C 2 CORDIC に基く複素数計算

一 松　　信 （京大数理解析研）

0. 要　　約

座標変換に基いて初等函数を計算する着想は，少なくとも1956年頃からあるが，近年J. S. Waltherは，統一的手法にまとめ上げ，超小型計算機や電卓に実用化した。

初等函数が，四則演算と同一水準で計算できるとなると，計算法にも大きな変革が生ずる。その一例として，複素数計算（乗除，平方根，初等函数）に有効に応用ができる。この報告は，その実用化にあたって筆者の直面した細かい技術的注意が主であるが，今後の科学技術専用計算機の設計に多少のヒントも含まれるように思った。

1. 原　　理

Waltherのもとの形は少しわかりにくいので，$m = +1$ の場合をまずのべる（また後の都合上多少変更してある）。

平面上の点 $P_k = (x_k , y_k)$ につぎの変換をほどこす，ここに δ_k はある定数である。

$$\begin{aligned} x_{k+1} &= x_k - \delta_k y_k \\ y_{k+1} &= y_k + \delta_k x_k \end{aligned} \qquad (1)$$

極座標 (R_k , A_k) を使うと，（1）はつぎの変換になる。

$$\begin{aligned} R_{k+1} &= R_k \times K_k \ , \quad K_k = (1 + \delta_k{}^2)^{1/2} \\ A_{k+1} &= A_k + \alpha_k \ , \quad \alpha_k = \arctan \delta_k \end{aligned} \qquad (2)$$

したがって，$\delta_0 , \delta_1 , \cdots\cdots$ をとり，(x_0 , y_0) からはじめて毎回変換 $(x_k , y_k) \to (x_{k+1} , y_{k+1})$（$\delta_k$ による）を行ない，(x_n , y_n) に到達したとすると，つぎの結果をうる。

$$R_n = R_0 \times K , \quad K = \prod_{k=0}^{n-1} (1 + \delta_k{}^2)^{1/2} \qquad (3)$$

$$A_n = A_0 + \alpha , \quad \alpha = \sum_{k=0}^{n-1} \arctan \delta_k$$

$$\begin{aligned} x_n &= K (x_0 \cos \alpha - y_0 \sin \alpha) \\ y_n &= K (y_0 \cos \alpha + x_0 \sin \alpha) \end{aligned} \qquad (4)$$

便宜上（1）と平行して，つぎの変換を加える：

$$x_{k+1} = z_k - \alpha_k \ ; \ (z_n = z_0 - \alpha) \qquad (5)$$

適当な δ_k の列に対し，つぎのいずれかをあらかじめ選ぶ（後述参照）。

I. A_n あるいは y_n を0に近づける。

II. z_n を0に近づける。

IIが達せられ，$z_n = 0$ になったとすると，$\alpha = z_0$ であるから，最後にえられた値は（4）で $\alpha = z_0$ と

したものである。したがってたとえば，あらかじめ $x_0 = 1/K$, $y_0 = 0$ としておけば，$x_n = \cos z$, $y_n = \sin z_0$ が同時に求められる。

またIが達せられ，$y_n = 0$ となったとすると，

$$x_n = K\sqrt{x_0^2 + y_0^2}, \quad z_n = z_0 + \arctan(y_0/x_0) \qquad (6)$$

である。これによって逆正接（およびある種の平方根）が求められ，また極座標⟷直交座標の変換が容易にできる。

具体的には，$\delta_k = \pm 2^{-k}$ （ $k = 0, 1, 2, \cdots\cdots$ ）とし，α_k の値はあらかじめ計算しておく（便宜上末尾に付録としてつけておいた）。こうすると（2進法の計算機では），（1）は加減算とシフトだけで（乗算なしで）可能であり，全体として，加減算，シフト，判断，定数読出しの機能のみで計算が可能である。これが超小型計算機などに特に適する所以である。

2. 具体的算法

便宜上 $\varepsilon_k = 2^{-k}$, $\beta_k = \arctan 2^{-k}$（あらかじめ計算しておく）とおく。また簡単のため $x_0 \geqq 0$ としておく。

