Complex Interval Functions in C-XSC

Blomquist, F., Hofschuster, W., Krämer, W., Neher, M.

Preprint 2005/2

Wissenschaftliches Rechnen/
Softwaretechnologie

Impressum

Herausgeber:	Prof. Dr. W. Krämer, Dr. W. Hofschuster Wissenschaftliches Rechnen/Softwaretechnologie Fachbereich C (Mathematik und Naturwissenschaften) Bergische Universität Wuppertal Gaußstr. 20 D-42097 Wuppertal

Internet-Zugriff

Die Berichte sind in elektronischer Form erhältlich über die World Wide Web Seiten

`http://www.math.uni-wuppertal.de/wrswt/literatur.html`

Autoren-Kontaktadressen

Frithjof Blomquist
Adlerweg 6
D-66436 Püttlingen

E-mail: `blomquist@math.uni-wuppertal.de`

Werner Hofschuster, Walter Krämer
Bergische Universität Wuppertal
Gaußstr. 20
D-42097 Wuppertal

E-mail: {`hofschuster, kraemer`}`@math.uni-wuppertal.de`

Markus Neher
Institut für Angewandte Mathematik
Universität Karlsruhe (TH)
D-76128 Karlsruhe

E-mail: `markus.neher@math.uni-karlsruhe.de`

Complex Interval Functions in C-XSC

Blomquist, F., Hofschuster, W., Krämer, W., Neher, M.

Contents

Abstract

In **C-XSC** the complex interval arithmetic is completely implemented in the modules `cinterval.hpp` and `cinterval.cpp`. In release 2.1 or higher the complex interval standard functions like the exponential or trigonometric functions are additionally implemented in `cimath.hpp` and `cimath.cpp`. For such a function $f(z)$ and a given rectangular complex interval Z, some interval W is computed that contains all function values of f for all $z \in Z$. In nearly all cases the inclusion intervals W are nearly optimal, such that W is almost the smallest rectangular interval containing the range of f. For all complex standard functions the user will find many hints, how to calculate these intervals W and how the implemented multi-valued functions f are defined on their principal branches. In general these definitions are chosen, such that the functions f are *inclusion isotone*. The functions under consideration are root and power functions, the exponential, trigonometric and hyperbolic functions, and their inverse functions.

MSC Subject Classifications: 30-04, 65G30, 65K05.
Key Words: Complex interval functions, reliable computations, mathematical software, C-XSC

1 Introduction

In [27, 19, 20] and [9, 8, 10] algorithms for calculating the complex standard functions are published together with guaranteed error bounds for the real and imaginary part of the function values, evaluated in floating-point arithmetic. Computing such a priori error bounds is a very demanding task. Because of the great effort associated with this method, M. Neher [25] used a more simple way, which in general will enlarge the runtime of the algorithms and which is briefly described as follows:

With $z = x + i \cdot y,\ i = \sqrt{-1}$ the function $f(z) = u(x, y) + i \cdot v(x, y)$ under consideration is assumed to be analytic on the given interval Z, so that the extremal values of $u(x, y)$ and $v(x, y)$ lie on the boundary of Z. In general these extremal points are the corner points of Z, but in some cases the extremal points lie somewhere on the boundary and are possibly not representable in the IEEE-System. In these cases the extremal points must be enclosed by small machine intervals or otherwise by appropriate point intervals. If for example the real intervals enclosing the maximum point of the real part $u(x, y)$ are denoted by[1] $M_x = [x_1, x_2]$ and $M_y = [y_1, y_2]$, an upper bound of the maximum of $u(x, y)$ can be found by evaluating $u(x, y)$ with interval arguments, substituting x and y by M_x, M_y respectively. If then U is a machine inclusion of $u(M_x, M_y) \subseteq U$, a guaranteed upper bound of the maximum of the real part is given by $\mathsf{Sup}(U) \geq u(x, y)\, \forall z \in Z$. In this way the inclusion of the real and imaginary part of $f(z)$ with $z \in Z$ can be computed without the need of a priori error bounds (as it has been done in [10],[20]).

The advantage of this strategy is the simple interval evaluation of the two functions $u(x, y)$ and $v(x, y)$ composed in most cases of the elementary interval functions already implemented in C-XSC [18, 17]. The disadvantage is the rather expensive interval evaluation of $u(x, y)$ and $v(x, y)$ described above.

[1]At least one of the intervals M_x, M_y is a point interval because the extremal points lie on the boundary of the machine interval Z.

Some simple expressions like $\sqrt{x^2+y^2}$ are evaluated in the CoStLy library [25] in naive interval arithmetic by using the instruction `U = sqrt(sqr(x)+sqr(y))`, so for sufficient great values of x or y an overflow occurs and leads to a significant overestimation of the inclusions. In this example the overestimation can be avoided by using the C-XSC function call `U = sqrtx2y2(x,y)`. In this function a premature overflow doesn't occur. In the same manner the following expressions are treated:
$\ln(x^2+y^2), \sqrt{1+x^2}, \sqrt{1-x^2}, \sqrt{x^2-1}, \arctan(y/x), \sqrt{1+x}-1, \ln(1+x), \text{arcosh}(1+x)$.
The last three functions allow an evaluation without overestimations for $x \to 0$ and the first function is used for $x^2+y^2 \to +\infty$ or $x^2+y^2 \to +1$. The appropriate improvements in the algorithms are not described here, more details can be found in [4].

The implementation of the complex interval functions distributed with C-XSC are essentially based on the source code of the CoStLy library developed by M. Neher in 2004 [25].

2 Notation and Preliminary Remarks

2.1 Real Intervals

- $X = [\underline{x}, \overline{x}] := \{r \in \mathbb{R} \,|\, \underline{x} \le r \le \overline{x}]\}$. Intervals are denoted by capital letters.
- The set of all compact real intervals is denoted by $\mathbb{IR}$.
- $\text{diam}(X) := \overline{x} - \underline{x}$ is called the *diameter* of X.
- $\text{sqr}(X) := \{r^2 \,|\, r \in X\}$
- $\text{abs}(X) := \{|r| \,\big|\, r \in X\}$

2.2 Complex Numbers, Intervals and Sets

- Complex numbers are denoted by $z = x + iy$, where $x = \Re\{z\}$ and $y = \Im\{z\}$. The set of all complex numbers is denoted by $\mathbb{C}$.
- $\text{abs}(z) = |z| := \sqrt{x^2+y^2}$ is the absolute value of z.
- A rectangular complex interval Z is defined by $Z := X + i \cdot Y$, with $X = [\underline{x}, \overline{x}]$ and $Y = [\underline{y}, \overline{y}]$.
- The set of all complex rectangular intervals is denoted by $\mathbb{IC}$.
- $\text{abs}(Z) := \{r \in \mathbb{R} \,\big|\, r = \text{abs}(z) \wedge z \in Z\}$.
- A complex analytic function can be written in the form

  $$f(z) = u(x,y) + i \cdot v(x,y),$$

  where $u(x,y) = \Re\{f(z)\}$ and $v(x,y) = \Im\{f(z)\}$.

- The set $\mathbb{C}-(-\infty, 0]$, the complex plane with the negative real axis and the origin removed, is denoted by $\mathbb{C}^-$.
- The set $\mathbb{C} - (-\infty, 0)$ is denoted by $\mathbb{C}_0^-$ containing the origin.
- For a given bounded subset M of $\mathbb{C}$ the *intervall hull* $\square M$ of M is the smallest rectangular interval that contains M.
- $z = |z| \cdot \{\cos(\varphi) + i \cdot \sin(\varphi)\}$. Due to the periodicity of the sine and the cosine functions this polar form of z is not unique. But for $z \in \mathbb{C}^-$ together with the restriction $-\pi < \varphi < +\pi$ the argument function $\varphi = \mathrm{Arg}(z)$ is unique. For complex intervals Z the appropriate argument function(s) are defined separately.

2.3 Definition and Features of Inclusion Functions

- For $D \subseteq \mathbb{C}$, the range $\{f(z) \mid z \in D\}$ of a function $f : D \to \mathbb{C}$ is denoted by $\mathrm{Rg}(f, D)$.
- An *inclusion function* F of a given function $f : \mathbb{C} \to \mathbb{C}$ is an interval function $F : \mathbb{IC} \to \mathbb{IC}$ that encloses the range $\mathrm{Rg}(f, Z)$ of f on all intervals $Z \subseteq D$: $\mathrm{Rg}(f, Z) \subseteq F(Z)\ \forall\ Z \subseteq D$, however in general we have $\mathrm{Rg}(f, Z) \subset F(Z)$ and not $F(Z) = \mathrm{Rg}(f, Z)$; see the example on page 43.
- If $F(Z) = \square\mathrm{Rg}(f, Z)$ holds for all $Z \subseteq D$, then F is called an *optimal* inclusion function.
- F is called *inclusion isotone*, if $Z_1 \subseteq Z_2 \implies F(Z_1) \subseteq F(Z_2)$ holds for all $Z_1, Z_2 \subseteq D$. This important feature should in general be realized for all inclusion functions, if they are used in self-validated algorithms.

