1.5+d$	

表 4.2 Knuth の方法の演算量
Table 4.2 Total number of operations in Knuth method.

$r=4m+2$

演算	回数	時間比
乗算	$2m+1$	1
二乗演算	2	0.5
除算	1	d
合計	$c_r=2m+2+d$ $=r/2+1+d$	

$r=4m+3$

演算	回数	時間比
乗算	$2m+1$	1
二乗演算	2	0.5
除算	1	d
合計	$c_r=2m+2+d$ $=r/2+0.5+d$	

であるから，( )の中を計算するのに要する演算の回数は，二乗演算 2 回 ($X=x^2$ と，$Y=X^2$)，乗算 $2m+1$ 回 ($A(x)$ に $m+1$ 回，$B(x)$ に m 回)，除算 1 回，となる．その他の r について同様の解析を行った結果を表 4 に示す．

(5) Paterson-Stockmeyer 法[4]

Paterson-Stockmeyer の方法というのは，多項式の演算において定数倍の演算が無視できるとき，m次の多項式が $O(\sqrt{m})$ の時間計算量で計算できる最も高速な方法である．以下，Paterson-Stockmeyer 法を用いて多項式 $A(x)$, $B(x)$ を計算する方法について考察する．

$r=2m+1$ の場合を考える．

$$A_r(x)=x(X^m+a_1X^{m-1}+\cdots+a_m)$$
$$B_r(x)=(rX^m+b_1X^{m-1}+\cdots+b_m)$$

となる．ここで，X の m 次多項式 $A_r'=A_r/x$ と B_r にこの方法を適用する．まず，

$$k=\lceil\sqrt{(m+1)}\rceil,\quad i=\lfloor m/k\rfloor,\quad p=m-ik$$

とし，$Y=X^k$ と置いて A', B を

$$\begin{aligned}A'_r(X)=&(X^p+a_1X^{p-1}+\cdots+a_p)Y^i\\&+(a_{p+1}X^{k-1}+a_{p+2}X^{k-2}+\cdots+a_{p+k})Y^{i-1}\\&\cdots\\&+(a_{m-k+1}X^{k-1}+a_{m-k+2}X^{k-2}+\cdots+a_m)\end{aligned}\tag{4.13}$$

$$B_r(X)=(rX^p+b_1X^{p-1}+\cdots+b_p)Y^i$$

表 5.1 Paterson-Stockmeyer の方法の演算量 (a=単長数，$r=2m$)
Table 5.1 Total number of operations in Paterson-Stockmeyer method (a=single length number, $r=2m$).

演算	回数	時間比
乗算	$i+j-\delta_p-\delta_q+k-1$	1
二乗演算	2	0.5
除算	1	d
合計	$c_r=i+j-\delta_p-\delta_q+k+d$	

表 5.2 Paterson-Stockmeyer の方法の演算量 (a=単長数，$r=2m+1$)
Table 5.2 Total number of operations in Paterson-Stockmeyer method (a=single length number, $r=2m+1$).

演算	回数	時間比
乗算	$2(i-\delta_p)+k-1$	1
二乗演算	2	0.5
除算	1	d
合計	$c_r=2(i-\delta_p)+k+d$ $(k=\lceil\sqrt{m+1}\rceil,\ i=\lfloor m/k\rfloor,\ j=\lfloor(m-1)/k\rfloor,\ p=m-ik,\ q=m-1-jk)$	

$$\begin{aligned}&+(b_{p+1}X^{k-1}+b_{p+2}X^{k-2}+\cdots+b_{p+k})Y^{i-1}\\&\cdots\\&+(b_{m-k+1}X^{k-1}+b_{m-k+2}X^{k-2}+\cdots+b_m)\end{aligned}\tag{4.14}$$

と変形する．計算手順は，

(i) 始めに二乗演算を用いてXを二乗し，

$$\begin{aligned}X^3&=X^2X\\X^4&=X^3X\\&\vdots\\Y&=X^k=X^{k-1}X\end{aligned}\tag{4.15}$$