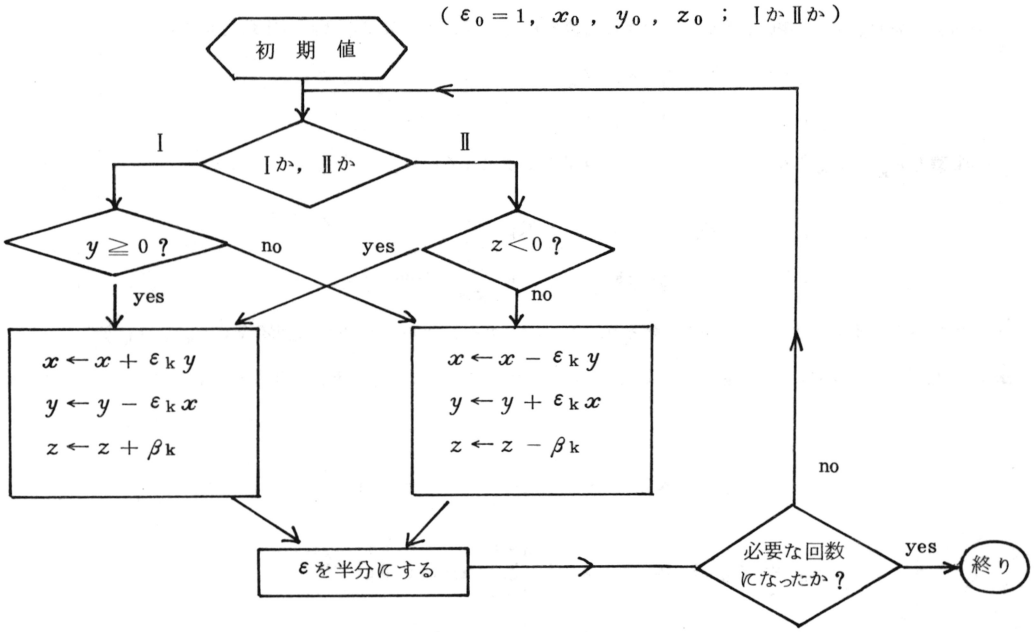

図　1

その流れ図は図1のようになる。IかⅡかを切換えずに，別々のサブルーチンにしてもよいが，算法に重複が多いので，一つにまとめた。

表 1.　初期値のおき方：

目　　標	x_0	y_0	z_0	I, II
$\cos\alpha$, $\sin\alpha$	$1/K$	0	α	II
$r\cos\alpha$, $r\sin\alpha$	r/K	0	α	II
$a\cos\alpha - b\sin\alpha$ $a\sin\alpha + b\cos\alpha$	a/K	b/K	α	II
$\arctan(b/a)$ 主値	a	b	0	I
$\arctan(b/a)$ $]-\pi, +\pi]$ の値 $a \geqq 0$	a	b	0	I
$\arctan(b/a)$ $]-\pi, +\pi]$ の値 $a < 0,\ b \geqq 0$	$-a$	$-b$	$+\pi$	I
$\arctan(b/a)$ $]-\pi, +\pi]$ の値 $a < 0,\ b < 0$	$-a$	$-b$	$-\pi$	I

$$K = \prod_{k=0}^{n-1} \left(1+2^{-2k}\right)^{1/2}$$　　実用上は無限乗積にした値でよい：

$1/K = 0.607252935 0$

$1/K^2 = 0.368756127 1$

Walther の原論文では，初期値が小さいとき，その大きさに応じて $\varepsilon_0 = 1$ からでなく，$\varepsilon_0 = 2^{-j}$ から始める工夫がある。たしかにそのほうが精度が高い。しかし絶対誤差だけを問題にするのなら，一律に $\varepsilon_0 = 1$ から始めても，実用上さしつかえない。

3. 吟　　味（収束性）

この算法が収束するためには，$\delta_k = \pm\varepsilon_k$ を全部同一符号にとって0にもってこれる必要がある。$\varepsilon_k = 2^{-k}$ とすると

$$\beta = \sum_{k=0}^{\infty} \arctan 2^{-k} \fallingdotseq 1.72 > \pi/2 \qquad (7)$$