2.4 Notation of Inclusion Functions

The notation of the complex inclusion functions is identical with the names of the elementary functions in C-XSC. If different definitions for the inclusion of a specific elementary function are provided, then of course different notations are necessary. The following set of elementary functions is considered:

$$S_f := \{\ \mathrm{exp}, \mathrm{ln}, \mathrm{arg}, \mathrm{sqr}, \mathrm{sqrt}, \mathrm{power}, \mathrm{pow}, \mathrm{cos}, \mathrm{sin}, \mathrm{tan}, \mathrm{cot}, \mathrm{cosh}, \mathrm{sinh}, \\ \mathrm{tanh}, \mathrm{coth}, \mathrm{acos}, \mathrm{asin}, \mathrm{atan}, \mathrm{acot}, \mathrm{acosh}, \mathrm{asinh}, \mathrm{atanh}, \mathrm{acoth}\ \}$$

The notation in S_f is the notation of the elementary functions in C-XSC. Using such a C-XSC function is indicated with the `typewriter` font, so the inclusion of $e^{\sqrt{2}}$ can be written in the form:

$$e^{\sqrt{2}} \in \texttt{exp(sqrt(interval(2)));}$$

The program input of an interval $Z \in \mathbb{IC}$ of type `cinterval` can be done by

(1) $$Z = X + i \cdot Y = (\,[\underline{x}, \overline{x}], [\underline{y}, \overline{y}]\,)$$

using the right-hand side in (1). Then with $Z = ([2,2],[1,1]) \ni z = 2+i$ the inclusion of e^{2+i} can be written in the form:

$$e^{2+i} \in \texttt{W = exp(Z)}, \quad \texttt{W,Z} \text{ of type } \texttt{cinterval},$$

where the exponential function in `cimath.hpp` is declared as follows:

```
cinterval exp(const cinterval& Z);
```

The available inclusion functions for complex interval arguments Z are listed in the following Table:

Table 1: Complex inclusion functions in C-XSC

interval function	**Definition of the appropriate inclusion function declared in cimath.hpp**
$\exp(Z)$	`cinterval exp(const cinterval& Z);`
$\ln(Z)$	`cinterval Ln(const cinterval& Z);`
	`cinterval ln(const cinterval& Z);`
$\mathrm{Arg}(Z)$	`interval Arg(const cinterval& Z);`
$\arg(Z)$	`interval arg(const cinterval& Z);`
Z^2	`cinterval sqr(const cinterval& Z);`
$\sqrt{Z}$	`cinterval sqrt(const cinterval& Z);`
	`std::list<cinterval>sqrt_all(const cinterval& Z);`
$\sqrt[n]{Z}$	`cinterval sqrt(const cinterval& Z, int n);`
	`std::list<cinterval>sqrt_all(const cinterval& Z,int n);`
Z^n	`cinterval power_fast(const cinterval& Z, int n);`
	`cinterval power(const cinterval& Z, int n);`
Z^p	`cinterval pow(const cinterval& Z, const interval p);`
	`cinterval pow(const cinterval& Z, const cinterval p);`
$\cos(Z)$	`cinterval cos(const cinterval& Z);`
$\sin(Z)$	`cinterval sin(const cinterval& Z);`
$\tan(Z)$	`cinterval tan(const cinterval& Z);`
$\cot(Z)$	`cinterval cot(const cinterval& Z);`
$\cosh(Z)$	`cinterval cosh(const cinterval& Z);`

Table 1: (Continuation)

$\sinh(Z)$	`cinterval sinh(const cinterval& Z);`
$\tanh(Z)$	`cinterval tanh(const cinterval& Z);`
$\coth(Z)$	`cinterval coth(const cinterval& Z);`
acos(Z)	`cinterval acos(const cinterval& Z);`
asin(Z)	`cinterval asin(const cinterval& Z);`
atan(Z)	`cinterval atan(const cinterval& Z);`
acot(Z)	`cinterval acot(const cinterval& Z);`
acosh(Z)	`cinterval acosh(const cinterval& Z);`
asinh(Z)	`cinterval asinh(const cinterval& Z);`
atanh(Z)	`cinterval atanh(const cinterval& Z);`
acoth(Z)	`cinterval acoth(const cinterval& Z);`

For `power_fast(Z,n), power(Z,n), sqrt(Z,n)` it holds $n \in \mathbb{Z} = \{0, \pm 1, \ldots\}$.
In `sqrt_all(Z,n)` n is restricted to $n \in \mathbb{N}_0 = \{0, 1, 2, \ldots\}$.

2.5 Design of Complex Inclusion Functions

The rigorous definition of an inclusion function often conflicts with the different requirements of software users. For example, if analyticity is imposed on the square root function, then its maximal domain is a slit complex plane. The plane is usually slit at the negative real axis and the domain will be $\mathbb{C}^-$. Then $\sqrt{-1}$ is undefined and a function call with this argument should terminate program execution. On the other hand, if analyticity is dropped, there are several possibilities to define an inclusion function:

1. Only $+i$ is included
2. Only $-i$ is included
3. Both values $+i, -i$ are included: $\quad \sqrt{-1} \in W = ([0,0], [-1,+1])$.
4. A list of the two inclusions for $+i$ and $-i$ is generated;

In C-XSC the implementation of the square root is realized by 1., where the domain is $\mathbb{C}_0^-$ and the function values on the branch cut are defined coming from quadrant II, this implies $\sqrt{-1} \in W = ([0,0], [+1,+1])$ and many users will expect this result. The function `sqrt_all(Z)` is implemented by 4., and with $Z = ([-1,-1], [0,0])$ a list of inclusions for $+i$ and $-i$ is calculated. This simple example shows that the definition of each inclusion function must be described in all details.

All definitions of the complex-valued inclusion functions must be chosen in a way that the feature of *inclusion isotonicity* is guaranteed, because in self-validated algorithms this feature is mostly involved as a crucial point.

3 Inclusion Functions for the Complex Standard Functions and their Implementation in C-XSC

Complex functions fall into two categories: single-valued functions, such as the exponential function e^z, and multi-valued functions, like roots or logarithms.

3.1 Single-valued Functions

The exponential function, the trigonometric and hyperbolic functions and the power function for integer exponents are single-valued. Some of these functions are also separable, which simplifies the construction of an inclusion function significantly.

3.1.1 Separable Functions $e^z, \sin(z), \cos(z), \sinh(z), \cosh(z)$

A complex function $f(z) = u(x, y)+i\cdot v(x, y)$ is called separable, if both $u(x, y)$ and $v(x, y)$ can be written as products of two univariate real functions. The following functions are separable:

$$\begin{aligned}
e^z &= e^x \cdot \cos(y) + i \cdot e^x \cdot \sin(y),\\
\sin(z) &= \sin(x) \cdot \cosh(y) + i \cdot \cos(x) \cdot \sinh(y),\\
\cos(z) &= \cos(x) \cdot \cosh(y) - i \cdot \sin(x) \cdot \sinh(y),\\
\sinh(z) &= \sinh(x) \cdot \cos(y) + i \cdot \cosh(x) \cdot \sin(y),\\
\cosh(z) &= \cosh(x) \cdot \cos(y) + i \cdot \sinh(x) \cdot \sin(y).
\end{aligned}$$

For a given complex interval $Z = X + i \cdot Y$ the optimal inclusion function is found by substituting x, y on the above right hand side by the real intervals X, Y, respectively and then by evaluating the appropriate interval terms using the standard interval functions implemented in C-XSC.
With the complex-valued input interval

```
Z = ([3,3],[-6,-6])
```

the sample program

```
// Sample program for complex-valued inclusion functions

       #include <iostream>   // for cout
       #include "cimath.hpp" // for exp(...)

       using namespace cxsc;
       using namespace std;
```

```
int main()
{
    cinterval Z;
    cout << "Z = ? ";  cin >> Z;
    cout << SetPrecision(15,15)
         << Scientific << "exp(Z) = " << exp(Z) << endl;
}
```

delivers the following output

```
exp(Z) = ([1.928553574506357E+001,1.928553574506368E+001],
          [5.612210305985390E+000,5.612210305985420E+000])
```

and with `exp(Z)` we have a guaranteed inclusion of $e^{3-6i} \in$ `exp(Z)`.
With `Z = ([3,4],[-6,-6])` the following inclusion is calculated:

```
exp(Z) = ([1.928553574506357E+001,5.242352136790396E+001],
          [5.612210305985390E+000,1.525556929225075E+001])
```

3.1.2 The Square Function z^2

The optimal inclusion function for z^2 is given by

$$Z^2 := (X - Y) \cdot (X + Y) + i \cdot (2 \cdot X \cdot Y).$$

After substituting X, Y by abs(X), abs(Y) respectively, the above real part $(X - Y) \cdot (X + Y)$ has to be evaluated on the appropriate extremal points in quadrant I, [4]. A likewise optimal inclusion for the imaginary part is given by the simple interval product $2 \cdot X \cdot Y$. Replacing `exp(Z)` by `sqr(Z)` in the sample program on page 11, then with `Z = ([0,0],[1,1])` we get:

$$(0+i)^2 = (-1) \in \big(\; \begin{array}{l}[-1.000000000000000, -1.000000000000000], \\ [+0.000000000000000, +0.000000000000000] \; \big) \end{array}$$

and for `Z = ([2,3],[1,1])` the program output is given by:

```
sqr(Z) = ([3.000000000000000E+000,8.000000000000000E+000],
          [4.000000000000000E+000,6.000000000000000E+000])
```

3.1.3 Power Function for Integer Exponents z^n

The power function z^n is single-valued for $n \in \mathbb{Z}$. If $n < 0$, then $0 \notin Z$ must always be realized. For $n = -1, 0, 1, 2$, the inclusion functions for Z^n are implemented as $1/Z, 1, Z, Z^2$,respectively. For $n \leq -2$ or $n \geq +3$, see [25].
There are two different inclusion functions:

- `power_all(Z,n)` uses $z^n = |z|^n \cdot \{\cos(n\varphi) + i \cdot \sin(n\varphi)\}$, where $|z|^n$ and $\cos(n\varphi), \sin(n\varphi)$ are evaluated in a naive interval arithmetic, which leads to some overestimations, if Z is not a point interval. Significant overestimations are produced, if the angle arg(z) for $z \in Z$ intersects the cartesian axes more than once. The advantage of this function is the much cheaper calculation of the inclusion intervals in comparison to the following function.
- `power(Z,n)` calculates even for thick intervals a nearly optimal inclusion for all Values z^n, with $z \in Z \in \mathbb{IC}$.

