STICHTING
MATHEMATISCH CENTRUM
2e BOERHAAVESTRAAT 49
AMSTERDAM

AFDELING TOEGEPASTE WISKUNDE

TW 105

Complex Arithmetic in ALGOL 60

by

R.P. van de Riet

March 1967

The Mathematical Centre at Amsterdam, founded the 11th of February, 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications, and is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.) and the Central Organization for Applied Scientific Research in the Netherlands (T.N.O.), by the Municipality of Amsterdam and by several industries.

Table of contents

Summary

This report contains a set of ALGOL 60 procedures by means of which ALGOL 60 programs may be written for computations with complex numbers. Besides procedures for the elementary operations, procedures are given for the elementary functions: exp, sin, cos, tan, ln, sqrt, arctan, arcsin en arccos.
The result of the last five multivalued functions is not necessarily the ordinary principal value; as the user may specify his own principal value.

Introduction

In designing a system of ALGOL 60 procedures for formula manipulation [1], it turned out to be desirable to have a set of procedures for complex arithmetic in ALGOL 60; since formula manipulation systems are used and will be used in the future for generating programs which perform ordinary as well as complex arithmetic.
In a forth-coming report such a system will be described. Of course, a system of procedures for complex arithmetic may very well be of use without the connection with formula manipulation.

The procedures in this report have been tested on the Electrologica X8 computer of the Mathematical Centre, using the X8-ALGOL 60 compiler written by F.E.J. Kruseman Aretz and B.J. Mailloux.

The author is grateful to Dr. T.J. Dekker for his critical review of this report.

2.1. Complex arithmetic in ALGOL 60

ALGOL 60 does not admit expressions of complex arithmetic type. In overcoming this difficulty one may split the expressions into real and imaginary parts, which leads to efficient use of computer time (but not of programmer's time), or the expressions may be written in polish prefix form which will be done in this report.
In the latter case the symbols +, ×, ... are replaced with operators S, P, ...
Thus:

a + b + c is written as S(a,S(b,c))
and a × b + c is written as S(P(a,b),c).

The form of the ALGOL 60 procedures S and P will be investigated. As S and P are used recursively, it is necessary to use a stack in order to save partially formed results. The stack is defined as array R,I[0 : bound], where bound is some large enough integer, say, 1000.
The stack pointer is called p.

If S and P are chosen procedures of type integer, their values on execution may refer to indexes of R and I. Note that the parameters of S and P should also be of type integer.
The rough structure is now nearly determined. There remains to investigate the effect of S and P on the stack pointer p. There are three possibilities but only the first one will turn out to be useful:
1. p is unaltered
2. p is decreased
3. p is increased.
Decreasing p is only possible if the evaluation of the parameters has as side-effect an increase in p.
It is then not possible to write S(a,b), where a and b are variables, but instead one should write

S(TAKE(a), TAKE(b)).

Increasing p means that all intermediate results of S(a,S(b,S(c,S(d,e)))) remain stored, which is a waste of storage space.

To be sure that partially formed results as in S(P(a,b),P(c,d)), are not effected by future calculations, p should be temporarily increased before the evaluation of the second parameter of S and P. This means that the parameters (at least the second) should be called by name.

We have then the following declaration:

```
integer p; array R,I[0 : bound];
integer procedure S(u,v); integer u,v;
begin integer a,b; a:= u; p:= p+1; b:= v; S:= p:= p-1;
    R[p]:= R[a] + R[b]; I[p]:= I[a] + I[b]
end;
```

And a similar declaration for P.

The result of the calculation should be extracted from R and I and assigned to a complex variable. The simplest way to perform this is to write:

z:= S(a,S(b,c)) for z:= a + b + c.

Note then that the programmer has to augment p explicitly in order to save z from being erased in future calculations.
Moreover it is difficult to reuse the space of R and I if the value of z becomes uninteresting.
The simplest way to overcome the last difficulty is to reserve space in R and I for z. This should be done immediately after z is declared as integer variable.
One manner in which to program this is to use a second stack declared as integer array POINTER[1 : 50], with pointer kp.
Immediately after block entry, the current value of p is stored in POINTER, and space in R and I is reserved for the complex variables to be used. This space is erased by a procedure call ERASE immediately before block-exit, giving p its old value.

The declaration is thus extended with:

```
integer kp; integer array POINTER[1 : 50];
Boolean procedure DE(z,first); integer z; Boolean first;
begin if first then begin kp:= kp + 1; POINTER[kp]:= p; p:= p+1  end;
     z:= p; p:= p+1; DE:= false
end;
procedure ERASE; begin p:= POINTER[kp]; kp:= kp-1 end;
integer procedure ASSIGN(a,z); value a,z; integer a,z;
begin ASSIGN:= z;
     R[a]:= R[z]; I[a]:= I[z];
end;
```

Remarks

1. The construction of the procedure DE makes it possible to declare several variables by one statement. For example, the variables a, b and c by: DE(a,DE(b,DE(c,true))).

   One may prefer a simpler construction, that is, a procedure BLOCK ENTRY and a procedure DE such that block entry and declaration of a, b and c is performed by

   BLOCK ENTRY; DE(a); DE(b); DE(c)

   An advantage of our approach is that the explicit block entry call: DE(c,true) will in general not be forgotten as might "BLOCK ENTRY" in the other case.

2. Upon block entry, p is augmented in DE by 1. The reason is that in the calculation of a function designator for example: ARCTAN(S(a,b)), S(a,b) is stored at the top of the stack, so that the intermediate results of the ARCTAN calculation can not be stored at this top.

3. Due to the construction of the procedure ASSIGN, a repeated assignment is possible. That is to say a:= b:= z can be written as

ASSIGN(a,ASSIGN(b,z))

2.2. Directions for use

A program using the system of procedures, described in the following section, should have the following structure:

begin <declaration of the variables and procedures from the next section>

INITIALIZE;

<the declarations and statements defining the desired complex arithmetic computations>

end

The data on input tape should start with a number defining the upper bound of the arrays R and I.

All complex variables should be declared either as simple variables or as subscripted variables of type integer. If only simple variables occur then the declaration should be followed by a statement of the form:

<declaration statement> ::= DE(<variable>,true) |
DE(<variable>, <declaration statement>)

Example: DE(a,true)
DE(a,DE(b,DE(c,true)))

If subscripted variables also occur, then the declaration should be followed first by a declaration statement and second by a for statement in which DE is used with second parameter false.