であるから，右半平面が含まれる。

arctan t は $t \geqq 0$ のとき上に凸であるから，不等式

$$\beta_{k-1} - \sum_{j=k}^{n-1} \beta_j < \beta_{n-1} \qquad (8)$$

が成立する。このことから，$|A_0| < \beta$ ならば，上記の算法で $|A_k| < \beta_{k-1}$ であることが証明される。ゆえにIについては $x_0 \geqq 0$ なら必ず収束し，IIについては，$|z_0| < \beta$（したがって $|z_0| \leqq \pi/2$ ならなおさら）収束する。

最後の y または z が正確に0にならず，ε だけ残ったときの誤差も容易に解析でき，いずれも ε のオーダーであることがたしかめられる（ただし計算中の丸め誤差は無視する）丸め誤差についても，それが成長して不安定性をひきお

こすことがないことは容易にたしかめられる。

したがって直交座標 → 極座標については，右半平面に限定すればよく，逆は偏角を 〔 $-\pi/2$ ， $+\pi/2$ 〕に限定すればよい。いずれも実用上ではほとんど問題にならない制限である。

4. 複素数計算への応用

複素数の乗除算などへは，つぎのように適用する。

乗 法 (a + ib)(p + iq)

1. p + iq をIで極座標に直す。このとき偏角はつねに 〕$-\pi/2$ ，$\pi/2$ 〔にとり，p < 0 のときは，**絶対値を負**とするほうが便利である。すなわち初期値：(p = 0 のときは sgn p = 1 とする)

$$x_0 = |p|, \quad y_0 = (\text{sgn } p)\cdot q \quad z_0 = 0$$

からCORDIC-I でx，y(=0)， を求め，p < 0 ならば，えられたxの $-x$ をxとする。

2. 因数Kを補正するため， $x/K^2 = r$ とする。

3. $x_0 = ra$ ， $y_0 = rb$ ， $z_0 = z$ から CORDIC-Ⅱ でx，y，z (=0) を求める。x，y が積の実部，虚部を与える。

除 法 (a + ib)/(p + iq)

1. は乗法と同じ。

2. は不要。

3. $x_0 = a/r$ ， $y_0 = b/r$ ， $z_0 = -z$ から CORDIC-Ⅱ でx，y，z (=0) を求める。x ，yが商の実部，虚部を与える。

平方根 $\sqrt{p + iq}$

このときは，CORDIC-I で正規の偏角を 〕$-\pi$ ，$+\pi$ 〔の間に求め，えられたxの平方根に$K^{-3/2}$を乗じ，それを x_0 ， $y_0 = 0$ ， $z_0 = z/2$ としてCORDIC-Ⅱ を適用する。x，y が，実部が正の平方根を与える。他は $-x$ ，$-y$ である。

指数函数 exp (p + iq)

1. $r = e^p/K$ を求める。

2. qを 〔 $-\pi/2$ ，$+\pi/2$ 〕に還元する，具体的には，つぎのようにする：

$$q \times (2/\pi) = n + \xi, \quad 0 \leqq \xi < 1, \quad n\text{整数}$$

n + 1 の(2進で)最下位2ビットを n_1 n_0 としたとき，

もし $n_1 = 1$ならば，$-r$をrとする。

もし $n_0 = 0$ならば，$1 - \xi$ を ξとする。

$q_0 = \xi \times (\pi/2)$ とする。

3. $x_0 = r$ ， $y_0 = 0$ ， $z_0 = q_0$ に，CORDIC-I を適用する。えられたx，y，z (=0) が指数函数の実部，虚部を与える。

対数函数 log (p + iq)

1. (p ， q) を正規の極座標(偏角 〕$-\pi$ ，$+\pi$ 〔)に直す。そのzがlogの虚部(主値)である。

2. そのxの対数を求め，log Kを減ずる。それがlogの実部である。(桁落ちを避けるにはx/Kの log を求めてもよい)。

cordicによる変換(1回)が乗除算と同じ位の速さでできるとして、

なお平方根，指数函数，対数函数は，下記のように，双曲型のCORDIC で計算することもできる。

乗法は普通の算法（乗法4回）より少し手間がかかるが，同じ乗数の反復乗算（たとえば多項式の計算）や，乗数が始めから極座標形で与えられているとき（たとえばＦＦＴ）には，１（極座標への変換）の算法は１回のみ，または不要であるから，不利とはいえない。除法では分母分子のKが約されて，Kの補正が不要であるため，普通の算法より有利である。平方根，指数函数，対数函数も普通の算法より有利である。