For the complex point interval $Z = ([1,1],[1,1])$ we get:

```
power_fast(Z,4) = ([-4.000000000000015E+00,-3.999999999999987E+00],
                   [-2.793185071074522E-14,+2.358249763186233E-14])
power(Z,4)      = ([-4.000000000000015E+00,-3.999999999999987E+00],
                   [-2.793185071074522E-14,+2.358249763186233E-14])
```

For the complex interval $Z = ([1,1.0001],[1,1])$ we get:

```
power_fast(Z,4) = ([-4.000800080004018E+00,-3.999999920007986E+00],
                   [-2.793743763955234E-14,+8.001200040244581E-04])
power(Z,4)      = ([-4.000799999996026E+00,-3.999999999999987E+00],
                   [-2.793185071074522E-14,+8.001200040244581E-04])
```

For the complex interval $Z = ([1,1.125],[1,1])$ we get:

```
power_fast(Z,4) = ([-5.133056640625019E+00,-3.890035671819244E+00],
                   [-3.584394294393423E-14,+1.195312500000034E+00])
power(Z,4)      = ([-4.991943359375036E+00,-3.999999999999987E+00],
                   [-2.793185071074522E-14,+1.195312500000034E+00])
```

For the complex interval $Z = ([1,1.125],[1,1.25])$ we get:

```
power_fast(Z,4) = ([-7.998291015625031E+00,-3.614515169541913E+00],
                   [-3.425799840125003E+00,+1.862527125445950E+00])
power(Z,4)      = ([-7.822021484375056E+00,-3.999999999999987E+00],
                   [-2.812500000000063E+00,+1.195312500000034E+00])
```

For the complex interval $Z = ([1,1.125],[1,1.25])$ we get:

```
power_fast(Z,8) = ([+1.012943982169734E+01,+6.397265917062802E+01],
                   [-2.897499678121650E+01,+4.951984112013258E+01])
power(Z,8)      = ([+1.599999999999992E+01,+5.839538103342139E+01],
                   [-1.193386459350621E+01,+3.337646484375070E+01])
```

3.1.4 The Functions $\tan(z), \cot(z), \tanh(z), \coth(z)$

The real and imaginary parts of the tangent are defined by

$$\tan(z) = u(x,y) + i \cdot v(x,y) = \frac{\sin(2x)}{\cos(2x) + \cosh(2y)} + i \cdot \frac{\sinh(2y)}{\cos(2x) + \cosh(2y)}$$

If one of the poles $z_k = \pi(k + 1/2),\ k \in \mathbb{Z}$ lie in a given complex interval Z, then program execution is terminated. Replacing `exp(Z)` by `tan(Z)` in the sample program on page 11, then with `Z = ([1,1],[3,3])` we get:

$$\begin{aligned} \tan(1+3i) \in (\ & [4.517137276658401E-003, 4.517137276658444E-003], \\ & [1.002054988245806E+000, 1.002054988245816E+000]\ ) \end{aligned}$$

and for `Z = ([1,1.5],[3,3.125])` the program output is given by:

```
   tan(Z) = ([5.469399204418454E-004,4.517137276658444E-003],
             [1.001601819138448E+000,1.004919718908787E+000])
```

The cotangent can be defined as

$$\cot(z) = \tan(\frac{\pi}{2} - z),$$

where the poles are given by $z_{p,k} := k\pi,\ k \in \mathbb{Z}$. If such a pole lie in a given complex interval Z, then program execution is terminated. Near the zeros $z_{s,k} := \pi(k+1/2),\ k \in \mathbb{Z}$, tight inclusion intervals can only be calculated for real poin-intervals $X = \Re\{Z\}$ and $|k| < 5$.

The hyperbolic tangent is defined by

$$\tanh(z) := \frac{e^z - e^{-z}}{e^z + e^{-z}} = -i \cdot \tan(iz).$$

If one of the poles $z_k = i \cdot \pi(k + 1/2),\ k \in \mathbb{Z}$ lie in a given complex interval Z, then program execution is terminated.

The hyperbolic cotangent is defined by

$$\coth(z) := \frac{e^z + e^{-z}}{e^z - e^{-z}} = i \cdot \cot(iz).$$

If one of the poles $z_k = i \cdot k\pi,\ k \in \mathbb{Z}$ lie in a given complex interval Z, then program execution is terminated.

3.2 Multi-valued Functions

Several multi-valued functions are implemented in C-XSC: the argument function, the natural logarithm, root and power functions, the inverse trigonometric and inverse hyperbolic functions. For each of these functions, there is a C-XSC procedure that implements the principal branch on its analyticity domain. Where suitable, additional procedures for function evaluations on larger domains have been implemented.

3.2.1 Argument Functions

The polar representation of a complex number $z = x + i \cdot y$ is given by

(2) $$x = r \cdot \cos(\varphi), \qquad y = r \cdot \sin(\varphi), \qquad r = |z| = \sqrt{x^2 + y^2},$$

where r is the absolute value of z and $\varphi = \arg(z)$ is the *argument* of z. Due to the periodicity of the sine and cosine functions, the argument φ of z is not uniquely defined, but with $z \in \mathbb{C}^-$, then the principal value of the argument function is the unique polar angle $\varphi = \text{Arg}(z) \in (-\pi, +\pi)$ that fulfills (2). Notice that with this definition $\varphi = \text{Arg}(z)$ is not defined on the branch cut including the origin.

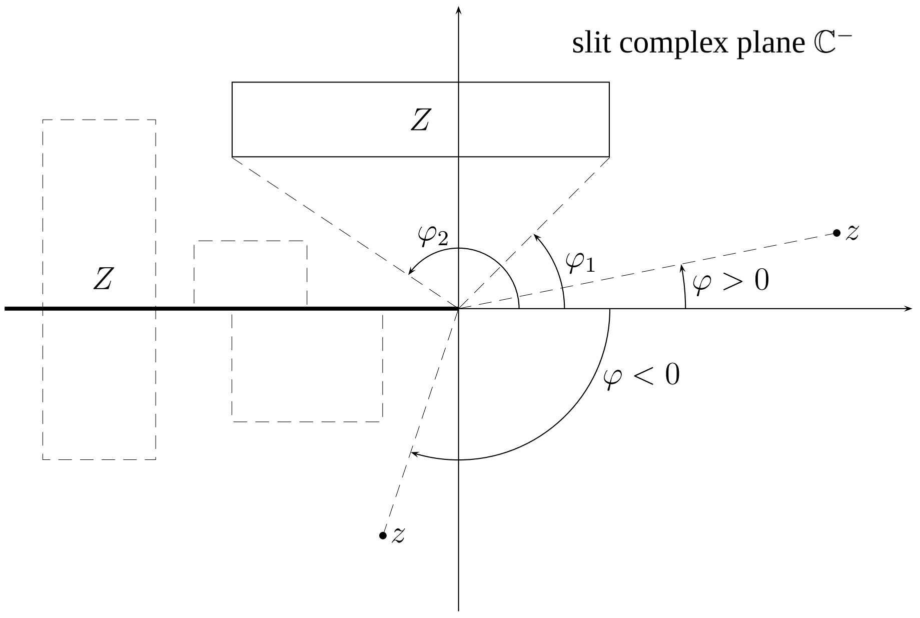

Figure 1: Arg$(Z) = [\varphi_1, \varphi_2]$. The dashed intervals Z are not allowed.

Let us now consider rectangular complex Intervals $Z \subset \mathbb{C}_0^-$ that may contain the origin, but do not intersect the negative real axis. For those intervals the interval argument function Arg(Z) is defined as

(3) $$\text{Arg}(Z) := \begin{cases} \text{undefined}, & Z \cap (-\infty, 0) \neq \emptyset, \\ 0, & Z = 0, \\ \Box\{\text{Arg}(z) \mid z \in Z \wedge z \neq 0\}, & \text{otherwise.} \end{cases}$$

Because of $Z \cap (-\infty, 0) \neq \emptyset$ the dashed intervals Z in Figure 1 lie not in the domain $\mathbb{C}_0^-$ of Arg(Z). $\Box\{....\}$ denotes the interval hull of the set $\{....\}$, and so in Figure 1 it holds: Arg$(Z) = \Box\{\text{Arg}(z) \mid z \in Z \wedge z \neq 0\} = [\varphi_1, \varphi_2]$.