として，Xの必要なベキを求め，

(ii) これらを用いて(4.13)，(4.14)式の( )の中を計算し，

(iii) 最後にYのi次多項式A'，Bを Horner の方法で計算する，

というものである．ここで，aが単長数であると仮定すると，係数 a_j, b_j も単長数となり(4.13)，(4.14)式の( )内の計算に要する時間は無視できる．この計算法では，全体で，乗算 $2(i-\delta_p)+k-1$ 回，二乗演算2回 ($X=x^2$ と X^2)，除算1回となる．ここで，δ_p は

$$\delta_p=\begin{cases}1; & p=0,\\ 0; & p>0\end{cases}\tag{4.16}$$

となる関数である．この方法では，r が十分大きいとき乗算回数はおおよそ $2k\fallingdotseq\sqrt{2r}$ 回で済むことに注意

する．

いまここで a が単長数であると仮定したが，a が多倍長数であると仮定すると，上に示した演算のほかに (4.13)，(4.14)式の( )内の計算に要する演算が加わり，全体で乗算

$$2(i-\delta_p)+k-1+2[\max\{p-1,0\}+i(k-1)]$$ 回

となる（二乗演算と除算は同じ）．

$r=2m$ の場合についても同様の解析をした結果を**表5**にまとめる．

（6） 逆平方根法(4.1)について

この解法は，r 次収束法(3.9)と異なり式が単純なため計算方法を工夫する必要はない．また，Paterson-Stockmeyer の方法で r 次収束法を計算した場合と同様に，a が多倍長数であるか単長数であるかによって演算回数が異なってくる．

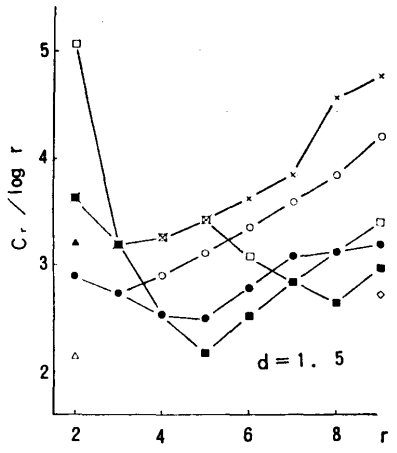
直ちにわかるように，(4.1)式をそのまま計算すると，a が多倍長数のときは，乗算2回，二乗演算1回，除算0回であり，a が単長数のときは乗算1回，二乗演算1回，除算0回である．この解法では，最後に $1/\sqrt{a}$ の近似値 x_k に a を掛けて $\sqrt{a}$ の近似値を求めるのだが，解法の次数が2と低いため，超高精度計算のときは反復回数が多くなり，最後に a を掛ける演算の時間は無視できる．

［数値例1］

乗算に対する除算の時間比 d を $d=1.5$, 2.5, 4.0, 10.0 と変えていき，各々の d に対して時間計算量(4.2)を決定する値 $c_r/\log r$ の理論値の変化を**図1～4**に示す．これらの図より，a が単長数のときは，除算を含まない逆平方根法が非常に高速で，$d=1.5$ のとき5次の Paterson-Stockmeyer 法が辛うじてこれに匹敵する程度であることがわかる．逆に a が多倍長数で d の値が比較的小さいとき，除算を含む他の解法のなかにも逆平方根法より高速なものがいくつか存在することがわかる．例えば，2次因子分解法 ($r=3$, 4, 5, 6)，Todd の方法 ($r=9$)，Knuth の方法 ($r=7$) 等である．また $d=10$ のとき，除算を含む方法は，a が多倍長であるか単長であるかにかかわらず，どれも逆平方根法にはとても及ばないことがわかる．