μ-step の数のCC技

5. 双曲型のCORDIC

Walther の論文で，m ＝ 0 に相当するのは，（1）の下の式のみを反復するもので，$\delta_k = \pm 2^{-k}$ としたときは，2進法の non - restorting 法による除法またはその逆の乗法となる。乗除算回路のない超小型計算機では，これによって乗除算が実行される。

しかしそれよりも興味があるのは，m ＝ －1 に相当する双曲型の場合である。変換は1のものを修正してつぎのようになる。ただし以下 $x \geqq |y|$ とする。（5）は1と同じ。

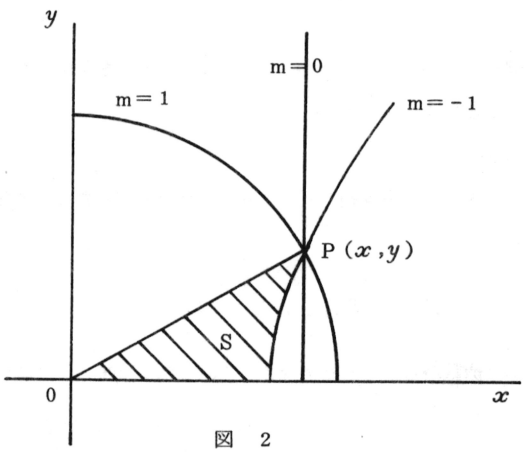

図 2

$$\begin{cases} x_{k+1} = x_k + \delta_k y_k \\ y_{k+1} = y_k + \delta_k x_k \end{cases} \tag{1'}$$

双曲線座標：$x = R \cosh A$，$y = R \sinh A$，

$R = (x^2 - y^2)^{1/2}$，　$A = \operatorname{arctanh}(y/x)$

$$\begin{cases} R_{k+1} = R_k \times K_k, \quad K_k = (1 - \delta_k{}^2)^{1/2} \\ A_{k+1} = A_k + \alpha_k, \quad \alpha_k = \operatorname{arctanh} \delta_k \end{cases} \tag{2'}$$

$$\begin{cases} R_n = R_0 \times \widetilde{K}, \quad \widetilde{K} = \prod_{k=0}^{n-1} (1 - \delta_k{}^2)^{1/2} \\ A_n = A_0 + \widetilde{\alpha}, \quad \widetilde{\alpha} = \sum_{k=0}^{n-1} \operatorname{arctanh} \delta_k \end{cases} \tag{3'}$$

$$x_n = \tilde{K}\,(x_0 \cosh \tilde{\alpha} + y_0 \sinh \tilde{\alpha})$$
$$y_n = \tilde{K}\,(y_0 \cosh \tilde{\alpha} + y_0 \sinh \tilde{\alpha}) \qquad (4')$$

Aは点P（x, y ）を通るR ＝ 一定の線，OP，x軸で囲まれる部分の面積をSとすると，$2S/R^2$ に等しい。

表2　初期値のおき方（双曲型のとき）

目　標　値	答	x_0	y_0	z_0	I, II
$\cosh \alpha$, $\sinh \alpha$	x , y	$1/\tilde{K}$	0	α	II
$\exp \alpha$	$x=y$	$1/\tilde{K}$	$1/\tilde{K}$	α	II
$\operatorname{arctanh}(b/a)$	z	a	b	0	I
$\log(b/a)$	$2z$	a+b	a−b	0	I
$\tilde{K}(a^2 - b^2)^{1/2}$	x	a	b	0	I
$2\tilde{K}\sqrt{b}\sqrt{a}$	x	a+b	a−b	0	I

$\tilde{K}$ ＝ 0.8281593608　　（ k ＝ 4, 13, ……を反復した無限乗積 ）

このときには，$|\delta_k| < 1$ である必要があるから

$$\varepsilon_k = 2^{-k} \quad (k = 1, 2, \cdots\cdots), \qquad \delta_k = \pm\varepsilon_k$$