Example: the variables a, v[1], ..., v[100] are declared by:

```
integer i, a; integer array v[1 : 100]; DE(a,true);
for i:= 1 step 1 until 100 do DE(v[i],false);
```

A calculation can only be performed through the use of an assignment statement, a modulus calculation statement, or an argument calculation statement of the following form:

```
<assignment statement>::= ASSIGN(<variable>,
                    <complex arithmetic expression>)|
                    ASSIGN(<variable>, <assignment statement>)

<modulus calculation statement>::=
                    mod(<complex arithmetic expression>)

<argument calculation statement>::=
                    arg(<complex arithmetic expression>)
```

A complex arithmetic expression is defined by:

```
<complex arithmetic expression>::= <variable> | 0 | 1 | iu|
            CN(<arithmetic expression>, <arithmetic expression>)|
            RN(<arithmetic expression>)|
            <sum> | <difference> | <product> | <quotient> |
            <power> | <integral power> |
            <sum with real number> |
            <product with real number> |
            <elementary function designator>

<cae>::= <complex arithmetic expression>
<sum>::= S(<cae>, <cae>)
<difference>::= D(<cae>, <cae>)
<product>::= P(<cae>, <cae>)
<quotient>::= Q(<cae>, <cae>)
<power>::= POWER(<cae>, <cae>)
<integral power>::= INT POW(<cae>, <arithmetic expression of type integer>)
<sum with real number>::= SR(<arithmetic expression>, <cae>)
<product with real number>::= PR(<arithmetic expression>, <cae>)
<elementary function designator>::= <el function identifier>(<cae>)
<el function identifier>::= EXP | SIN | COS | TAN | LN |
                            ARCSIN | ARCCOS | ARCTAN | SQRT
```

Remark: Complex arithmetic expressions occur in the ALGOL 60 program as particular expressions of type integer.

An example of an assignment statement is

ASSIGN(a,P(CN(0,-.5),LN(Q(SR(1,P(iu,z)),D(1,P(iu,z))))))

which corresponds to a:= -i/2 × ln((1+iz)/(1-iz))

The variable "iu" stands for imaginary unit with R[iu] = 0 and I[iu] = 1. Note that R[0] = 0, I[0] = 0, R[1] = 1 and I[1] = 0, so that the complex arithmetic expressions 0 and 1 refer to the complex numbers 0+0i and 1+0i.

Every block in which complex variables are declared should end with a procedure statement ERASE.

In using the elementary multivalued functions one may specify which value should be calculated by means of the non-local real variable LOWER BOUND ON ARGUMENT. This variable should be used in connection with the Boolean variable SPECIAL ARGUMENT.
If SPECIAL ARGUMENT has the value false, the principal values are calculated; if not, the values are calculated according to the wishes of the user specified by LOWER BOUND ON ARGUMENT.
More information is given in section 3.2.
Typical examples of how complex arithmetical computation may be programmed are the procedure body of ARCTAN (sect. 3.2.3) and the program of section 4.

3. The system of complex arithmetic procedures

This section contains the procedure declarations.
The real variables "pi" and "null" are used with the values 3.1415926535897 and $_{10}$-600 respectively.

3.1. The elementary operation and singlevalued functions

```
comment CALCULATION PROGRAM FOR COMPLEX ARITHMETIC
  R 1050 RPR 221266/06;
real pi,null,LOWER BOUND ON ARGUMENT;
integer p,kp,iu;
real array R,I[0:READ]; integer array POINTER[1:50];
comment The MC standard function READ reads a number from input tape;
Boolean SPECIAL ARGUMENT;
Boolean procedure DE(z,first); integer z; Boolean first;
comment DE should be used in declaring complex variables,
  for example a and b are declared by: DE(a,DE(b,true));
begin if first then begin kp:= kp + 1; POINTER[kp]:= p; p:= p + 1 end;
  z:= p; p:= p + 1; DE:= false
end DE;
procedure ERASE; comment ERASE should occur just before block exit;
begin p:= POINTER[kp]; kp:= kp - 1 end;
integer procedure ASSIGN(a,z); value a,z; integer a,z;
comment ASSIGN is used in an assignment statement a:= z;
begin ASSIGN:= z; R[a]:= R[z]; I[a]:= I[z] end ASSIGN;
integer procedure RN(x); value x; real x;
comment RN(x) is the real number x;
begin RN:= p; R[p]:= x; I[p]:= 0 end RN;
integer procedure CN(x,y); value x,y; real x,y;
comment CN(x,y) is the complex number x + i y;
begin CN:= p; R[p]:= x; I[p]:= y end CN;
integer procedure S(u,v); integer u,v; comment S:= u + v;
begin integer a,b; a:= u; p:= p + 1; b:= v; S:= p:= p - 1;
  R[p]:= R[a] + R[b]; I[p]:= I[a] + I[b]
end S;
integer procedure D(u,v); integer u,v; comment D:= u - v;
begin integer a,b; a:= u; p:= p + 1; b:= v; D:= p:= p - 1;
  R[p]:= R[a] - R[b]; I[p]:= I[a] - I[b]
end D;
```

```
integer procedure P(u,v); integer u,v; comment P:= u × v;
begin integer a,b; real ra,rb,ia,ib; a:= u; p:= p + 1; b:= v;
  P:= p:= p - 1; ra:= R[a]; rb:= R[b]; ia:= I[a]; ib:= I[b];
  R[p]:= ra × rb - ia × ib; I[p]:= ra × ib + ia × rb
end P;
integer procedure Q(u,v); integer u,v; comment Q:= u/v;
begin integer a,b; real c,ra,rb,ia,ib; a:= u; p:= p + 1; b:= v;
  Q:= p:= p - 1; ra:= R[a]; rb:= R[b]; ia:= I[a]; ib:= I[b];
  c:= rb × rb + ib × ib; R[p]:= (ra × rb + ia × ib)/c;
  I[p]:= (ia × rb - ra × ib)/c
end Q;
integer procedure POWER(u,v); integer u,v; comment POWER:= u ↑ v;
POWER:= EXP(P(LN(u),v));
integer procedure SR(r,z); value r,z; real r; integer z;
comment SR:= r + z, where r is real;
begin SR:= p; R[p]:= r + R[z]; I[p]:= I[z] end SR;
integer procedure PR(r,z); value r,z; real r; integer z;
comment PR:= r × z, where r is real;
begin PR:= p; R[p]:= r × R[z]; I[p]:= r × I[z] end PR;
integer procedure INT POW(z,n); value z,n; integer z,n;
comment INT POW:= z ↑ n;
if abs(I[z]) ≤ null then INT POW:= RN(R[z] ↑ n) else
if n < 0 then INT POW:= Q(1,INT POW(z,- n)) else
if n = 0 then INT POW:= 1 else
if n = 1 then INT POW:= z else
if n = 2 then INT POW:= CN(R[z] × R[z] - I[z] × I[z], 2 × R[z] × I[z]) else
begin integer k; DE(k,true); ASSIGN(k,INT POW(INT POW(z,n : 2),2));
  if (n : 2) × 2 - n ≠ 0 then ASSIGN(k,P(k,z));
  ERASE; ASSIGN(p,k); INT POW:= p
end INT POW;
```

```
integer procedure EXP(z); value z; integer z;
begin real r,i; r:= exp(R[z]); i:= I[z];
  EXP:= if abs(i) ≤ null then RN(r) else CN(r × cos(i),r × sin(i))
end EXP;
procedure hyperbolic function(sinh,cosh,y); value y; real sinh,cosh,y;
comment this procedure is taken from P.WYNN [2];
begin real y1; y1:= exp(y); cosh:= .5 × (y1 + 1/y1);
  if abs(y) ≥ 1.0 then sinh:= .5 × (y1 - 1/y1) else
  begin integer r; real br,brplus1,brplus2; array CWC[0:5];
    CWC[0]:= 1.13031 82079 8497; CWC[1]:= 4.43368 49848 66₁₀-2;
    CWC[2]:= 5.42926 31191₁₀-4; CWC[3]:= 3.19843 646₁₀-6;
    CWC[4]:= 1.10367 7₁₀-8; CWC[5]:= 2.498₁₀-11;
    brplus1:= brplus2:= 0.0; y1:= 2.0 × (2.0 × y × y - 1.0);
    for r:= 5 step -1 until 0 do
    begin br:= y1 × brplus1 - brplus2 + CWC[r]; if r ≠ 0 then
      begin brplus2:= brplus1; brplus1:= br end
    end;
    sinh:= y × (br - brplus1)
  end
end hyperbolic function;
integer procedure SIN(z); value z; integer z;
if abs(I[z]) ≤ null then SIN:= RN(sin(R[z])) else
begin real r,s,c; r:= R[z]; hyperbolic function(s,c,I[z]);
  SIN:= CN(sin(r) × c,cos(r) × s)
end SIN;
```

```
integer procedure COS(z); value z; integer z;
if abs(I[z]) ≤ null then COS:= RN(cos(R[z])) else
begin real r,s,c; r:= R[z]; hyperbolic function(s,c,I[z]);
  COS:= CN(cos(r) × c,- sin(r) × s)
end COS;
integer procedure TAN(z); value z; integer z;
if abs(I[z]) ≤ null then TAN:= RN(sin(R[z])/cos(R[z])) else
begin real s,c,si,co,r; r:= R[z];
  hyperbolic function(s,c,I[z]); si:= sin(r); co:= cos(r);
  TAN:= Q(CN(si × c,co × s),CN(co × c,-si × s))
end TAN;
```