The exact intervals Arg(Z) are included by the C-XSC function

```
interval Arg(const cinterval& Z);
```

$Z = X + i \cdot Y$	$\mathrm{Arg}(Z)$	$\mathtt{Arg}(Z)$
$([0,0],[0,0])$	$[0,0]$	$[0,0]$
$([0,0],[2^{-1074},+1])$	$[\frac{\pi}{2},\frac{\pi}{2}]$	$<\frac{\pi}{2}>$
$([0,0],[0,+1])$	$[\frac{\pi}{2},\frac{\pi}{2}]$	$<\frac{\pi}{2}>$
$([0,0],[-1,-2^{-1074}])$	$[-\frac{\pi}{2},-\frac{\pi}{2}]$	$<-\frac{\pi}{2}>$
$([0,0],[-1,0])$	$[-\frac{\pi}{2},-\frac{\pi}{2}]$	$<-\frac{\pi}{2}>$
$([0,0],[-1,+1])$	$[-\frac{\pi}{2},+\frac{\pi}{2}]$	$<-\frac{\pi}{2},+\frac{\pi}{2}>$
$([0,+1],[-1,+1])$	$[-\frac{\pi}{2},+\frac{\pi}{2}]$	$<-\frac{\pi}{2},+\frac{\pi}{2}>$
$([0,+1],[0,+1])$	$[0,\frac{\pi}{2}]$	$<0,\frac{\pi}{2}>$
$([0,+1],[-1,0])$	$[-\frac{\pi}{2},0]$	$<-\frac{\pi}{2},0>$
$([-1,+1],[-1,+1])$	undefined	abnormal termination
$([-1,0],[0,+1])$	undefined	abnormal termination
$([-1,0],[-1,0])$	undefined	abnormal termination
$([-1,0],[-1,+1])$	undefined	abnormal termination
$([-2,-1],[-1,+1])$	undefined	abnormal termination
$([-2,-1],[-1,0])$	undefined	abnormal termination
$([-2,-1],[0,+1])$	undefined	abnormal termination

Table 2: Inclusion results for $\mathrm{Arg}(Z) \subseteq \mathtt{Arg}(Z)$

and for some machine intervals $Z \subset \mathbb{C}_0^-$ the inclusion results are listed in Table 2:

In Table 2 $<0,\frac{\pi}{2}>$ denotes the optimal machine interval including $[0,\frac{\pi}{2}] \subset <0,\frac{\pi}{2}>$. The inclusion intervals of the last column can be calculated with the program on page 11 substituting `Arg(Z)` for `exp(Z)`.

Some remarks concerning the **inclusion isotonicity** of $\mathrm{Arg}(Z)$:
If $0 \neq Z_1 \subseteq Z_2 \subseteq \mathbb{C}_0^-$, then $\mathrm{Arg}(Z_1) \subseteq \mathrm{Arg}(Z_2)$, so that inclusion isotonicity is fulfilled for $Z_1 \neq 0$. However, if $Z_1 = 0$ and for example $Z_2 = 0 + i[0,1]$, then $Z_1 \subseteq Z_2$ holds, but $\mathrm{Arg}(Z_1) = [0,0] \not\subseteq [\frac{\pi}{2},\frac{\pi}{2}] = \mathrm{Arg}(Z_2)$, so inclusion isotonicity is not realized in this case.

A frequent use of the argument function is the inclusion of a rectangular interval $Z = X + i \cdot Y$ into a polar interval $(R,\Phi) = ([r_1,r_2],[\varphi_1,\varphi_2])$ such that

(4) $$Z \subseteq (R,\Phi)$$

holds. In Figure 2 the polar interval (R,Φ) is plotted in bold face.
The optimal inclusion (inclusion with minimal overestimation) is

(5) $$R := \mathrm{abs}(Z), \quad \Phi := \mathrm{Arg}(Z),$$

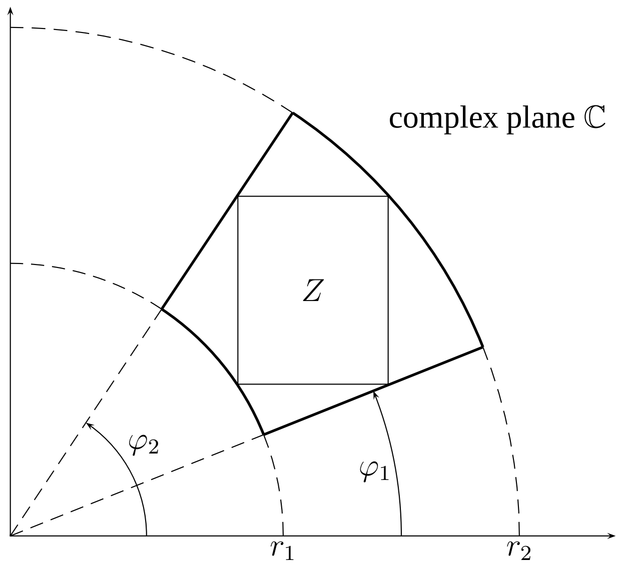

Figure 2: Inclusion of Z into the polar interval (R, Φ).

provided that Arg(Z) is defined. This implies that the interval Z must not intersect the negative real axis, even though the inclusion into a polar interval is well defined for any $Z \in \mathbb{IC}$. To remove this restriction on Z, we develop an argument function arg(Z) that will be used in (5) instead of Arg(Z).
For minimal overestimation in (4), the following properties are required:

(i) $Z \subseteq \big(\text{abs}(Z), \text{arg}(Z)\big)$ for all $Z \in \mathbb{IC}$

(ii) for every $Z \in \mathbb{C}$, there is no interval Φ such that
$Z \subseteq \big(\text{abs}(Z), \Phi\big)$ and $\text{diam}(\Phi) < \text{diam}(\text{arg}(Z))$.

With $Z = [\underline{x}, \overline{x}] + i \cdot [\underline{y}, \overline{y}]$ and

$$\widetilde{\text{arg}}(z) := \begin{cases} \text{Arg}(z), & \Re\{z\} \le 0,\ \Im\{z\} > 0, \quad \text{see page 14,} \\ \pi, & \Re\{z\} \le 0,\ \Im\{z\} = 0, \\ \text{Arg}(z) + 2\pi, & \Re\{z\} \le 0,\ \Im\{z\} < 0. \end{cases}$$

(i) and (ii) are fulfilled by the following argument function

$$\text{(6)} \qquad \text{arg}(Z) := \begin{cases} [0,0], & Z = 0 \\ [\pi,\pi], & \underline{x} < 0,\ \overline{x} \le 0,\ \underline{y} = \overline{y} = 0 \\ [0,\pi], & \underline{x} < 0,\ \overline{x} < 0,\ \underline{y} = \overline{y} = 0 \\ \square\{\widetilde{\text{arg}}(z) \mid z \in Z\}, & \underline{x} < 0,\ \overline{x} \le 0,\ \underline{y} < 0 < \overline{y} \\ \square\{\text{Arg}(z) \mid z \in Z \cap \mathbb{C}^-\}, & \text{otherwise.} \end{cases}$$

arg(Z) has the following properties:

- arg(Z) is defined for all $Z \in \mathbb{IC}$.
- arg$(Z) =$ Arg(Z), if Arg(Z) is defined.
- arg$(Z) \subseteq [-\pi, \frac{3\pi}{2}]$, diam(arg$(Z)$) $\leq 2\pi$.
- arg(Z) is not inclusion isotone if Z intersects the negative real axis. For example, if

  $$Z_1 = [-2,-1] + i \cdot [-1,0], \quad Z_2 = [-2,-1] + i \cdot [-1,+1],$$

  then we have

  $$\text{arg}(Z_1) = [-\pi, -\frac{3\pi}{4}], \quad \text{arg}(Z_2) = [\frac{3\pi}{4}, \frac{5\pi}{4}].$$

  Even though $Z_1 \subset Z_2$ holds, the computed function intervals have an empty intersection.
- The optimal inclusion of a rectangular interval Z into a polar interval is given by

  $$R := \text{abs}(Z), \quad \Phi := \text{arg}(Z).$$

The exact intervals arg(Z) are included by the C-XSC function

```
interval arg(const cinterval& Z);
```

For some machine intervals $Z \in \mathbb{IC}$ the inclusion results are listed in Table 3:

The inclusion intervals of the last column can be calculated with the program on page 11 substituting `arg(Z)` for `exp(Z)`.

With $Z_2 = [-2,-1] + i \cdot [-1,+1]$ it holds:

$$R := \text{abs}(Z_2) = [1, \sqrt{5}], \qquad \Phi := \text{arg}(Z_2) = [\frac{3\pi}{4}, \frac{5\pi}{4}]$$

and with (R, Φ) we have the optimal polar interval including the rectangular complex interval Z_2. An inclusion of the interval R can be calculated with the C-XSC function `interval abs(const cinterval& Z);` declared in `cinterval.hpp`.
The following function is defined for all rectangular complex intervals $Z \in \mathbb{IC}$:

$$\text{arg\_inclmon}(Z) := \begin{cases} [-\pi, +\pi], & \underline{x} < 0, \quad \underline{y} \leq 0 \leq \overline{y} \\ \text{Arg}(Z), & \text{otherwise.} \end{cases}$$

The exact intervals arg_inclmon(Z) are included by the C-XSC function

```
interval arg_inclmon(const cinterval& Z);
```