ととる。しかし $\beta_k = \operatorname{arctanh} \varepsilon_k$ とすると，不等式（8）は成立しない（逆向きの＞になる）。そのため収束しない隙間が生ずる。幸いこれを修正して

$$\beta_{k-1} - \sum_{j=k}^{n-1} \beta_j - \beta_\ell < \beta_{n-1}, \quad \ell \leqq 3k - 2 \qquad (8')$$

は成立する。（これはarctanhのテイラー級数からすぐに証明できる）。したがって

$$k_j = 3k_{j-1} + 1, \quad k_0 = 1 \qquad (9)$$

で与えられる列　k_1 , k_2 , ……　すなわち

$$k = 4, 13, 40, 121, \cdots\cdots$$

において，同じ ε_k , β_k を使ってもう一度反復すれば，

$$|z_0| < \beta = \sum \beta_k \doteqdot 1.12$$

において収束することがたしかめられる。10進10桁程度なら4と13（偶然だが縁起の悪い数？）で，10進36桁までなら，4，13，40 で一度足踏みをすればよい。この種の収束のための修正技巧は，他にもいろいろ考えられるが，これがもっとも簡単なようである。

流れ図は図1のXの式の ε_k の符号をかえ，$\varepsilon_1 = 1/2$ から始め，$\beta_k = \operatorname{arctanh} \varepsilon_k$ とし，さらに k ＝ 4, 13,（40）で足踏みするように修正すればよい。

6. 双曲型CORDICの応用

以上がWalther の方法であるが，指数函数，対数函数，平方根の個々のサブルーチンには，少し修正したほうが

よい。

指数函数 cosh , sinh が不要で exp だけがほしいときには, $x_0 = y_0 = 1/\tilde{K}$ ではじめればよいが, こうするとつねに $x = y$ で同一の計算を2度やっている。このむだを省くには, x と z のみとし, つぎの計算を(k = 4, 13, 40で足踏みして) 反復するとよい。

```
if z < 0 then begin z := z + β ; x := x − ε x end
         else begin z := z − β ; x := x + ε x end ;
```

収束域は, $|z_0| < \beta$ (≒ 1.12) である。したがって, あらかじめつぎのようなスケーリングがいる。

e^t に対し, $t/\log_e 2 = n + q$, n整数, $-1 < q \leqq 0$, nが2進の指数部,

$$x_0 = 1/\tilde{K}, \quad z_0 = q \log_e 2$$

複数変数の指数函数用には, $x_0 = 1/K\tilde{K} = 0.7332561385$ から始めるとよい。

対数函数 log は exp の上記の逆をやるとよい。ただし除法を避けるため, 判断用の変数 w をおき, つぎの計算を反復する:

```
if x ≧ w then begin z := z + β ; w := w + ε w end
         else begin z := z − β ; w := w − ε w end ;
```

初期値は x_0 = 引数, $z_0 = 0$, $w_0 = 1/\tilde{K}$ であるが, 収束域が $e^{-\beta} < x < e^{\beta}$ なので, 次のスケーリングを行なう: 引数の仮数部を a, (1/2 ≦ a < 1), 指数部を n とするとき

$n \leqq 0$ ならば $x_0 = a$, $z_0 = n \times \log_e 2$

$n \geqq 1$ ならば $x_0 = 2a$, $z_0 = (n - 1) \times \log_e 2$

(こうしたのは, 引数≒1のとき桁落ちを防ぐためである)

これは $\log \sqrt{3}$, $\log \sqrt{5/3}$, $\log \sqrt{9/7}$, …… でちじめた一種の変形二分法とも解釈できる。複素変数の対数函数用には, z_0 からさらに log K を引いて補正をしておくとよい。

平方根 もし副産物として arctanh が出るのが「産業廃棄物」(公害の元兇?) だと思うのならば, z と定数 β_k を廃止し x と y とだけにつぎの反復を行なえばよい。

```
if y ≧ 0 then begin x := x − ε y ; y := y − ε x end
         else begin x := x + ε y ; y := y + ε x end ;
```