3.2. The multivalued functions

The multivalued functions are defined by:

LOWER BOUND ON ARGUMENT $<arg(z) \leq$ LOWER BOUND ON ARGUMENT + 2pi

$LN(z) = \ln|z| + i\ arg(z)$

$SQRT(z) = EXP(\frac{1}{2} LN(z))$

$ARCSIN(z) = \frac{1}{i} LN(iz + SQRT(1 - z^2))$

$ARCCOS(z) = \frac{1}{i} LN(z + i\ SQRT(1 - z^2))$

$ARCTAN(z) = \frac{1}{2i} LN((1 + iz)/(1 - iz))$

We do not always use these definitions, however, to calculate the functions.
By means of certain transformations, e.g. addition of a multiple of 2pi, the resulting values are adapted in order to satisfy the above definitions. For this the procedure "mult of 2pi" is used.

Remark: It may occur that little changes of z result in completely different values for the multivalued functions. Therefore, one should be careful in choosing LOWER BOUND ON ARGUMENT. It may be desirable to choose it somewhat too large (say $_{10}$-10).

```
real procedure mult of 2pi(a); value a; real a;
mult of 2pi:= (entier((LOWER BOUND ON ARGUMENT - a)/
  6.28318530717958) + 1) × 6.28318530717958;
real procedure arg(z); value z; integer z;
begin real r,i,a; r:= R[z]; i:= I[z];
  a:= if abs(r) ≤ null then sign(i) × pi/2 else
      if r > null then arctan(i/r) else
      if abs(i) ≤ null then pi else
      sign(i) × pi/2 - arctan(r/i);
  if SPECIAL ARGUMENT then a:= a + mult of 2pi(a);
  arg:= a
end arg;
real procedure mod(z); value z; integer z;
mod:= sqrt(R[z] × R[z] + I[z] × I[z]);
```

3.2.1. The logarithmic function

The logarithm is in general calculated by means of the expression

$$\ln(z) = \ln|z| + i\ \arg(z)$$

However, when $z = a + i\varepsilon$ or $z = \varepsilon + ia$, where a is exactly equal to $\pm$ 1, the logarithm is calculated from the expression

$$\ln(1+z) = 2i \arctan \frac{z}{i(z+2)}$$

The reason for this is that the first expression gives an absolute error which is of the order of the relative error (δ) of ε, whereas the second formula gives a relative error of the order δ.
Obviously no improvement can be obtained from the second formula if a is not exactly equal to $\pm$ 1.

```
integer procedure LN(z); value z; integer z;
begin real r,i,n; r:= R[z]; i:= I[z];
  if (abs(r - 1) ≤ null ∧ abs(i) < .1) ∨ (abs(i - 1) ≤ null ∧ abs(r) < .1) then
  begin if abs(i) < .1 then
    begin n:= 4 + i × i; LN:= if sign(r) = +1 then
      P(CN(0,2),ARCTAN(CN(2 × i/n, - i × i/n))) else
      S(CN(0,pi × sign(i + null)),P(CN(0,2),ARCTAN(CN(- 2 × i/n,- i × i/n))))
    end else
    begin n:= 4 + r × r; LN:= if sign(i) = +1 then
      S(CN(0,pi/2),P(CN(0,2),ARCTAN(CN(- 2 × r/n, - r × r/n)))) else
      S(CN(0,-pi/2),P(CN(0,2),ARCTAN(CN(2 × r/n,- r × r /n))))
    end; if SPECIAL ARGUMENT then I[p]:= I[p] + mult of 2pi(I[p])
  end else LN:= CN(.5 × ln(r × r + i × i),arg(z))
end LN;
```

3.2.2. The square root function

```
integer procedure SQRT(z); value z; integer z;
begin real a,b,r,i; a:= R[z]; b:= I[z];
  r:= a × a + b × b; if abs(r) ≤ null then
  begin r:= 0; i:= 0; goto END end; r:= sqrt(r);
  r:= sqrt((abs(a) + r)/2); i:= b:= b/(2 × r);
  if a < 0 then
  begin i:= if b ≥ 0 then r else  - r; r:= abs(b) end;
  if SPECIAL ARGUMENT then
  begin real phi; integer k;
    SPECIAL ARGUMENT:= false;
    phi:= arg(CN(r,i));
    k:= entier((phi - LOWER BOUND ON ARGUMENT/2)/pi);
    if k : 2 × 2 - k ≠ 0 then begin r:= - r; i:= - i end;
    SPECIAL ARGUMENT:= true
  end;
END: SQRT:= CN(r,i)
end SQRT;
```

3.2.3. The arctan function

For $|z| > 2$ the ARCTAN is calculated by means of the formula

$$\arctan(z) = \pi/2 - \arctan(1/z).$$

For $.5 \leq |z| \leq 2$ it is calculated by means of

$$\arctan(z) = \frac{1}{2i} \ln((1+iz)/(1-iz))$$

Finally for $|z| < .5$ the ARCTAN is calculated by means of a process to be discussed below.

Let $a_0 = 1$, $b_0 = (1 + z^2)^{\frac{1}{2}}$,

$$a_{n+1} = (a_n + b_n)/2 \ , \quad b_{n+1} = (a_{n+1} \cdot b_n)^{\frac{1}{2}},$$

where the SQRT is always taken with positive real part i.e. SPECIAL ARGUMENT = false.