For some machine intervals $Z \in \mathbb{IC}$ the inclusion results are listed in Table 4:
The inclusion intervals of the last column can be calculated with the program on page 11 substituting `arg_inclmon(Z)` for `exp(Z)`.
Please notice that `arg_inclmon`(Z) is inclusion isotone. For example, if again we use

$$Z_1 = [-2,-1] + i \cdot [-1,0], \quad Z_2 = [-2,-1] + i \cdot [-1,+1],$$

$Z = X + i \cdot Y$	$\text{arg}(Z)$	$\texttt{arg}(Z)$
$([0,0],[0,0])$	$[0,0]$	$[0,0]$
$([0,1],[0,0])$	$[\pi,\pi]$	$<\pi>$
$([-1,0],[0,0])$	$[0,0]$	$[0,0]$
$([0,0],[2^{-1074},+1])$	$[\frac{\pi}{2},\frac{\pi}{2}]$	$<\frac{\pi}{2}>$
$([0,0],[0,+1])$	$[\frac{\pi}{2},\frac{\pi}{2}]$	$<\frac{\pi}{2}>$
$([0,0],[-1,-2^{-1074}])$	$[-\frac{\pi}{2},-\frac{\pi}{2}]$	$<-\frac{\pi}{2}>$
$([0,0],[-1,0])$	$[-\frac{\pi}{2},-\frac{\pi}{2}]$	$<-\frac{\pi}{2}>$
$([0,0],[-1,+1])$	$[-\frac{\pi}{2},+\frac{\pi}{2}]$	$<-\frac{\pi}{2},+\frac{\pi}{2}>$
$([0,+1],[-1,+1])$	$[-\frac{\pi}{2},+\frac{\pi}{2}]$	$<-\frac{\pi}{2},+\frac{\pi}{2}>$
$([0,+1],[0,+1])$	$[0,\frac{\pi}{2}]$	$<0,\frac{\pi}{2}>$
$([0,+1],[-1,0])$	$[-\frac{\pi}{2},0]$	$<-\frac{\pi}{2},0>$
$([-1,+1],[-1,+1])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-1,0],[0,+1])$	$[\frac{\pi}{2},\pi]$	$<\frac{\pi}{2},\pi>$
$([-1,0],[-1,0])$	$[-\pi,-\frac{\pi}{2}]$	$<-\pi,-\frac{\pi}{2}>$
$([-1,0],[-1,+1])$	$[\frac{\pi}{2},\frac{3\pi}{2}]$	$<\frac{\pi}{2},\frac{3\pi}{2}>$
$([-2,-1],[-1,+1])$	$[\frac{3\pi}{4},\frac{5\pi}{4}]$	$<\frac{3\pi}{4},\frac{5\pi}{4}>$
$([-2,-1],[-1,0])$	$[-\pi,-\frac{3\pi}{4}]$	$<-\pi,-\frac{3\pi}{4}>$
$([-2,-1],[0,+1])$	$[\frac{3\pi}{4},\pi]$	$<\frac{3\pi}{4},\pi>$
$([-1,+1],[0,0])$	$[0,\pi]$	$<0,\pi>$

Table 3: Inclusion results for $\text{arg}(Z) \subseteq \texttt{arg}(Z)$

then for the argument intervals it follows

$$\text{arg\_inclmon}(Z_1) = \text{arg\_inclmon}(Z_2) = [-\pi,+\pi].$$

Remarks:

- The interval function $\text{arg\_inclmon}(Z)$ is not used in any function implemented in `cimath.cpp`.
- The functions $\text{Arg}(Z)$ and $\text{arg}(Z)$ are used for example in the natural logarithm, root and power functions.

$Z = X + i \cdot Y$	arg_inclmon(Z)	`arg_inclmon`(Z)
$([0,0],[0,0])$	$[0,0]$	$[0,0]$
$([0,1],[0,0])$	$[0,0]$	$[0,0]$
$([-1,0],[0,0])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([0,0],[2^{-1074},+1])$	$[\frac{\pi}{2},\frac{\pi}{2}]$	$<\frac{\pi}{2}>$
$([0,0],[0,+1])$	$[\frac{\pi}{2},\frac{\pi}{2}]$	$<\frac{\pi}{2}>$
$([0,0],[-1,-2^{-1074}])$	$[-\frac{\pi}{2},-\frac{\pi}{2}]$	$<-\frac{\pi}{2}>$
$([0,0],[-1,0])$	$[-\frac{\pi}{2},-\frac{\pi}{2}]$	$<-\frac{\pi}{2}>$
$([0,0],[-1,+1])$	$[-\frac{\pi}{2},+\frac{\pi}{2}]$	$<-\frac{\pi}{2},+\frac{\pi}{2}>$
$([0,+1],[-1,+1])$	$[-\frac{\pi}{2},+\frac{\pi}{2}]$	$<-\frac{\pi}{2},+\frac{\pi}{2}>$
$([0,+1],[0,+1])$	$[0,\frac{\pi}{2}]$	$<0,\frac{\pi}{2}>$
$([0,+1],[-1,0])$	$[-\frac{\pi}{2},0]$	$<-\frac{\pi}{2},0>$
$([-1,+1],[-1,+1])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-1,0],[0,+1])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-1,0],[-1,0])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-1,0],[-1,+1])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-2,-1],[-1,+1])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-2,-1],[-1,0])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-2,-1],[0,+1])$	$[-\pi,+\pi]$	$<-\pi,+\pi>$
$([-1,+1],[0,0])$	$[-\pi,\pi]$	$<-\pi,+\pi>$

Table 4: Inclusion results for arg_inclmon$(Z) \subseteq$ `arg_inclmon`(Z)

3.2.2 Natural Logarithm

For complex numbers $z = x + i \cdot y \in \mathbb{C}$, with $z \neq 0$, $|z| = r := \sqrt{x^2 + y^2}$ and $z = r \cdot e^{i(\varphi + 2k\pi)}$ the natural logarithm is defined by

$$\ln_k(z) := \ln(r) + i \cdot (\varphi + 2k\pi), \qquad k = 0, \pm 1, \pm 2, \ldots$$

In C-XSC the principal branch of this function is defined for $k = 0$:

$$\ln(z) := \ln(r) + i \cdot \varphi, \qquad -\pi < \varphi \leq +\pi;$$

$\ln(z)$ is analytic for all $z \in \mathbb{C}^-$. If we define $\ln(z)$ on the branch cut[2] by the appropriate limits coming from quadrant II, then $\ln(z)$ is continuous, but not analytic on the branch cut, for

[2]The branch cut is the negative real axis including the origin.

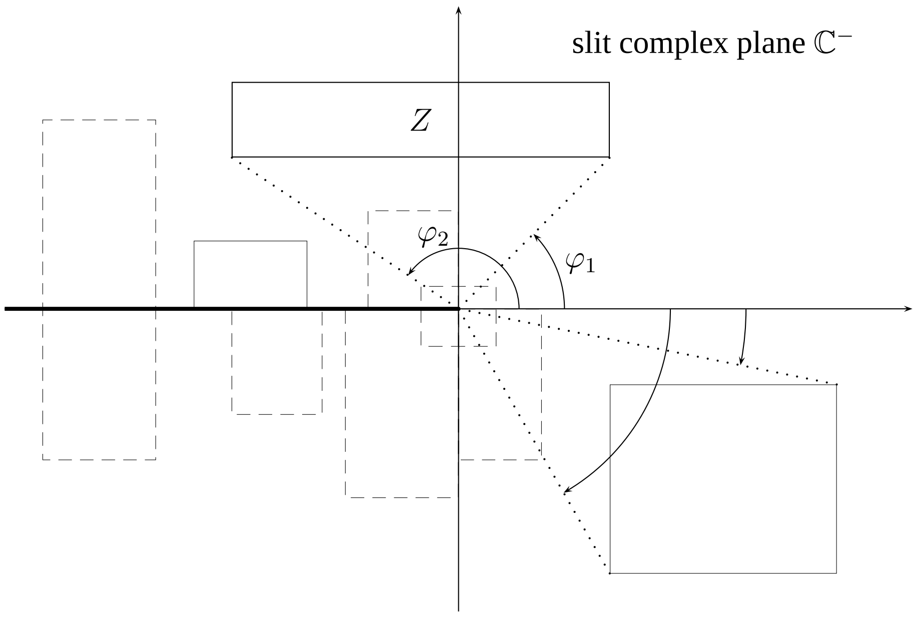

Figure 3: Allowed and not allowed complex arguments Z for Ln(Z).

example it holds: $\ln(-e + i0) = 1 + i\pi$.
For complex intervals $Z = [\underline{x}, \overline{x}] + i \cdot [\underline{y}, \overline{y}]$ together with (6) we define

(7) $$\text{Ln}(Z) := \ln(\,\text{abs}(Z)\,) + i \cdot \text{arg}(Z).$$

In the following cases

- $0 \in Z$ or
- $\underline{x} < 0$ and $\underline{y} < 0 \leq \overline{y}$

the complex interval function Ln(Z) is not defined. With these restrictions Z in quadrant III may not touch the real negative axis, but if Z lies in quadrant II then the branch cut may be touched from above. In Figure 3 allowed and not allowed argument intervals Z are drawn with solid and dashed lines respectively.
For Z in Figure 3 it holds: arg$(Z) = [\varphi_1, \varphi_2]$, with $0 < \varphi_1 < \varphi_2$, and for Z in quadrant III or IV the appropriate angles φ_i are negative, with $\varphi_i > -\pi$.

For some machine intervals $Z \in \mathbb{IC}$ the function values Ln(Z) are listed in Table 5.

The exact complex intervals Ln(Z) can be included by the C-XSC function

```
cinterval Ln(const cinterval& Z);
```

implemented in `cimath.cpp`.