初期値としては,

$$x = a + b, \quad y = a - b, \quad b = 1/4\tilde{K}^2 = 0.3645122922$$

をとる。 Walthen の原論文は b = 1/4 としているが, 上のようにとると, $2\tilde{K}\sqrt{b} = 1$ となり, 因数の補正なしに直接 $\sqrt{a}$ がでる。このときの収束域は $e^{-\beta} < a/b < e^{\beta}$ (ほぼ〔 0.04, 3 〕) で, $1/4 \leqq a \leqq 1$ または $1/16 \leqq a \leqq 1$ を十分に含む。これ以外の値に対しては, もちろんスケーリングをし, $\sqrt{0} = 0$ とおく。

複素数の平方根用には $2\tilde{K}K^{3/2}\sqrt{b} = 1$ であるようにすると, 補正が不要になるが, この b はほぼ 0.0816…… で, 収束域が〔 0.009, 0.75 〕と小さすぎる (じっさいこれに気づかずに大失敗をやらかした)。したがってこれを4倍し,

$$b = 1/\tilde{K}^2K^3 = 0.32649838486\cdots\cdots$$

とし (収束域はほぼ〔 0.04, 1.85 〕), $1/4 \leqq a \leqq 1$ を含ませてあとで 1/2 にする (1桁右にシフト) と

いうスケーリングをしたほうがよい。

なお実際には，これらの定数は理論値よりも，じつさいにいろいろな引数に対して求めた真値との差をなるべく小さくするように，実験的な現場合せの手法で末尾の桁を若干修正した値としたほうがよいである。そのような意味で，これまで筆者がやってきたFORTRAN によるシミュレーションも，十分に実用化試験の意味があることと信ずる。

7. むすび

CORDIC の欠点は下記のこととされる。

1° 定数記憶を数多く必要とする。

2° 同時並行処理ができないと（少なくとも2つのレジスタの同時運転がきかないと）利点が十分に発揮でない。

しかし arctan も arctanh も5乗以上が無視できれば $\varepsilon \mp \varepsilon^3/3$ でよいし，3乗以上が無視できれば ε でよい。したがってNビットの計算には合計2N／3（または2N／5＋若干）個でよく，これは各種の初等函数の近似式の係数全体とくらべて，それほど多くはない。2° は今後の科学技術計算用の計算機の金物設計者の責任としよう。

他方長所として下記の点が考えられる。

1° 加減算とシフトと判断のみで実行できる。　　したがって金物で組んでしまうことが可能である。

2° 定数さえ十分の桁数があれば，ビット数（精度）に関係なく同一算法でよい。したがってとくに超倍長精度計算用に有利である。

3° exp と log；sin，cos と arctan などが，同一プログラムの逆算として実行できる。また a cos t − b sin t などが一度に計算できる。　Walther の作つた回路では，乗除算より arctan のほうが早いという。初等函数は，乗除算と同一水準で計算できるわけである。

20 年前，函数近似公式が函数表にかわったが，いまや少なくとも初等函数に関する限り，函数近似公式は不要になった感がある。計算法の進展とともに，「ヤラレタ」というのが筆者の正直な感想である。

J. S. Walther　　SJCC　1971, P.379～385
S　　IEEE　1965

CO ordinate Rotation DIgital Computer

短波　Digital Resolver　計測と制御（～10年位前）

表3　$\tan^{-1} xi$，$xi=2^{-i}$ の　10進表示

1	0.46364	76090	00806	11621	42562	31461	21440	20285	37054	28612
2	0.24497	86631	26864	15417	20824	81211	27581	09141	44098	38119
3	0.12435	49945	46761	43503	13548	49163	87102	55731	70191	76980
4	0.06241	88099	95957	34847	39791	12985	50511	36062	73897	79749
5	0.03123	98334	30268	27625	37117	44892	49097	70324	95663	72540
6	0.01562	37286	20476	83090	28015	21256	57031	89111	14139	80090
7	0.00781	23410	60101	11129	64633	91842	19928	16212	22811	72501
8	0.00390	62301	31966	97182	76296	65311	42438	71403	57490	11520
9	0.00195	31225	16478	81368	51214	82625	07671	39316	10746	77723
10	0.00097	65621	89559	31943	04034	30199	71729	08516	34197	01581
11	0.00048	82812	11194	89827	54692	39625	64484	86661	92361	13313
12	0.00024	41406	20149	36176	40167	22943	25965	99862	12417	79097
13	0.00012	20703	11893	67020	42390	58646	11795	63009	30829	40901
14	0.00006	10351	56174	20877	50216	62569	17382	91537	85143	53683
15	0.00003	05175	78115	52609	68618	25953	43853	60197	50949	67511
16	0.00001	52587	89061	31576	21072	31935	81269	78851	37429	23814
17	0.00000	76293	94531	10197	02633	88482	34010	50905	86350	74391
18	0.00000	38146	97265	60649	62829	23075	61637	29937	22805	25730
19	0.00000	19073	48632	81018	70353	65369	30591	72441	68714	34216
20	0.00000	09536	74316	40596	08794	20670	68992	31123	90019	63412