Let $z_n = a_n/b_n$. Then z_n satisfies

$$z_{n+1}^2 = (1 + z_n)/2, \text{ with solution } z_n = \cos(\phi/2^n) \text{ and}$$

with $z_0 = \cos \phi = (1 + z^2)^{-\frac{1}{2}}$.

Now $b_n = z_n \cdot b_{n-1} = (\prod_{k=1}^{n} z_k)(1 + z^2)^{\frac{1}{2}}$.

From $\sin \frac{\phi}{2^n} \cdot b_n = \frac{1}{2^n} \sin \phi \cdot (1 + z^2)^{\frac{1}{2}}$ it follows that

$$\lim_{n\to\infty} b_n = \frac{\sin \phi}{\phi} (1 + z^2)^{\frac{1}{2}} = \frac{z}{\arctan z}$$

This process, although very elegant, converges only linearly and about 20 iterations are needed to obtain $|a_n - b_n| < 10^{-12}$.

It will be shown, however, that if the process is discontinued as soon as $|a_n - b_n| < {}_{10}-3$, i.e. after 4 iterations maximally, then b_∞ can be estimated (without calculating the SQRT) within the desired accuracy of ${}_{10}-12$.
(It turned out that this calculated value was better than the value obtained after 20 iterations.)

Let the iteration be discontinued for n = m, and let $a_m - b_m = \delta$, then b^*_{m+1} is calculated from:

$$b^*_{m+1} = (a_{m+1} + b_m)/2 = (a_m + 3b_m)/4.$$

Now $$b^{*\,2}_{m+1} - b^2_{m+1} = \frac{(a_{m+1} - b_m)^2}{4}$$

and $$b^*_{m+1} - b_{m+1} = \frac{\delta^2}{16} \frac{1}{b^*_{m+1} + b_{m+1}} \quad .$$

Thus $$b^*_{m+1} - b_{m+1} = \frac{\delta^2}{32 \cdot b^*_{m+1}} + \ldots$$

However,

$$\frac{1}{b^*_{m+1} + b_{m+1}} = \frac{1}{2b^*_{m+1}\left(1 + \frac{b_{m+1} - b^*_{m+1}}{2b^*_{m+1}}\right)} =$$

$$= \frac{1}{2b^*_{m+1}} \left(1 + \frac{\delta^2}{64\, b^{*\,2}_{m+1}} + \ldots\right),$$

thus

$$b^{**}_{m+1} = b^*_{m+1} - \frac{\delta^2}{32\, b^*_{m+1}} \quad \text{satisfies:}$$

$$b^{**}_{m+1} - b_{m+1} = \frac{\delta^4}{2^{11}\, b^{*\,3}_{m+1}} + O(\delta^5).$$

Furthermore, denoting the limit of b_n by $M(a_1,b_1)$, we have

$$b_\infty = M(a_1,b_1) = M(a_{m+1},b_{m+1}) = b_{m+1}\, M(\frac{a_{m+1}}{b_{m+1}}\ ,\ 1).$$

From $M(x,1) = \dfrac{\xi}{\arcsin\ \xi}$, with $\xi = (1 - x^2)^{\frac{1}{2}}$ we have

$$M(x,1) = 1/(1 + \frac{1}{6}\,\xi^2 + \frac{3}{40}\,\xi^4 + \frac{5}{112}\,\xi^6 + \frac{35}{1152}\,\xi^8 + \ldots\ )$$

and thus with $\xi^2 = \dfrac{-\delta}{2b_m}$

$$b_\infty = b_{m+1}/(1 + \frac{1}{6}\,\xi^2 + \ldots\ ) \text{ with } \xi^2 = \frac{-\delta}{2b_m}$$

$$= b^{**}_{m+1}/(1 + \frac{1}{6}\,\xi^2 + \frac{3}{40}\,\xi^4 + \frac{5}{112}\,\xi^6) +$$

$$+ \delta^4(\frac{1}{2^{11}\ b^{**3}_{m+1}} - \frac{b^{**}_{m+1}}{b^4_m} \cdot \frac{35}{1152}) + O(\delta^5).$$

Lower bounds for b^{*}_{m+1}, b^{**}_{m+1} and b_{m+1} will now be obtained.
Consider the region $G_k = \{z : \mathrm{Re}(z) \geq k > 0\}$ if $a_n \in G_k$, $b_n \in G_k$ then $a_{n+1} \in G_k$, due to the convexity of G_k.
Consider next the region $H_k = \{w : \exists\, z \in G_k,\ w = \ln(z)\}$, where $\ln(z)$ assumes its principal value. H_k is a convex region in the w plane, as the boundary is given by: $w = u + iv = \frac{1}{2}\ln(k^2 + y^2) + i \arctan y/k$ and

$$\frac{d^2u}{dv^2} = \frac{d}{dy}\,(\frac{du}{dy} \cdot \frac{1}{dv/dy}) \cdot \frac{1}{dv/dy}$$

$$= \frac{d}{dy}\,(\frac{y}{k^2 + y^2} \cdot \frac{k^2 + y^2}{k}) \cdot \frac{k^2 + y^2}{k} = \frac{k^2 + y^2}{k^2} > 0.$$

Thus, if $w_1 \in H_k$, $w_2 \in H_k$ then $(w_1 + w_2)/2 \in H_k$.
Choose $w_1 = \ln(a_{m+1})$, $w_2 = \ln(b_n)$, then

$$\ln(a_{m+1} \cdot b_n)^{\frac{1}{2}} = \ln(b_{n+1}) \in H_k$$

thus $b_{n+1} \in G_k$.

If $|z| = \alpha < .5$ then

$$\mathrm{Re}(b_0) = \mathrm{Re}((1 + z^2)^{\frac{1}{2}}) \geq (1 - \alpha^2)^{\frac{1}{2}}$$

and k can be taken $\frac{1}{2}\sqrt{3}$.
Thus all a_n and b_n satisfy $\mathrm{Re}(a_n) > \frac{1}{2}\sqrt{3}$, $\mathrm{Re}(b_n) > \frac{1}{2}\sqrt{3}$. Which, in particular, is the case for b_m, a_{m+1} and thus for b^*_{m+1}.
Hence $|b_{m+1}| > \frac{1}{2}\sqrt{3}$, $|b^*_{m+1}| > \frac{1}{2}\sqrt{3}$ and