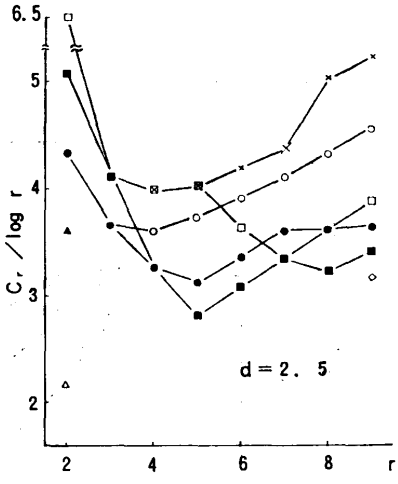
［数値例2］

次に，実際に多倍長演算を用いて計算した結果を示す．ここで用いた多倍長演算のソフトウェアは，乗算の時間計算量が $O(N^2)$ となるものであり，二乗演算：乗算：除算の時間比の実測値は，0.49：1.0：2.5 である．すなわち $d=2.5$ である．ここでは $a=2$（単長数）と $a=0.3333\cdots$（多倍長数）の2通りの場合を扱う．初期値は相対誤差がおおよそ 3×10^{-8} 程度と



—○— Horner —×— P.S. (M)
—●— 2次因子分解 —■— P.S. (S)
—△— 逆平方根(S) —◇— Todd
—▲— 逆平方根(M) —□— Knuth
P.S.: Paterson-Stockmeyer
M: a=Multiple-Precision
S: a=Single Precision

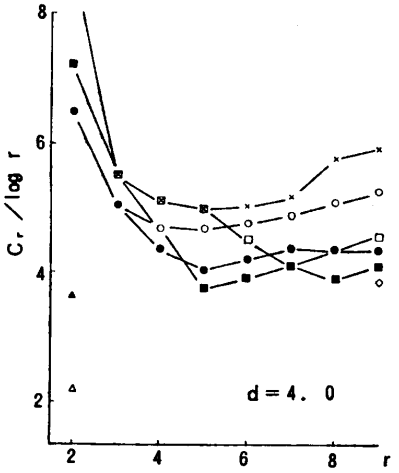
図1 r 次収束法の時間計算量 ($d=1.5$)
Fig. 1 Time complexity of r-th order converging method ($d=1.5$).



—○— Horner —×— P.S. (M)
—●— 2次因子分解 —■— P.S. (S)
—△— 逆平方根(S) —◇— Todd
—▲— 逆平方根(M) —□— Knuth
P.S.: Paterson-Stockmeyer
M: a=Multiple-Precision
S: a=Single Precision

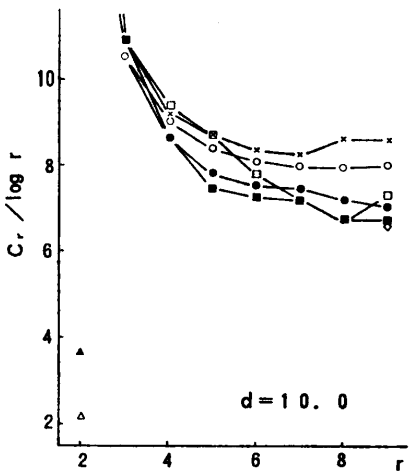
図2 r 次収束法の時間計算量 ($d=2.5$)
Fig. 2 Time complexity of r-th order converging method ($d=2.5$).

なるものを選び，演算時間と $\log_{10}$（|相対誤差|）の関係を図5に図示する．ここで用いた解法は，[数値例1]で $d=2.5$ のとき最も速いと予想されたものの中から，a が単長数のときは逆平方根法と Paterson-Stockmeyer 法 ($r=5$) を，a が多倍長数のときは2次因子分解法 ($r=5$)，Todd の方法 ($r=9$) それに Knuth の方法 ($r=7$) を選んだ．図5より，a が単長数のときは逆平方根法が最も高速であり，a が多倍長数のときは2次因子分解法 ($r=5$) が最も高速であるという結論が得られた．また，a が多倍長数のときは逆平方