Replacing `exp(Z)` by `Ln(Z)` in the sample program on page 11, then with $Z := [-4, -1] + i \cdot [-1, -2^{-150}]$ the following inclusion is calculated:

$Z = X + i \cdot Y$	$\mathrm{Ln}(Z)$
$([-4,-1],[0,1])$	$([0,\ln(\sqrt{17})],[\frac{3\pi}{4},\pi])$
$([-4,-1],[-1,-2^{-150}])$	$(\frac{1}{2}\cdot[\ln(1+2^{-300}),\ln(17)],[\text{atan}(2^{-152})-\pi,\frac{-3\pi}{4}])$
$([-4,-1],[-1,0])$	not allowed
$([-1,+1],[-2,-1])$	$([0,\frac{1}{2}\cdot\ln(5)],[-\frac{3\pi}{4},-\frac{\pi}{4}])$
$([1,3],[-1,+1])$	$([0,\frac{1}{2}\cdot\ln(10)],[-\frac{\pi}{4},+\frac{\pi}{4}])$
$([-1,+1],[-1,+1])$	not allowed
$([-1,0],[0,+1])$	not allowed

Table 5: Exact function values $\mathrm{Ln}(Z)$ for complex machine arguments Z.

```
Ln(Z) = ([+2.454546732648861E-091,+1.416606672028110E+000],
         [-3.141592653589794E+000,-2.356194490192343E+000])
```

Note: $\mathrm{Ln}(Z) \subset \mathtt{Ln}(Z)$, where the complex interval arguments Z of $\mathrm{Ln}(Z)$ must lie in the capital branch $\mathbb{C}^-$.

An inclusion function for the natural logarithm with argument intervals $Z \not\ni 0$ that may intersect the negative real axis is defined by

$$\ln(Z) := \ln(\,\text{abs}(Z)\,) + i \cdot \text{arg}(Z), \quad 0 \notin Z; \tag{8}$$

In Figure 4 allowed and not allowed argument intervals Z are drawn with solid and dashed lines respectively.

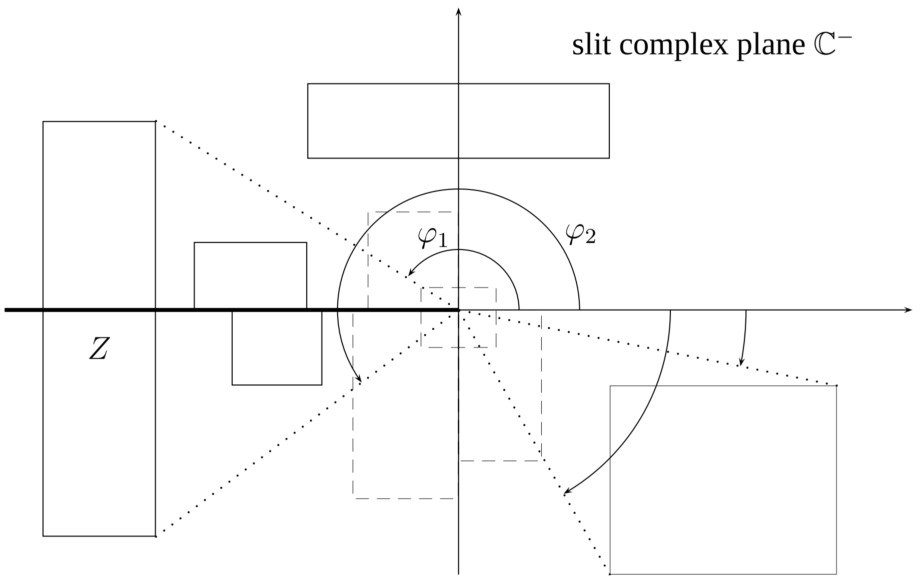

Figure 4: Allowed and not allowed complex arguments Z for $\ln(Z)$.

Please notice that for the above interval Z the function values are continued coming from quadrant II, so that the angles $0 < \varphi_1 < \varphi_2 < 3\pi/2$ are positive.

For some machine intervals $Z \in \mathbb{IC}$, with $0 \notin Z$ the function values $\ln(Z)$ are listed in the following Table 6

$Z = X + i \cdot Y$	$\ln(Z)$
$([-2,-1],[-1,+1])$	$([0, \frac{1}{2} \cdot \ln(5)], [\frac{3\pi}{4}, \frac{5\pi}{4}])$
$([-2,-1],[0,+1])$	$([0, \frac{1}{2} \cdot \ln(5)], [\frac{3\pi}{4}, \pi])$
$([-2,-1],[-1,0])$	$([0, \frac{1}{2} \cdot \ln(5)], [-\pi, -\frac{3\pi}{4}])$
$([-1,+1],[-2,-1])$	$([0, \frac{1}{2} \cdot \ln(5)], [-\frac{3\pi}{4}, -\frac{\pi}{4}])$
$([1,3],[-1,+1])$	$([0, \frac{1}{2} \cdot \ln(10)], [-\frac{\pi}{4}, +\frac{\pi}{4}])$
$([-1,+1],[-1,+1])$	not allowed, $0 \in Z$
$([0,+1],[-1,+1])$	not allowed, $0 \in Z$
$([0,+1],[0,+1])$	not allowed, $0 \in Z$
$([0,+1],[-1,0])$	not allowed, $0 \in Z$
$([-1,0],[-1,+1])$	not allowed, $0 \in Z$
$([-1,0],[0,+1])$	not allowed, $0 \in Z$
$([-1,0],[-1,0])$	not allowed, $0 \in Z$

Table 6: Exact function values $\ln(Z)$ for complex machine arguments Z.

The exact complex intervals $\ln(Z)$ can be included by the C-XSC function

```
cinterval ln(const cinterval& Z);
```

implemented in `cimath.cpp`.

Replacing `exp(Z)` by `ln(Z)` in the sample program on page 11, then with $Z := [-2,-1] + i \cdot [-1,+1]$ the following inclusion is calculated:

```
ln(Z) = ([0.000000000000000E+000,8.047189562170513E-001],
         [2.356194490192343E+000,3.926990816987244E+000])
```

Because of the possible analytic continuation we advice the user of $\ln(Z)$ to carefully check if analyticity is required in his computations, and if so, ensure that it is not violated by switching to different branches of the logarithm in the course of the computation.

3.2.3 Square Root

With $z = x + i \cdot y = |z| \cdot (\cos(\varphi) + i \cdot \sin(\varphi))$ and $|z| = r := \sqrt{x^2 + y^2}$ the principal branch of the square root $\sqrt{z}$ is defined for all $z \in \mathbb{C}^-$ by

(9) $$\sqrt{z} := \sqrt{r} \cdot \{\cos(\varphi/2) + i \cdot \sin(\varphi/2)\}, \quad -\pi < \varphi < +\pi;$$

If additionally we define $\sqrt{z} = 0$ for $z = 0$ and if we define $\sqrt{z}$ on the branch cut by the appropriate limits coming from quadrant II, than

$$(10) \qquad \sqrt{z} := \sqrt{r} \cdot \{\cos(\varphi/2) + i \cdot \sin(\varphi/2)\}, \quad -\pi < \varphi \leq +\pi$$

is analytic in $\mathbb{C}^-$ and continuous on the branch cut realized by the negative real axis including the origin.

We define the interval function sqrt(Z) as follows:

$$(11) \qquad \text{sqrt}(Z) := \square\{\text{sqrt}(z) \mid z \in Z \in \mathbb{IC}\}.$$

However for intervals $Z = [\underline{x}, \overline{x}] + i \cdot [\underline{y}, \overline{y}]$, with $\underline{x} < 0$ and $(\underline{y} < 0$ and $\overline{y} \geq 0)$ sqrt(Z) is not defined. In Figure 5 allowed and not allowed argument intervals Z are drawn with solid and dashed lines respectively.

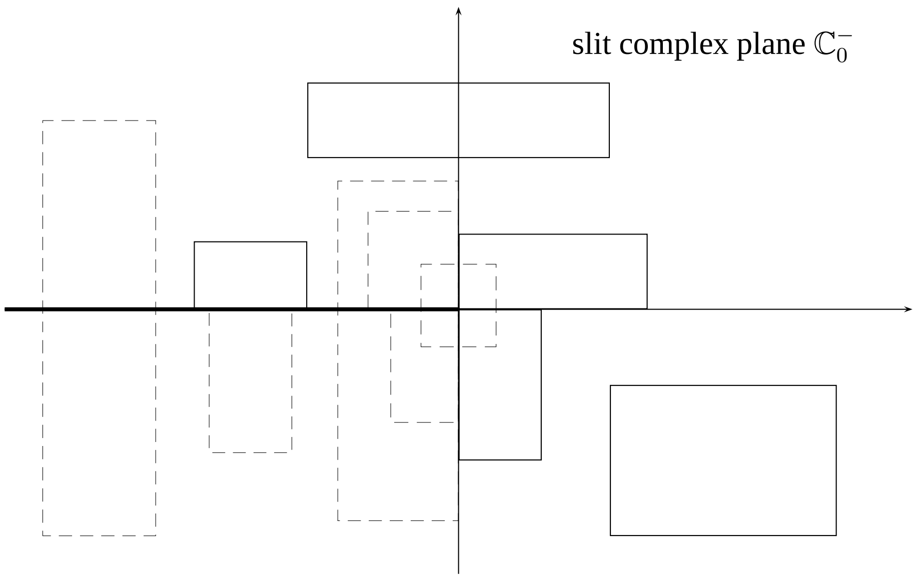

Figure 5: Allowed and not allowed complex arguments Z for sqrt(Z).