$$|b^{**}_{m+1}| > \frac{1}{2}\sqrt{3} - |\delta^4|.$$

From this it follows that if the calculations were performed exactly, the error made in estimating b_∞ is considerably less than $_{10}$-12. Theoretically the bound $|z| < .5$ may be made less restrictive, say $|z| < .99$, but in practice it turned out that the ordinary way of calculating arctan(z) was more precise and less time consuming for values of z in this region.

```
integer procedure ARCTAN(z); value z; integer z;
begin real modz; integer a; DE(a,true); modz:= mod(z);
  if modz > 2 then
  begin ASSIGN(a,D(RN(pi/2),ARCTAN(Q(1,z))));
    if SPECIAL ARGUMENT then R[a]:=  R[a] + mult of 2pi(2 × R[a])/2
  end else
  if modz < .5 then
  begin integer b,delta,ksi,bstar; Boolean SA; SA:= false;
    DE(b,DE(delta,DE(ksi,DE(bstar,true))));
    if SPECIAL ARGUMENT then
    begin SA:= true; SPECIAL ARGUMENT:= false end;
    ASSIGN(a,1); ASSIGN(b,SQRT(SR(1,P(z,z))));
  again: ASSIGN(delta,D(a,b));
    if mod(delta) ≤ 10-3 then goto out;
    ASSIGN(a,PR(.5,S(a,b)));
    ASSIGN(b,SQRT(P(a,b))); goto again;
```

```
  out: ASSIGN(ksi,PR(-.5,Q(delta,b)));
    ASSIGN(bstar,S(PR(.25,a),PR(.75,b)));
    ASSIGN(bstar,D(bstar,Q(P(delta,delta),PR(32,bstar))));
    ASSIGN(a,Q(P(z,SR(1,P(ksi,SR(1/6,P(ksi,SR(3/40,
    PR(5/112,ksi)))))),bstar));
  if SA then
    begin SPECIAL ARGUMENT:= true; R[a]:= R[a] + mult of 2pi(2 × R[a])/2
    end; ERASE
  end else ASSIGN(a,P(CN(0,-.5),LN(Q(SR(1,P(iu,z)),D(1,P(iu,z))))));
  ERASE; ASSIGN(p,a); ARCTAN:= p
end ARCTAN;
```

3.2.4. The arcsin function

For $|z| < .5$ ARCSIN is calculated by means of the expression:

$$\arcsin z = \arctan(z/(1 - z^2)^{\frac{1}{2}}).$$

For $|z| \geq .5$ ARCSIN is calculated by means of two different expressions. If $|iz + \sqrt{1 - z^2}| \geq 1$, it is calculated from

$$\arcsin z = \frac{1}{i} \ln(iz + \sqrt{1 - z^2})$$

and in the other case from

$$\arcsin z = -\frac{1}{i} \ln(-iz + \sqrt{1 - z^2}).$$

```
integer procedure ARCSIN(z); value z; integer z;
begin real modz; integer a,b,k; Boolean SA; DE(a,DE(b,true)); modz:= mod(z);
  if modz < .5 then
  begin SA:= false; if SPECIAL ARGUMENT then
    begin SA:= true; SPECIAL ARGUMENT:= false end;
```

```
      ASSIGN(a,ARCTAN(Q(z,SQRT(D(1,P(z,z))))));
      if SA then
      begin SPECIAL ARGUMENT:= true;
        k:= entier(LOWER BOUND ON ARGUMENT/(2 × pi));
        ASSIGN(a,if k - (k : 2) × 2 = 0 then
          D(RN(pi + mult of 2pi(pi - R[a])),a) else
          SR(mult of 2pi(R[a]),a))
      end
    end else
    begin ASSIGN(a,SQRT(D(1,P(z,z))));
      ASSIGN(b,S(P(iu,z),a)); if mod(b) ≥ 1 then
      ASSIGN(a,P(CN(0,-1),LN(b))) else
      begin ASSIGN(a,P(iu,LN(D(a,P(iu,z)))));
        if SPECIAL ARGUMENT then ASSIGN(a,SR(mult of 2pi(R[a]),a))
      end
    end; ERASE; ASSIGN(p,a); ARCSIN:= p
end ARCSIN;
```

3.2.5. The arccos function

For $|z + i\sqrt{1 - z^2}| \geq 1$ ARCCOS is calculated from

$$\arccos z = \frac{1}{i} \ln(z + i\sqrt{1 - z^2})$$

and in the other case from

$$\arccos z = -\frac{1}{i} \ln(z - i\sqrt{1 - z^2}).$$

To use a special formula for small $|z|$ is not necessary since

$$\arccos (z(1 + \delta)) \approx \arccos z - z\delta$$

and $$\arccos(z) \approx \pi/2 - z$$

thus the relative error δ remains a relative error in arccos(z).

```
integer procedure ARCCOS(z); value z; integer z;
begin integer a,b; DE(a,DE(b,true));
  ASSIGN(a,P(iu,SQRT(D(1,P(z,z)))));
  ASSIGN(b,S(z,a)); if mod(b) ≥ 1 then
  ASSIGN(a,P(CN(0,-1),LN(b))) else
  begin ASSIGN(a,P(iu,LN(D(z,a))));
    if SPECIAL ARGUMENT then
    ASSIGN(a,SR(mult of 2pi(R[a]),a))
  end; ERASE; ASSIGN(p,a); ARCCOS:= p
end ARCCOS;
```

3.2.6. The procedure INITIALIZE

```
procedure INITIALIZE;
begin p:= 0; pi:= 3.14159265358979; null:= 10-600;
  CN(0,0); p:= 1; CN(1,0); p:= iu:= 2; CN(0,1); kp:= 1; POINTER[1]:= p:= 3;
  LOWER BOUND ON ARGUMENT:= - pi + 10-12; SPECIAL ARGUMENT:= false
end;
```

4. A test program

In this section some results with regard to error analysis and computation time are given.
Moreover a test program is reproduced together with its output.
This test program performs the main part of the error analysis.

It calculates for

LOWER BOUND ON ARGUMENT = - pi, - pi/2, 0, pi/2, pi, 3/2 pi and 2pi,

for

Re(z) = -1000, -10, -1, -.1, -.001, 0, .001, .1, 1, 10 and 1000

and for

Im(z) = -1000, -10, -1, -.1, -.001, 0, .001, .1, 1, 10 and 1000,

the maximum in each quadrant of the complex z plane of the following expressions:

$|z - EXP(LN(z))|/|z|$,

$|z - (z^{1/3})^3|/|z|$,

$|z - (SQRT(z))^2|/|z|$,

$|z - SIN(ARCSIN(z))|/|z|$,

$|z - COS(ARCCOS(z))|/|z|$ and

$|z - TAN(ARCTAN(z))|/|z|$.

No calculations were performed in the last but one and the last cases for $|z| \leq .1$ and $|z| \geq 10$ respectively.