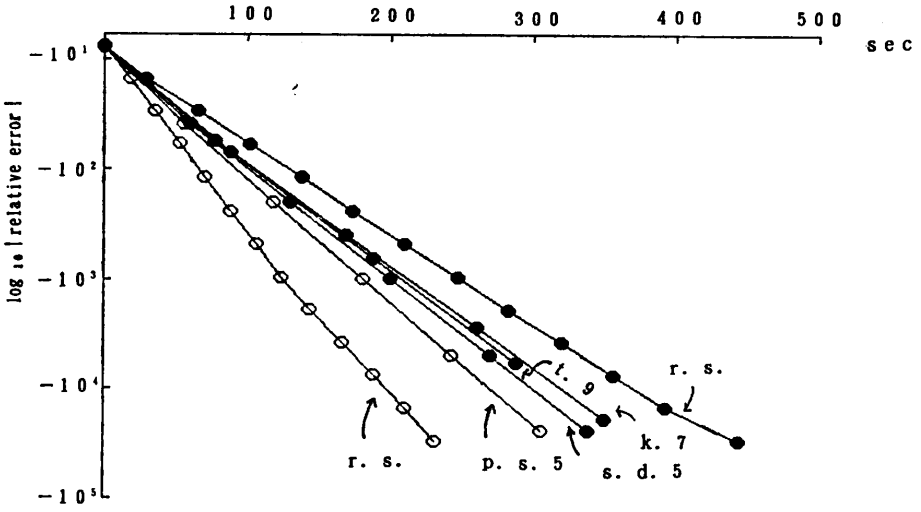
—○— Horner —×— P.S. (M)
—●— 2次因子分解 —■— P.S. (S)
—△— 逆平方根(S) —◇— Todd
—▲— 逆平方根(M) —□— Knuth
P.S.: Paterson-Stockmeyer
M: a=Multiple-Precision
S: a=Single Precision

図3 r 次収束法の時間計算量 ($d=4.0$)
Fig. 3 Time complexity of r-th order converging method ($d=4.0$).



—○— Horner —×— P.S. (M)
—●— 2次因子分解 —■— P.S. (S)
—△— 逆平方根(S) —◇— Todd
—▲— 逆平方根(M) —□— Knuth
P.S.: Paterson-Stockmeyer
M: a=Multiple-Precision
S: a=Single Precision

図4 r 次収束法の時間計算量 ($d=10.0$)
Fig. 4 Time complexity of r-th order converging method ($d=10.0$).



r.s.: 逆平方根法，s.d.5: 2次因子分解法 ($r=5$)，k.7.: Knuth 法 ($r=7$)，
t.9: Todd 法 ($r=9$)，p.s.5: Paterson-Stockmeyer 法 ($r=5$)
●: a=多倍長数 (a=Multiple-length)，○: a=単長数 (a=Single-length)

図5 r 次収束法の相対誤差（固定長演算）
Fig. 5 Relative errors of r-th order converging method (fixed-length arithmetic).

根法はかなり遅いことがわかった.

なお, このソフトウェアは FORTRAN 77 で作成したもので, 10 進浮動小数点表示を用い, 仮数部は整数型変数の1ワードに 10 進8桁を記憶し, 全体で 4096 ワード (すなわち $N=32,768$ である) としている. また, 指数と符号にそれぞれ整数型の1ワードを割り当てている. 除算のアルゴリズムは文献 8) で提案されたものを参考にした. 使用した計算機は東北大学大型計算機センターの ACOS 2020 であり, 最適化のオプションは OPT=0 (全く最適化しない) である.

以上の数値例は, 乗算の方法として $M(N)=O(N^2)$ のアルゴリズムを用いて得られたものであるが, ここで問題になっている「何次収束法が最も速いか」という問に対する答えは, (4.2)式を見ればわかるように, $c_r/\log r$ の値の大小で決まるのであって, $M(N)$ の形では決まらない. したがって, より高速な乗算法—例えば FFT を用いた方法とか Schönhage-Strassen[5] の方法—を用いたとしても, 乗算に対する他の演算の時間比が変わらない限り, この問に対する答えは同じになる.