The exact complex intervals sqrt(Z) can be included by the C-XSC function

```
cinterval sqrt(const cinterval& Z);
```

implemented in `cimath.cpp`.
Replacing `exp(Z)` by `sqrt(Z)` in the sample program on page 11, then with $Z1 := [-1, 0] + i \cdot [0, 0]$ and $Z2 := [-1, 0] + i \cdot [0, 1]$ the following inclusions are calculated:

```
sqrt(Z1) = ([0.000000000000000E+000,0.000000000000000E+000],
            [0.000000000000000E+000,1.000000000000001E+000])

sqrt(Z2) = ([0.000000000000000E+000,7.071067811865478E-001],
            [0.000000000000000E+000,1.098684113467811E+000])
```

For a given argument interval Z the C-XSC function

```
list<cinterval>sqrt_all(const cinterval& Z)
```

calculates a list of all two inclusion intervals $\pm\text{sqrt}(Z)$. With the complex interval $Z1 := [-1, 0] + i \cdot [0, 0]$ the sample program

```
#include <iostream>    // for cout ...
#include "cimath.hpp"  // for sqrt_all(Z);
#include <list>        // for list<cinterval> a;

using namespace cxsc;
using namespace std;

template<class T>
ostream& operator <<(ostream& os, const list<T>& l) {
    if (l.empty()) return os << "empty";
    for (typename list<T>::const_iterator i=l.begin();
                                i != l.end(); ++i)
    os << *i << " ";
    return os;
}

int main()
{
    list<cinterval> a;
    cinterval Z;
    while (1) {
        cout << "Z = ? ";
        cin >> Z;
        a = sqrt_all(Z); // list of all solutions w_i of w^2=z;
        cout << SetPrecision(15,15)
             << Scientific << "sqrt_all(Z) = " << a << endl;
    }

} // main
```

delivers the output:

```
sqrt_all(Z)=([+0.000000000000000E+000,+7.071067811865478E-001],
             [+0.000000000000000E+000,+1.098684113467811E+000])
            ([-7.071067811865478E-001,-0.000000000000000E+000],
             [-1.098684113467811E+000,-0.000000000000000E+000])
```

3.2.4 n-th Root

According to (9) the principal branch of the n-th root $\sqrt[n]{z}$ is for all $z \in \mathbb{C}^-$ defined by

(12) $$\sqrt[n]{z} := \sqrt[n]{r} \cdot \left\{ \cos(\frac{\varphi}{n}) + i \cdot \sin(\frac{\varphi}{n}) \right\}, \quad -\pi < \varphi < +\pi, \quad n \in \mathbb{N},$$

and with $\sqrt[n]{0} := 0$ the analytic function is additionally continuous in the origin, whereby $\sqrt[n]{z}$ is defined for all $z \in \mathbb{C}_0^-$.

sqrt(Z, n), the inclusion function for the principal branch of the n-th root function, is given by

$$\text{sqrt}(Z, n) := \begin{cases} \Box\{\sqrt[n]{z} \mid z \in Z\}, & Z \subset \mathbb{C}_0^-, \ n \in \mathbb{N}, \\ \text{undefined}, & Z \cap \mathbb{R}^- \neq \emptyset, \ n = 2, 3, \ldots \end{cases}$$

It holds sqrt$(Z, 0) \equiv [1, 1]$ for all $Z \in \mathbb{IC}$. In Figure 6 allowed and not allowed argument intervals Z are drawn with solid and dashed lines, respectively.

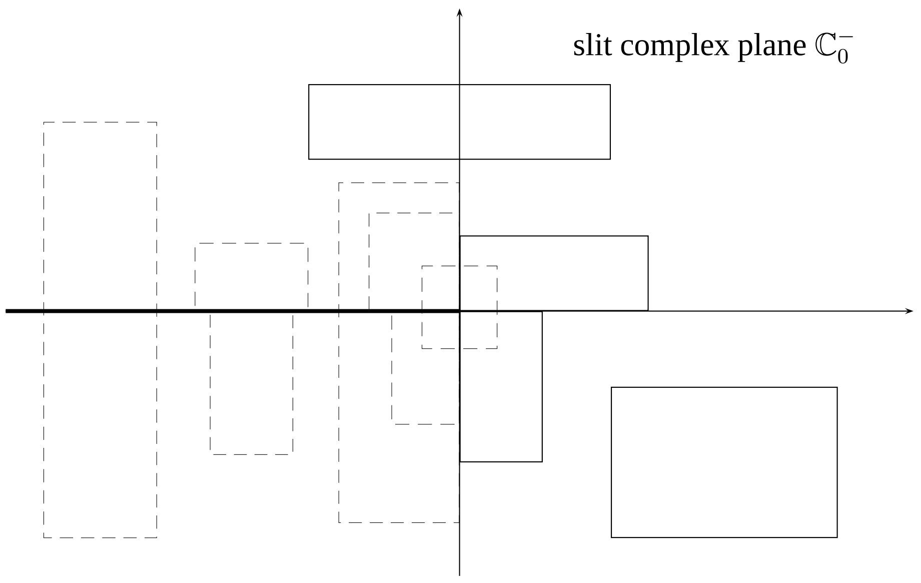

Figure 6: Allowed and not allowed complex arguments Z for sqrt(Z, n).

The complex intervals sqrt(Z, n) can be included by the C-XSC function

```
cinterval sqrt(const cinterval& Z, int n);
```

implemented in `cimath.cpp`.

Replacing `exp(Z)` by `sqrt(Z,3)` in the sample program on page 11, then with $Z1 := [-1, -1] + i \cdot [1, 1]$ and $Z2 := [0, 1] + i \cdot [-1, +1]$ the following inclusions are calculated:

```
sqrt(Z1,3) = ([7.937005259840970E-001,7.937005259841018E-001],
              [7.937005259840969E-001,7.937005259841018E-001])

sqrt(Z2,3) = ([+0.000000000000000E+000,1.084215081491354E+000],
              [-5.000000000000012E-001,5.000000000000012E-001])
```

It holds

$$\sqrt[3]{-1+i} = 2^{1/6}\{\cos(\pi/4) + i\,\sin(\pi/4)\} = 0.7937005\ldots + i\cdot 0.7937005\ldots$$

and consequently $\sqrt[3]{-1+i} \in$ `sqrt(Z1,3)`.
With $Z = [-1,-1] + i\cdot[0,0]$ the function call `sqrt(Z,3)` leads to an error, because Z must not touch the negative real axis.

For a given argument interval $Z \subset \mathbb{C} - \{0\}$ the C-XSC function

```
list<cinterval>sqrt_all(const cinterval& Z, int n)
```

calculates a list of intervals including all solutions of $w^n = z,\ z \in Z,\ n \in \mathbb{N}_0$. The only restriction for Z is now $0 \notin Z$, and so function values of the n-th root can be calculated, even if Z contains negative real values.

Replacing `sqrt_all(Z)` by `sqrt_all(Z,2)` in the sample program on page 24, then with $Z = [-1,-1] + 0\cdot i$ the program output is

```
sqrt_all(Z,2) = ([+0.000000000000000E+000,+0.000000000000000E+000],
                 [+9.999999999999998E-001,+1.000000000000001E+000])
                ([-0.000000000000000E+000,-0.000000000000000E+000],
                 [-1.000000000000001E+000,-9.999999999999998E-001])
```

In the next example all solutions w_i of $w^3 = -1+i$ are to be included. With

$$-1+i = \sqrt{2}\,\{\cos(3\pi/4 + 2k\pi) + i\cdot\sin(3\pi/4+2k\pi)\}\,, \quad k \in \mathbb{Z}$$

and (12) we get

$$\begin{aligned} w_1 &= 2^{1/6}\{\cos(\pi/4) + i\cdot\sin(\pi/4)\} \\ w_2 &= 2^{1/6}\{\cos(11\pi/12) + i\cdot\sin(11\pi/12)\} \\ w_3 &= 2^{1/6}\{\cos(19\pi/12) + i\cdot\sin(19\pi/12)\} \end{aligned}$$

Replacing `sqrt_all(Z)` by `sqrt_all(Z,3)` in the sample program on page 24, then with $Z = [-1,-1] + i\cdot[1,1]$ the program output is

```
sqrt_all(Z,3) = ([+7.937005259840979E-001,+7.937005259841020E-001],
                 [+7.937005259840979E-001,+7.937005259841020E-001])
                ([-1.084215081491354E+000,-1.084215081491348E+000],
                 [+2.905145555072493E-001,+2.905145555072533E-001])
                ([+2.905145555072494E-001,+2.905145555072535E-001],
                 [-1.084215081491354E+000,-1.084215081491348E+000])
```

and the above three complex intervals are inclusions of the $w_i,\ i = 1,2,3$.

3.2.5 Power Function for Real and Complex Exponents

The principal branch of the power function z^p is defined by

$$z^p = e^{p\cdot\ln(z)} = e^{p\cdot(\ln(r)+i\varphi)},\ -\pi < \varphi \leq \pi,\ z \in \mathbb{C} - \{0\},\ p \in \mathbb{C}.$$

z^p is analytic for all $z \in \mathbb{C}^-$ and continuous on the branch cut defined by the negative real axis including the origin.

For rectangular complex intervals Z and P the inclusion function is defined as

$$\text{(13)} \qquad \text{pow}(Z,P) := e^{P\cdot\text{Ln}(Z)} \supseteq \{z^p \mid z \in Z, p \in P\}, \quad P \in \mathbb{IC}.$$

Writing $Z = [\underline{x},\overline{x}] + i\cdot[\underline{y},\overline{y}]$, in the following cases

- $0 \in Z$ or
- $\underline{x} < 0$ and $\underline{y} < 0 \leq \overline{y}$

the complex interval function pow(Z,P) is not defined. Typical allowed and not allowed intervals Z are diagrammed in Figure 3 on page 20.

The complex interval pow(Z,P) can be included by the C-XSC function

```
    cinterval pow(const cinterval& Z, const cinterval& P);
```

implemented in `cimath.cpp`. With the following examples we show the practical application of this C-XSC function.