The output of the program appears in five columns. The first column indicates which function was tested, the next four columns contain for each quadrant of the z plane the following numbers: max error, real part of z max, imaginary part of z max, where max error is the maximum of all errors calculated in the corresponding quadrant and where z max is the value of z for which the maximal error occurred.

```
begin <declaration of the variables and procedures of section 3>
  INITIALIZE;
  begin comment Test program for Complex Arithmetic RPR 190367/01;
    integer case,i,z; real real part of z, imag part of z, mod of z;
    array max error[1:4,1:6]; integer array z max[1:4,1:6];
    procedure ERROR(f); integer f;
    begin real error; i:= i + 1;
      error:= mod(D(z,f))/mod of z;
      if max error[case,i] < error then
      begin max error[case,i]:= error; ASSIGN(z max[case,i],z) end;
    end ERROR;
    procedure OUTPUT(string); string string;
    begin i:= i + 1; NLCR; PRINTTEXT(string);
      for case:= 1,2,3,4 do
      begin FLOT(2,2,max error[case,i]);
        FLOT(1,1,R[z max[case,i]]);
        FLOT(1,1,I[z max[case,i]]);
        comment The procedure FLOT prints the value of its
          third parameter in floating point notation;
      end; PRINTTEXT(‡ ‡)
    end OUTPUT;
    DE(z,true); for case:= 1,2,3,4 do
    for i:= 1,2,3,4,5,6 do DE(z max[case,i],false);
    NLCR; PRINTTEXT(‡Results from test program RPR 190367/01‡); NLCR;
    for LOWER BOUND ON ARGUMENT:= -pi+10-10, -pi/2+10-10, 10-10,
      pi/2+10-10, pi+10-10, 3/2×pi+10-10, 2×pi+10-10 do
    begin for case:= 1,2,3,4 do for i:= 1,2,3,4,5,6 do
      begin ASSIGN(z max[case,i],0); max error[case,i]:= 0 end;
      for real part of z:=
        -1000, -10, -1, -.1, -.001, 0, .001, .1, 1, 10, 1000 do
      for imag part of z:=
        -1000, -10, -1, -.1, -.001, 0, .001, .1, 1, 10, 1000 do
```

```
      begin if abs(real part of z) + abs(imag part of z) < 10-5
        then goto END;
        i:= 0; ASSIGN(z,CN(real part of z, imag part of z));
        mod of z:= mod(z);
        case:= entier(arg(z)/(pi/2) - 10-5);
        case:= case - entier(case/4) × 4 + 1;
        ERROR(EXP(LN(z)));
        ERROR(INT POW(POWER(z,RN(1/3)),3));
        ERROR(INT POW(SQRT(z),2));
        ERROR(SIN(ARCSIN(z)));
        if mod of z > .1 then ERROR(COS(ARCCOS(z))) else i:= i + 1;
        if mod of z < 10 then ERROR(TAN(ARCTAN(z)));
      END: end; SPECIAL ARGUMENT:= true; i:= 0;
      NLCR; PRINTTEXT(‹LOWER BOUND ON ARGUMENT =›);
      FIXT(1,1,LOWER BOUND ON ARGUMENT/pi); PRINTTEXT(‹× pi›);
      OUTPUT(
          ‹EXP(LN(z))       ›);
      OUTPUT(
          ‹(z↑1/3)↑3        ›);
      OUTPUT(
          ‹SQRT(z)↑2        ›);
      OUTPUT(
          ‹SIN(ARCSIN(z))   ›);
      OUTPUT(
          ‹COS(ARCCOS(z))   ›);
      OUTPUT(
          ‹TAN(ARCTAN(z))›)
    end
  end; ERASE
end
```

The input tape contains the number 100.

Results from test program RPR 190367/01

LOWER BOUND ON ARGUMENT =-1.0 × pi				
EXP(LN(z))	+.56$_{10}$-11 +.1$_{10}$+1 +.1$_{10}$+4	+.58$_{10}$-11 -.1$_{10}$+4 +.1$_{10}$+1	+.58$_{10}$-11 -.1$_{10}$+4 -.1$_{10}$+1	+.56$_{10}$-11 +.1$_{10}$+1 -.1$_{10}$+4
(z↑1/3)↑3	+.13$_{10}$-10 +.1$_{10}$-2 +.1$_{10}$+4	+.13$_{10}$-10 -.1$_{10}$+4 +.1$_{10}$-0	+.13$_{10}$-10 -.1$_{10}$+4 -.1$_{10}$-0	+.14$_{10}$-10 +.1$_{10}$+4 -.1$_{10}$-2
SQRT(z)↑2	+.35$_{10}$-11 +.1$_{10}$-0 +.1$_{10}$+2	+.35$_{10}$-11 -.1$_{10}$-0 +.1$_{10}$+2	+.35$_{10}$-11 -.1$_{10}$-0 -.1$_{10}$+2	+.35$_{10}$-11 +.1$_{10}$-0 -.1$_{10}$+2
SIN(ARCSIN(z))	+.84$_{10}$-11 +.0$_{10}$-0 +.1$_{10}$+4	+.75$_{10}$-11 -.1$_{10}$+4 +.1$_{10}$+1	+.75$_{10}$-11 -.1$_{10}$+2 -.1$_{10}$+4	+.75$_{10}$-11 +.1$_{10}$+2 -.1$_{10}$+4
COS(ARCCOS(z))	+.20$_{10}$-10 +.1$_{10}$-0 +.1$_{10}$-2	+.34$_{10}$-10 -.1$_{10}$-0 +.1$_{10}$-2	+.34$_{10}$-10 -.1$_{10}$-0 -.1$_{10}$-2	+.20$_{10}$-10 +.1$_{10}$-0 -.1$_{10}$-2
TAN(ARCTAN(z))	+.48$_{10}$-11 +.1$_{10}$+1 +.1$_{10}$-0	+.57$_{10}$-11 -.1$_{10}$-0 +.0$_{10}$-0	+.46$_{10}$-11 -.1$_{10}$+1 -.1$_{10}$-0	+.57$_{10}$-11 +.1$_{10}$-0 +.0$_{10}$-0
LOWER BOUND ON ARGUMENT = -.5 × pi				
EXP(LN(z))	+.56$_{10}$-11 +.1$_{10}$+1 +.1$_{10}$+4	+.58$_{10}$-11 -.1$_{10}$+4 +.1$_{10}$+1	+.82$_{10}$-11 -.1$_{10}$+4 -.1$_{10}$+2	+.56$_{10}$-11 +.1$_{10}$+1 -.1$_{10}$+4
(z↑1/3)↑3	+.13$_{10}$-10 +.1$_{10}$-2 +.1$_{10}$+4	+.13$_{10}$-10 -.1$_{10}$+4 -.1$_{10}$-2	+.14$_{10}$-10 -.1$_{10}$-2 -.1$_{10}$+4	+.14$_{10}$-10 +.1$_{10}$+4 -.1$_{10}$-2
SQRT(z)↑2	+.35$_{10}$-11 +.1$_{10}$-0 +.1$_{10}$+2	+.35$_{10}$-11 -.1$_{10}$-0 +.1$_{10}$+2	+.35$_{10}$-11 -.1$_{10}$-0 -.1$_{10}$+2	+.35$_{10}$-11 +.1$_{10}$-0 -.1$_{10}$+2
SIN(ARCSIN(z))	+.84$_{10}$-11 +.0$_{10}$-0 +.1$_{10}$+4	+.75$_{10}$-11 -.1$_{10}$+4 +.1$_{10}$+1	+.10$_{10}$-10 -.1$_{10}$+4 -.1$_{10}$+1	+.75$_{10}$-11 +.1$_{10}$+2 -.1$_{10}$+4
COS(ARCCOS(z))	+.20$_{10}$-10 +.1$_{10}$-0 +.1$_{10}$-2	+.34$_{10}$-10 -.1$_{10}$-0 +.1$_{10}$-2	+.34$_{10}$-10 -.1$_{10}$-0 -.1$_{10}$-2	+.20$_{10}$-10 +.1$_{10}$-0 -.1$_{10}$-2
TAN(ARCTAN(z))	+.48$_{10}$-11 +.1$_{10}$+1 +.1$_{10}$-0	+.57$_{10}$-11 -.1$_{10}$-0 +.0$_{10}$-0	+.37$_{10}$-11 -.1$_{10}$+1 -.1$_{10}$-2	+.57$_{10}$-11 +.1$_{10}$-0 +.0$_{10}$-0
LOWER BOUND ON ARGUMENT = +.0 × pi				
EXP(LN(z))	+.62$_{10}$-11 +.1$_{10}$-2 +.1$_{10}$+1	+.58$_{10}$-11 -.1$_{10}$+4 +.1$_{10}$+1	+.98$_{10}$-11 +.1$_{10}$-2 -.1$_{10}$+4	+.12$_{10}$-10 +.1$_{10}$+4 -.1$_{10}$+1
(z↑1/3)↑3	+.13$_{10}$-10 +.1$_{10}$-2 +.1$_{10}$+4	+.13$_{10}$-10 -.1$_{10}$+4 -.1$_{10}$-2	+.16$_{10}$-10 +.1$_{10}$-2 -.1$_{10}$+4	+.15$_{10}$-10 +.1$_{10}$-0 -.1$_{10}$+1
SQRT(z)↑2	+.35$_{10}$-11 +.1$_{10}$-0 +.1$_{10}$+2	+.35$_{10}$-11 -.1$_{10}$-0 +.1$_{10}$+2	+.35$_{10}$-11 -.1$_{10}$-0 -.1$_{10}$+2	+.35$_{10}$-11 +.1$_{10}$-0 -.1$_{10}$+2
SIN(ARCSIN(z))	+.29$_{10}$- 8 +.0$_{10}$-0 +.1$_{10}$-2	+.41$_{10}$- 8 -.1$_{10}$-2 +.0$_{10}$-0	+.29$_{10}$- 8 +.0$_{10}$-0 -.1$_{10}$-2	+.45$_{10}$- 8 +.1$_{10}$-2 +.0$_{10}$-0
COS(ARCCOS(z))	+.12$_{10}$- 9 +.1$_{10}$-0 +.1$_{10}$-2	+.34$_{10}$-10 -.1$_{10}$-0 +.1$_{10}$-2	+.56$_{10}$-10 -.1$_{10}$-0 -.1$_{10}$-0	+.47$_{10}$-10 +.1$_{10}$-0 -.1$_{10}$-2
TAN(ARCTAN(z))	+.29$_{10}$- 8 +.0$_{10}$-0 +.1$_{10}$-2	+.16$_{10}$- 8 -.1$_{10}$-2 +.0$_{10}$-0	+.29$_{10}$- 8 +.0$_{10}$-0 -.1$_{10}$-2	+.57$_{10}$-11 +.1$_{10}$-0 +.0$_{10}$-0