表4　$\tan^{-1} xi$，$xi=2^{-i}$ の　8進表示

1	0.35530	63405	30335	51732	12677	44213	66274	16506	40333	41202
2	0.17533	35374	45440	15654	25333	43636	75373	64205	76265	23772
3	0.07752	67246	52605	73334	31310	45714	23717	50421	67570	54712
4	0.03775	25566	71317	67516	15545	44744	55601	61340	65211	43160
5	0.01777	52533	56513	54222	50235	71656	73335	52143	04116	00332
6	0.00777	75252	67355	62465	33574	74305	31000	53735	40220	20357
7	0.00377	77525	25567	35227	13200	30432	43756	76301	27507	32053
8	0.00177	77752	52533	56733	45135	42253	73326	66517	37413	17406
9	0.00077	77775	25252	67356	72456	24653	36526	01045	07164	36220
10	0.00037	77777	52525	25567	35667	12271	32003	17674	72457	46466
11	0.00017	77777	75252	52533	56735	65134	51354	22542	22501	17160
12	0.00007	77777	77525	25252	67356	73556	24562	46533	65335	74736
13	0.00003	77777	77752	52525	25567	35673	52271	22713	20032	00304
14	0.00001	77777	77775	25252	52533	56735	67334	51345	13542	25422
15	0.00000	77777	77777	52525	25252	67356	73567	24562	45624	65336
16	0.00000	37777	77777	75252	52525	25567	35673	56671	22712	27132
17	0.00000	17777	77777	77525	25252	52533	56735	67356	51345	13451
18	0.00000	07777	77777	77752	52525	25252	67356	73567	35562	45624
19	0.00000	03777	77777	77775	25252	52525	25567	35673	56735	22712
20	0.00000	01777	77777	77777	52525	25252	52533	56735	67356	73345

表5　$\tanh^{-i}xi$，$xi=2^{-i}$ の　10進表示

1	0.54930	61443	34054	84569	76226	18461	26285	23237	45278	91137
2	0.25541	28118	82995	34160	27570	48151	83096	74390	55398	22288
3	0.12565	72141	40453	03884	25688	65200	93583	98289	48193	03181
4	0.06258	15714	77003	00712	67650	23862	20659	57075	55550	24723
5	0.03126	01784	90666	99476	40122	45172	64889	70096	29240	63978
6	0.01562	62717	52052	21137	92017	78751	63755	26994	01726	90068
7	0.00781	26589	51540	42091	03234	71276	04017	26663	58809	38910
8	0.00390	62698	68396	82605	31275	63360	70778	59267	14952	15496
9	0.00195	31274	83532	54999	86507	70886	85417	53190	06680	15695
10	0.00097	65628	10441	03584	09644	50029	88532	62542	38417	84778
11	0.00048	82812	88805	11282	67610	06626	27116	04168	89227	70820
12	0.00024	41406	29850	63858	29279	72252	10244	36712	63816	17295
13	0.00012	20703	13106	32980	66029	63078	73708	93765	95048	86859
14	0.00006	10351	56325	79122	53171	50609	72789	09844	54395	13193
15	0.00003	05175	78134	47390	31487	61958	40214	27344	92506	32814
16	0.00001	52587	89063	68423	78930	98936	43232	33259	32529	29012
17	0.00000	76293	94531	39802	97366	21857	41755	18222	59585	16815
18	0.00000	38146	97265	64350	37170	77247	50105	37848	03630	23937
19	0.00000	19073	48632	81481	29646	34640	79150	23426	60236	76710
20	0.00000	09536	74316	40653	91205	79329	62562	12496	98385	08804