5. 高次収束法とその計算効率 (2)

前章では, r 次収束法を「固定長演算」のもとで実行したときの時間計算量について考察した. これに対して本章では, 任意に基本演算の演算桁数を増減できるような演算—これをここでは「可変長演算」とよぶことにする—を用いて r 次収束法, および逆平方根法を計算したときの時間計算量を考察する.

Brent[6], 伊理[7] らは, ニュートン法を用いて根の近似を求めるとき, 現在得られている q 桁近似を更新し $2q$ 桁近似を得るためには, 増分の計算は q 桁で行えば十分であることを指摘している. したがって, 増分の計算を最初は短い桁数で行い, 反復1回ごとに2倍ずつ増やしていくという方法をとれば, 「固定長演算」で初めから長い桁数で行うよりもはるかに計算時間の節約になることがわかる. この考え方を $\sqrt{a}$ を求める r 次収束法にも適用する.

まず, r 次収束法(3.9)を次のように書き換える:

$$x_{k+1}=x_k+\Delta_r(x_k), \tag{5.1}$$

$$\Delta_r(x_k)=\sum_i\left[\binom{r}{i}-\binom{r}{2i+1}\right]a^i x_k^{r-2i}/B_r(x_k).$$

いま, x_k が $\sqrt{a}$ の b 進 q 桁の近似値であるとき, x_k の持つ誤差 e_k と増分 Δ_r は,

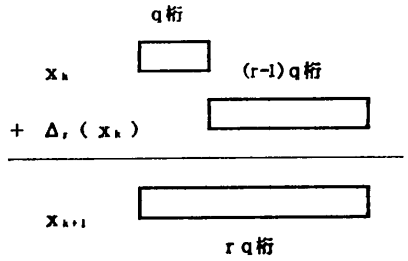


図 6 x_k と $\Delta_r(x_k)$ および x_{k+1} の関係
Fig. 6 Relation between x_k, $\Delta_r(x_k)$ and x_{k+1}.

$$|e_k/\sqrt{a}|\fallingdotseq|\Delta_r(x_k)/x_k|=O(b^{-q}) \tag{5.2}$$

を満たすから, 増分 Δ_r の先頭桁は x_k のそれよりおおよそ q 桁だけ下位に位置することがわかる. したがって, q 桁近似 x_k に増分 Δ_r を加えた値 x_{k+1} を rq 桁近似にするためには, 増分 Δ_r は上位 $(r-1)$ q 桁だけ計算すれば十分であることがわかる* (図6参照). しかし, ここでは計算中の丸め誤差の影響を考慮し, 少しゆとりを見て $r^{1-\delta}q$ 桁 $(0\leqq\delta<1)$ 程度で計算することにする. このような「手抜き」の計算を「可変長演算」を用いて行った場合, r 次収束法の時間計算量はどれくらいになるか, また何次収束法が最も高速であるかを考察する.

前章と同じように r 次収束法を p 回反復し t 桁近似から s 桁近似を得るための時間計算量を $T_r(t;s)$ とすると, この場合は

$$T_r(t;s)=c_r\sum_{i=1}^{p}M(r^{i-\delta}t) \tag{5.3}$$

である. ここで, 乗算の時間計算量を表す関数 $M(N)$ は N の単調増加関数で, $0<\alpha<1$ となる定数と任意の整数 $N>1$ に対して常に,