LOWER BOUND ON ARGUMENT = +.5 × pi

EXP(LN(z))	$+.13_{10}-10$ $+.1_{10}+4$ $+.1_{10}+2$	$+.58_{10}-11$ $-.1_{10}+4$ $+.1_{10}+1$	$+.98_{10}-11$ $+.1_{10}-2$ $-.1_{10}+4$	$+.12_{10}-10$ $+.1_{10}+4$ $-.1_{10}+1$
(z↑1/3)↑3	$+.17_{10}-10$ $+.0_{10}-0$ $+.1_{10}+4$	$+.13_{10}-10$ $-.1_{10}+4$ $-.1_{10}-2$	$+.16_{10}-10$ $+.1_{10}-2$ $-.1_{10}+4$	$+.15_{10}-10$ $+.1_{10}-0$ $-.1_{10}+1$
SQRT(z)↑2	$+.35_{10}-11$ $+.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $-.1_{10}+2$	$+.35_{10}-11$ $+.1_{10}-0$ $-.1_{10}+2$
SIN(ARCSIN(z))	$+.29_{10}-8$ $+.0_{10}-0$ $+.1_{10}-2$	$+.41_{10}-8$ $-.1_{10}-2$ $+.0_{10}-0$	$+.29_{10}-8$ $+.0_{10}-0$ $-.1_{10}-2$	$+.45_{10}-8$ $+.1_{10}-2$ $+.0_{10}-0$
COS(ARCCOS(z))	$+.12_{10}-9$ $+.1_{10}-0$ $+.1_{10}-2$	$+.56_{10}-10$ $-.1_{10}-0$ $+.1_{10}-0$	$+.56_{10}-10$ $-.1_{10}-0$ $-.1_{10}-0$	$+.12_{10}-9$ $+.1_{10}-0$ $-.1_{10}-2$
TAN(ARCTAN(z))	$+.29_{10}-8$ $+.0_{10}-0$ $+.1_{10}-2$	$+.16_{10}-8$ $-.1_{10}-2$ $+.0_{10}-0$	$+.29_{10}-8$ $+.0_{10}-0$ $-.1_{10}-2$	$+.12_{10}-8$ $+.1_{10}-2$ $+.0_{10}-0$

LOWER BOUND ON ARGUMENT =+1.0 × pi

EXP(LN(z))	$+.13_{10}-10$ $+.1_{10}+4$ $+.1_{10}+2$	$+.12_{10}-10$ $-.1_{10}+4$ $+.1_{10}+1$	$+.98_{10}-11$ $+.1_{10}-2$ $-.1_{10}+4$	$+.12_{10}-10$ $+.1_{10}+4$ $-.1_{10}+1$
(z↑1/3)↑3	$+.17_{10}-10$ $+.0_{10}-0$ $+.1_{10}+4$	$+.16_{10}-10$ $-.1_{10}+4$ $+.1_{10}-0$	$+.16_{10}-10$ $+.1_{10}-2$ $-.1_{10}+4$	$+.15_{10}-10$ $+.1_{10}-0$ $-.1_{10}+1$
SQRT(z)↑2	$+.35_{10}-11$ $+.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $-.1_{10}+2$	$+.35_{10}-11$ $+.1_{10}-0$ $-.1_{10}+2$
SIN(ARCSIN(z))	$+.72_{10}-8$ $+.1_{10}-2$ $+.1_{10}-2$	$+.41_{10}-8$ $-.1_{10}-2$ $+.0_{10}-0$	$+.88_{10}-9$ $-.1_{10}-2$ $-.1_{10}-2$	$+.72_{10}-8$ $+.1_{10}-2$ $-.1_{10}-2$
COS(ARCCOS(z))	$+.12_{10}-9$ $+.1_{10}-0$ $+.1_{10}-2$	$+.56_{10}-10$ $-.1_{10}-0$ $+.1_{10}-0$	$+.56_{10}-10$ $-.1_{10}-0$ $-.1_{10}-0$	$+.12_{10}-9$ $+.1_{10}-0$ $-.1_{10}-2$
TAN(ARCTAN(z))	$+.29_{10}-8$ $+.0_{10}-0$ $+.1_{10}-2$	$+.16_{10}-8$ $-.1_{10}-2$ $+.0_{10}-0$	$+.29_{10}-8$ $+.0_{10}-0$ $-.1_{10}-2$	$+.12_{10}-8$ $+.1_{10}-2$ $+.0_{10}-0$