表6　$\tan^{-}/xi$，$xi=2^{-i}$ の　8進表示

1	0.43117	52365	26403	02513	55220	31432	52405	16676	26557	41210
2	0.20261	27372	40212	12151	62355	36057	40402	07332	47642	63405
3	0.10025	42216	23653	57355	14506	36407	55344	26614	44513	77126
4	0.04002	53042	55156	55315	62611	42251	26170	64122	25363	73164
5	0.02000	25261	04432	73422	26447	35177	56133	31567	23707	65602
6	0.01000	02525	42105	32162	30701	51645	30526	35514	50552	05437
7	0.00400	00252	53042	11065	51564	17212	37421	35502	27362	55704
8	0.00200	00025	25261	04212	64327	34173	71150	02007	45655	37106
9	0.00100	00002	52525	42104	22153	21623	06236	46601	06075	14375
10	0.00040	00000	25252	53042	10425	50655	15641	56552	72023	71345
11	0.00020	00000	02525	25261	04210	44326	43273	41734	22263	30507
12	0.00010	00000	00252	52525	42104	21053	21532	16230	62307	01511
13	0.00004	00000	00025	25252	53042	10421	10655	06551	56415	64172
14	0.00002	00000	00002	52525	25261	04210	42126	43264	32734	17341
15	0.00001	00000	00000	25252	52525	42104	21042	21532	15321	62306
16	0.00000	40000	00000	02525	25252	53042	10421	04255	06550	65515
17	0.00000	20000	00000	00252	52525	25261	04210	42104	43264	32643
18	0.00000	10000	00000	00025	25252	52525	42104	21042	10532	15321
19	0.00000	04000	00000	00002	52525	25252	53042	10421	04211	06550
20	0.00000	02000	00000	00000	25252	52525	25261	04210	42104	21264

本 PDF ファイルは 1965 年発行の「第 6 回プログラミング―シンポジウム報告集」をスキャンし、項目ごとに整理して、情報処理学会電子図書館「情報学広場」に掲載するものです。

この出版物は情報処理学会への著作権譲渡がなされていませんが、情報処理学会公式 Web サイトの https://www.ipsj.or.jp/topics/Past_reports.html に下記「過去のプログラミング・シンポジウム報告集の利用許諾について」を掲載して、権利者の捜索をおこないました。そのうえで同意をいただいたもの、お申し出のなかったものを掲載しています。

過去のプログラミング・シンポジウム報告集の利用許諾について

情報処理学会発行の出版物著作権は平成 12 年から情報処理学会著作権規程に従い、学会に帰属することになっています。

プログラミング・シンポジウムの報告集は、情報処理学会と設立の事情が異なるため、この改訂がシンポジウム内部で徹底しておらず、情報処理学会の他の出版物が情報学広場 (=情報処理学会電子図書館) で公開されているにも拘らず、古い報告集には公開されていないものが少からずありました。

プログラミング・シンポジウムは昭和 59 年に情報処理学会の一部門になりましたが、それ以前の報告集も含め、この度学会の他の出版物と同様の扱いにしたいと考えます。過去のすべての報告集の論文について、著作権者（論文を執筆された故人の相続人）を探し出して利用許諾に関する同意を頂くことは困難ですので、一定期間の権利者捜索の努力をしたうえで、著作権者が見つからない場合も論文を情報学広場に掲載させていただきたいと思います。その後、著作権者が発見され、情報学広場への掲載の継続に同意が得られなかった場合には、当該論文については、掲載を停止致します。

この措置にご意見のある方は、プログラミング・シンポジウムの辻尚史運営委員長 (tsuji@math.s.chiba-u.ac.jp) までお申し出ください。

加えて、著作権者について情報をお持ちの方は事務局まで情報をお寄せくださいますようお願い申し上げます。

期間：2020 年 12 月 18 日 〜 2021 年 3 月 19 日
掲載日：2020 年 12 月 18 日

プログラミング・シンポジウム委員会

情報処理学会著作権規程
https://www.ipsj.or.jp/copyright/ronbun/copyright.html