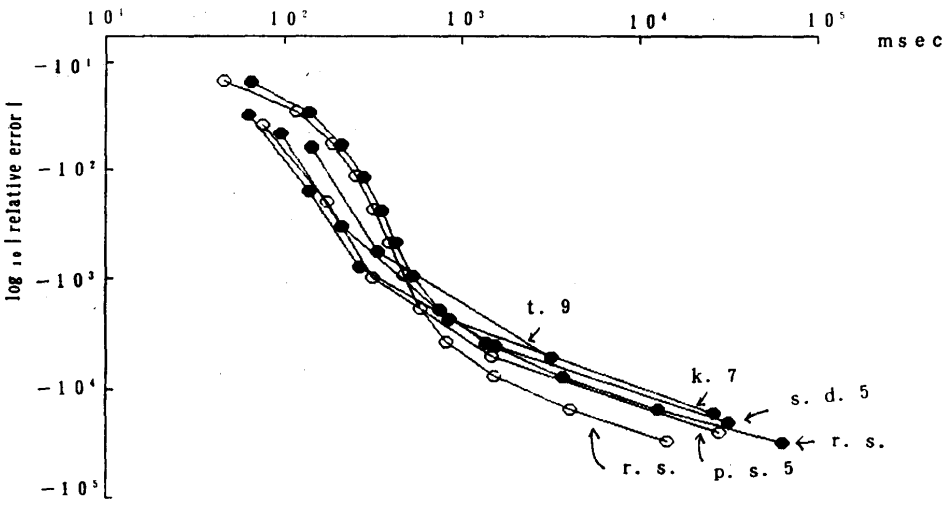
$$M(\alpha N)<\alpha M(N) \tag{5.4}$$

が成り立っているものとする. この仮定は, 例えば $M(N)=cN^2$ とか $M(N)=cN\log N$ とか $M(N)=c(N\log N)(\log\log N)$ であれば成立する. したがって, 通常の乗算法のほかに FFT による方法あるいは Schönhage-Strassen の乗算法でも有効である. このとき, 時間計算量 $T_r(t;s)$ は, (5.1)式の実質的な演算回数をやはり c_r と置くことによって,

$$T_r(t;s)=c_r\sum_{i=1}^{p}M(r^{i-\delta}t)$$

$$<c_r\sum_{i=0}^{p-1}M(sr^{-i})$$

* r 次収束法の計算を(3.9)式のように, $x_{k+1}=A_r(x_k)/B(x_k)$ として計算すると, A/B を rq 桁で計算しないと x_{k+1} は rq 桁近似にならなくなる. したがって, 「可変長演算」のときは, (5.1)式の形で計算する方が多少得である.



r.s.: 逆平方根法, s.d.5: 2次因子分解法 ($r=5$), k.7.: Knuth 法 ($r=7$),
t.9: Todd 法 ($r=9$), p.s.5: Paterson-Stockmeyer 法 ($r=5$)
●: a=多倍長数 (a=Multiple-length), ○: a=単長数 (a=Single-length)

図7 r次収束法の相対誤差（可変長演算）
Fig. 7 Relative errors of r-th order converging method (variable length arithmetic).

$$<c_r M(s)\sum_{i=0}^{p-1} r^{-i}$$
$$<c_r r(r-1)^{-1}M(s)$$
$$<2c_r M(s) \qquad (5.5)$$

となる．この式は，「可変長演算」を用いると s 桁の「固定長演算」の高々反復2回分で計算が完了することを意味している．

ここで r に無関係な定数 K が存在して，c_r が

$$c_r<K(r-1), \quad r>1 \qquad (5.6)$$

となっていると仮定すると，

$$T_r(t;s)<KrM(s) \qquad (5.7)$$

という結果が得られる．この式は r が小さいほど計算時間も小さくなることを意味している．また，当然のことながら，同じ r に対しては c_r が小さいほど時間計算量 $T_r(t;s)$ が小さくなるから，「可変長演算」の場合は逆平方根法(4.1)が最も高速であることが期待される．なお，(5.5)，(5.7)式が示す結果は，ここで問題にしている $\sqrt{a}$ を求める r 次収束法に限ったものではなく，r 次収束法全般についていえることでもある．

なお(5.1)式を見ればわかるとおり，増分 $\Delta_r(x)$ の分子は多項式 A とその次数および係数の形が同じになるため，(5.1)式を計算するとき前章で提案したどの方法を用いても，実質的な演算回数 c_r の値は同じになる．したがってここでは c_r の値についての考察は省略する．

[数値例3]