LOWER BOUND ON ARGUMENT =+1.5 × pi

EXP(LN(z))	$+.13_{10}-10$ $+.1_{10}+4$ $+.1_{10}+2$	$+.15_{10}-10$ $-.1_{10}+4$ $-.1_{10}-2$	$+.19_{10}-10$ $-.1_{10}-2$ $-.1_{10}+4$	$+.12_{10}-10$ $+.1_{10}+4$ $-.1_{10}+1$
(z↑1/3)↑3	$+.17_{10}-10$ $+.0_{10}-0$ $+.1_{10}+4$	$+.21_{10}-10$ $-.1_{10}+4$ $-.1_{10}-2$	$+.19_{10}-10$ $-.1_{10}+1$ $-.1_{10}-2$	$+.15_{10}-10$ $+.1_{10}-0$ $-.1_{10}+1$
SQRT(z)↑2	$+.35_{10}-11$ $+.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $-.1_{10}+2$	$+.35_{10}-11$ $+.1_{10}-0$ $-.1_{10}+2$
SIN(ARCSIN(z))	$+.72_{10}-8$ $+.1_{10}-2$ $+.1_{10}-2$	$+.98_{10}-8$ $-.1_{10}-2$ $+.0_{10}-0$	$+.88_{10}-9$ $-.1_{10}-2$ $-.1_{10}-2$	$+.72_{10}-8$ $+.1_{10}-2$ $-.1_{10}-2$
COS(ARCCOS(z))	$+.12_{10}-9$ $+.1_{10}-0$ $+.1_{10}-2$	$+.64_{10}-10$ $-.1_{10}-2$ $+.1_{10}-0$	$+.64_{10}-10$ $-.1_{10}-2$ $-.1_{10}-0$	$+.12_{10}-9$ $+.1_{10}-0$ $-.1_{10}-2$
TAN(ARCTAN(z))	$+.29_{10}-8$ $+.0_{10}-0$ $+.1_{10}-2$	$+.16_{10}-8$ $-.1_{10}-2$ $+.0_{10}-0$	$+.29_{10}-8$ $+.0_{10}-0$ $-.1_{10}-2$	$+.12_{10}-8$ $+.1_{10}-2$ $+.0_{10}-0$

LOWER BOUND ON ARGUMENT =+2.0 × pi

EXP(LN(z))	$+.13_{10}-10$ $+.1_{10}+4$ $+.1_{10}+2$	$+.15_{10}-10$ $-.1_{10}+4$ $-.1_{10}-2$	$+.19_{10}-10$ $-.1_{10}-2$ $-.1_{10}+4$	$+.18_{10}-10$ $+.1_{10}+4$ $-.1_{10}-0$
(z↑1/3)↑3	$+.17_{10}-10$ $+.0_{10}-0$ $+.1_{10}+4$	$+.21_{10}-10$ $-.1_{10}+4$ $-.1_{10}-2$	$+.19_{10}-10$ $-.1_{10}+1$ $-.1_{10}-2$	$+.14_{10}-10$ $+.1_{10}+4$ $+.1_{10}-2$
SQRT(z)↑2	$+.35_{10}-11$ $+.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $+.1_{10}+2$	$+.35_{10}-11$ $-.1_{10}-0$ $-.1_{10}+2$	$+.35_{10}-11$ $+.1_{10}-0$ $-.1_{10}+2$
SIN(ARCSIN(z))	$+.11_{10}-7$ $+.0_{10}-0$ $+.1_{10}-2$	$+.13_{10}-7$ $-.1_{10}-2$ $+.0_{10}-0$	$+.11_{10}-7$ $+.0_{10}-0$ $-.1_{10}-2$	$+.98_{10}-8$ $+.1_{10}-2$ $+.0_{10}-0$
COS(ARCCOS(z))	$+.64_{10}-10$ $+.1_{10}-2$ $+.1_{10}-0$	$+.64_{10}-10$ $-.1_{10}-2$ $+.1_{10}-0$	$+.62_{10}-10$ $-.1_{10}-0$ $-.1_{10}-2$	$+.62_{10}-10$ $+.1_{10}-0$ $-.1_{10}-2$
TAN(ARCTAN(z))	$+.57_{10}-8$ $+.0_{10}-0$ $+.1_{10}-2$	$+.10_{10}-7$ $-.1_{10}-2$ $+.0_{10}-0$	$+.57_{10}-8$ $+.0_{10}-0$ $-.1_{10}-2$	$+.12_{10}-8$ $+.1_{10}-2$ $+.0_{10}-0$

Some other tests were performed for ARCTAN, ARCSIN and ARCCOS.
z assumed the values

$$z = \rho\ e^{i\phi}$$

where ϕ = 0, (pi/10), 1.9 pi.

Only the principal values were calculated, i.e. SPECIAL ARGUMENT = false.

The results are:

|z - TAN(ARCTAN(z))|/ρ

reached for ρ = .49 the maximal value $.87_{10}$-11 for ϕ = 1.9 pi

" ρ = .1 " " " $.69_{10}$-11 " ϕ = 1.8 pi

" ρ = .001 " " " $.36_{10}$-11 " ϕ = 0

" ρ = 10 " " " $.36_{10}$-10 " ϕ = .8 pi

|z - SIN(ARCSIN(z))|/ρ

reached for ρ = .49 the maximal value $.59_{10}$-11 for ϕ = 1.7 pi

" ρ = .1 " " " $.57_{10}$-11 " ϕ = 1.1 pi

" ρ = .001 " " " $.36_{10}$-11 " ϕ = .2 pi

" ρ = 10 " " " $.39_{10}$-11 " ϕ = 1.2 pi

" ρ = 100 " " " $.54_{10}$-11 " ϕ = 1.7 pi

|z - COS(ARCCOS(z))|/ρ

reached for ρ = 10 the maximal value $.47_{10}$-11 for ϕ = .2 pi

" ρ = 100 " " " $.52_{10}$-11 " ϕ = 1.7 pi

The time needed for the different operations was

for

ASSIGN	.72	m.sec.
S	.92	m.sec.
D	.92	m.sec.
P	1.32	m.sec.
Q	1.52	m.sec.
INTPOW(z,2)	1.92	m.sec.
INTPOW(z,5)	9.52	m.sec.
INTPOW(z,10)	15.5	m.sec.
EXP	3.12	m.sec.
LN(1+i)	3.92	m.sec.
LN(1+.1i)	42.9	m.sec.
SIN	7.92	m.sec.
COS	7.92	m.sec.
TAN	10.7	m.sec.
ARCSIN(1+i)	21.9	m.sec.
ARCSIN(.01+.01i)	50.9	m.sec.
ARCTAN(1+i)	15.9	m.sec.
ARCTAN(.01+.01i)	38.9	m.sec.
ARCCOS(1+i)	18.9	m.sec.
ARCCOS(.01+.01i)	18.9	m.sec.
SQRT	2.32	m.sec.

References

[1] R.P. van de Riet — Formula manipulation in ALGOL 60 (preliminary report)
TW 101 Mathematical Centre 1966.

[2] P. Wynn — An arsenal of ALGOL procedures for Complex Arithmetic
B.I.T. 2 (1962) p. 232 - 255.