実際に「可変長演算」を用いて平方根を計算してみる．「固定長演算」の場合と同様に，$a=2$（単長数）と $a=0.3333\cdots$（多倍長数）の2通りを扱う．初期値として10進8桁（1ワード）を与え，演算桁数の取り方は，多項式の計算は反復1回ごとに $8r, 8r^2, 8r^3, \cdots$，と増やしていき，除算の方は $8(r-1), 8r(r-1), 8r^2(r-1), \cdots$，としていく．計算時間と相対誤差の関係を**図7**に示す．図7を見ればわかるとおり，a が単長数のときも多倍長数のときも，逆平方根が最も高速である．これは，(5.5)，(5.7)式から予想される結論と一致する．

6. あとがき

本論文では，実数の平方根に収束する高次収束法のアルゴリズム群を導出し，その多倍長演算における計算方法をいくつか提案した．ここで提案したアルゴリズムおよび計算法の時間計算量を詳細に検討した結果，次のような結論が得られた：

「固定長演算」のとき，

（1） 除算が比較的速くできるときは，5次収束法を「2次因子分解法」によって計算する方法，あるいは9次収束法を Todd の4次式によって計算する方法が最も高速である．

（2） 逆に除算が高速にできないとき，「逆平方根」が最も高速である．

(3) 単長数の平方根を計算するときは,「逆平方根法」および5次収束法を Paterson-Stockmeyer 法で計算する方法が高速である.

一方,「可変長演算」では,

(4) 単長数,多倍長数を問わず「逆平方根法」が最も高速である.

(5) どの方法を用いても,「固定長演算」の反復2回分以下の計算時間で計算が完了する.

なお,以上の結論は乗算法としてこれまで知られているどのようなものを用いたとしても,全く同じである.

謝辞 本研究の内容は,著者と仙台電波高専助教授海野啓明氏とのディスカッションによるところが大である.ここに海野助教授に深甚の謝意を表す.また,研究会の席上,多項式の計算方法について多くの有益な助言をいただいた国際基督教大学教授野崎昭弘氏,および,論文の不備を指摘して下さった査読者にも深甚の謝意を表す.なお,本研究の一部は,東北大学大型計算機センター研究開発公募「多倍長整数演算サブルーチンの作成とその整数論への応用に関する研究」の援助によるものである.

参考文献

1) Dutka, J.: The Square Root of 2 to 1,000,000 Decimals, *Math. Comp.*, Vol. 25, pp. 927-930 (1971).
2) 和田秀男:数の世界(科学ライブラリー), pp. 169-175, 岩波書店, 東京 (1981).
3) Traub, J. F.: *Iterative Methods for the Solution of Equations*, p. 19, Chelsa Publ. Comp., New York (1982).
4) 野崎昭弘:計算機数学(共立数学講座 11), pp. 54-95, 共立出版, 東京 (1984).
5) Knuth, D. E.: *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, pp. 258-279, Addison-Wesley, Reading, Massachusetts (1969).
6) Brent, R. P.: Fast Multiple-Precision Evaluation of Elementary Functions, *J. ACM*, Vol. 23, No. 2, pp. 242-251 (1976).
7) 伊理正夫:ニュートン法の実際,数理科学, Vol. 218, pp. 10-16 (1981年8月号).
8) 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,電総研彙報, Vol. 42, pp. 66-71 (1978).
9) 小沢一文,海野啓明:連分数を用いた平方根の高速発生法とその計算効率,情報処理学会アルゴリズム研究会資料, SIG-AL 5-3, pp. 17-24 (1989).
10) 金田康正:円周率 800 万桁及び 1/sqrt(2), sqrt(2), 1600 万桁の検証・統計結果,数理解析講究録, Vol. 498, pp. 66-88 (1983).

(平成元年8月31日受付)
(平成2年4月17日採録)

小沢 一文(正会員)

1947年生. 1974年早稲田大学大学院理工学研究科修士課程修了. 工学博士. 現在,仙台電波工業高等専門学校助教授として教育に従事するかたわら,常微分方程式の数値解法および多倍長演算のアルゴリズムの開発等の研究に従事. 著書には,「数値計算法」(共立出版),「PASCAL」(共著,日刊工業新聞社)がある. 本学会のほかに,応用統計学会,日本応用数理学会, SIAM 各会員.