Example 1.
Include the principal value of $w = i^i$. It holds

$$\begin{aligned} i^i &= e^{i\cdot\ln(i)} = e^{i\cdot\{0+i(\pi/2+2\pi k)\}} = e^{-\pi/2+2\pi k},\ k \in \mathbb{Z} \\ &= e^{-\pi/2}, \quad k = 0 \text{ for the principal branch.} \\ &= 0.207879\ldots \end{aligned}$$

The sample program

```
#include <iostream>     // for cout
#include "cimath.hpp"   // for pow()

using namespace cxsc;
using namespace std;

real pi1 = 7074237752028440.0 / 2251799813685248.0;
// PI is an optimal inclusion of pi
interval PI = interval(pi1,succ(pi1));

int main()
{
    cinterval Z,P,W;
    Z = cinterval( interval(0,0), interval(1,1) );
    P = cinterval( interval(0,0), interval(1,1) );
    W = pow(Z,P);
    cout << SetPrecision(15,15)
         << Scientific << "pow(Z,P) = " << W << endl;
}
```

delivers the output

```
pow(Z,P) = ([2.078795763507614E-001,2.078795763507622E-001],
            [0.000000000000000E+000,0.000000000000000E+000])
```

and this complex interval is a guaranteed inclusion of the principal value of $w = i^i$.

Example 2.
Include the principal value of $w = \pi^i$. It holds

$$\begin{aligned} \pi^i &= e^{i\cdot\ln(\pi)} = e^{i\cdot\{\ln(\pi)+i(0+2\pi k)\}} = e^{i\cdot\ln(\pi)}, \text{ for } k=0, \\ &= \cos(\ln(\pi)) + i\cdot\cos(\ln(\pi)) = 0.4132\ldots + i\cdot 0.9105\ldots \end{aligned}$$

With `Z = cinterval( PI,interval(0,0) );` the above sample program delivers the output

```
pow(Z,P) = ([4.132921161015923E-001,4.132921161015961E-001],
            [9.105984992126129E-001,9.105984992126171E-001])
```

This complex machine interval is a guaranteed inclusion of the principal value of $w = \pi^i$.
In C-XSC another power function is implemented

```
      cinterval pow(const cinterval& Z, const interval& P);
```

whereby the exponent is now a real interval $P \in \mathbb{IR}$. As in (13) it holds

$$\text{pow}(Z,P) := e^{P\cdot\text{Ln}(Z)} \subseteq \texttt{pow(Z,P)},$$

and typical allowed and not allowed intervals Z are diagrammed in Figure 3 on page 20. As an application we consider

$$\begin{aligned} i^\pi &= e^{\pi\cdot\ln(i)} = e^{\pi\cdot\{0+i(\pi/2+2\pi k)\}}, \quad k\in\mathbb{Z} \\ &= e^{i\cdot\pi^2/2}, \text{ with } k=0 \text{ for the principal value} \\ &= \cos(\pi^2/2) + i\cdot\sin(\pi^2/2) = 0.22058\ldots - i\cdot 0.97536\ldots \end{aligned}$$

With `Z = cinterval(interval(0),interval(1)); P = interval(PI);` the sample program[3] on page 28 delivers the output

```
 pow(Z,P) = ([+2.205840407496965E-001,+2.205840407496999E-001],
             [-9.753679720836335E-001,-9.753679720836295E-001])
```

This complex interval is a guaranteed inclusion of the principal value of $w = i^\pi$.

[3]In the sample program the declaration   interval P;   is necessary.

3.2.6 The Inverse Sine

For $Z \subset R_S := \mathbb{C} - \{(-\infty, -1) \cup (1, \infty)\}$ the C-XSC function

```
cinterval asin(const cinterval& Z);
```

calculates a rectangular complex interval `asin(Z)` that includes the set

$$\{\arcsin(z) \mid z \in Z \subset R_S\} \subseteq \texttt{asin(Z)}.$$

In Figure 7 some typical allowed and not allowed interval arguments Z are drawn with solid and dashed lines respectively. On the two branch cuts the function values are defined by limits using directions that are characterized by the respective double arrows.

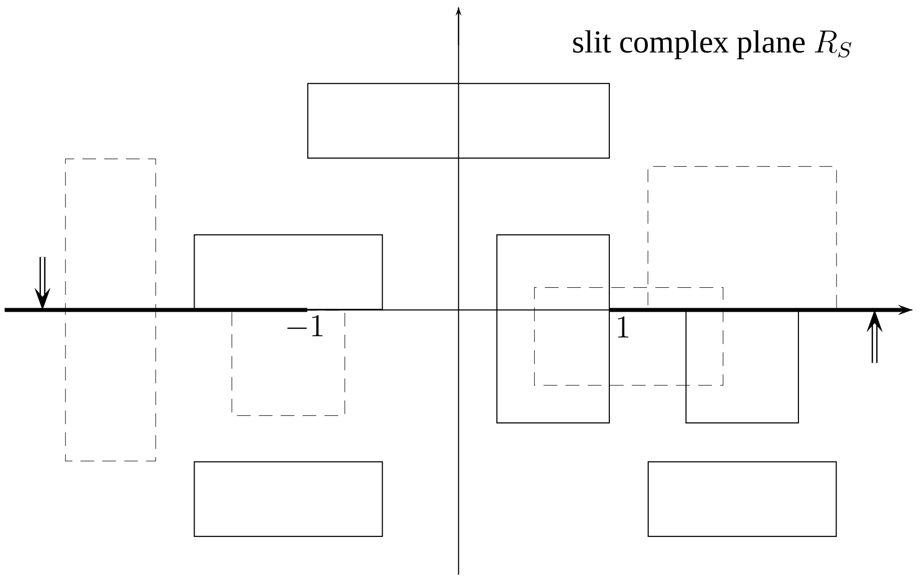

Figure 7: Allowed and not allowed complex arguments Z for asin(Z).

Replacing `exp(Z)` by `asin(Z)` in the sample program on page 11, then with
$Z = [1, 1] + i \cdot [2^{-200}, 2^{-200}]$ the program output is

```
asin(Z) = ([1.570796326794896E+000,1.570796326794897E+000],
           [7.888609052210116E-031,7.888609052210122E-031]).
```

With the input $Z =$ `([1E+200,1E+200],[-1E+200,-1E+200])` we get the program output[4]

```
asin(Z) = ([+7.853981633974459E-001,+7.853981633974511E-001],
           [-4.615567393696502E+002,-4.615567393696481E+002]).
```

[4]It holds $[10^{200}, 10^{200}] + i \cdot [-10^{200}, -10^{200}] \subset Z$, but Z is no point interval!

3.2.7 The Inverse Cosine

Because of the identity $\arccos(z) \equiv \frac{\pi}{2} - \arcsin(z)$ the inverse cosine function is defined on the same slit complex plane $R_S := \mathbb{C} - \{(-\infty, -1) \cup (1, \infty)\}$, which as well is the domain of the principal value of the inverse sine.

Hence, for $Z \subset R_S$ the C-XSC function

```
cinterval acos(const cinterval& Z);
```

calculates a rectangular complex interval `acos(Z)` including the set

$$\{\arccos(z) \,|\, z \in Z \subset R_S\} \subseteq \texttt{asin(Z)}.$$

In Figure 7 some typical allowed and not allowed interval arguments Z are drawn with solid and dashed lines respectively. On the two branch cuts the function values are defined by limits using directions that are characterized by the respective double arrows.

Replacing `exp(Z)` by `acos(Z)` in the sample program on page 11, then with $Z = [1, 1] + i \cdot [2^{-200}, 2^{-200}]$ the program output is

```
acos(Z) = ([+7.888609052210116E-031,+7.888609052210124E-031],
           [-7.888609052210122E-031,-7.888609052210116E-031]),
```

and with $Z = [-1, -1] + i \cdot [2^{-200}, 2^{-200}]$ we get

```
acos(Z) = ([+3.141592653589792E+000,+3.141592653589794E+000],
           [-7.888609052210122E-031,-7.888609052210116E-031]).
```

For $\arccos(4+0\cdot i)$ together with $Z = [4, 4]+i\cdot[0, 0]$ the sample program delivers the inclusion

```
acos(Z) = ([0.000000000000000E+000,0.000000000000000E+000],
           [2.063437068895556E+000,2.063437068895566E+000]).
```

Be aware that $z = 4 + 0 \cdot i$ lies on one of the two branch cuts. Now with $z = 4 + i \cdot 10^{-200}$ we choose an argument close above this branch cut, and the input
`Z = ([4,4],[1e-200,1e-200])` $\ni z$ delivers the inclusion

```
acos(Z) = ([+2.581988897471607E-201,+2.581988897471615E-201],
           [-2.063437068895566E+000,-2.063437068895556E+000]).
```

That is, $\arccos(z) \in$ `acos(Z)`.

With $z = 4 - i \cdot 10^{-200}$ and `Z = ([4,4],[-1e-200,-1e-200])` $\ni z$ the output of the sample program is

```
acos(Z) = ([2.581988897471607E-201,2.581988897471615E-201],
           [2.063437068895556E+000,2.063437068895566E+000]).
```

3.2.8 The Inverse Hyperbolic Sine

For $Z \subset R_T := \mathbb{C} - \{i(-\infty, -1) \cup i(1, \infty)\}$ the C-XSC function

```
cinterval asinh(const cinterval& Z);
```

calculates a rectangular complex interval `asinh(Z)` including the set

$$\{\text{arsinh}(z) \mid z \in Z \subset R_T\} \subseteq \texttt{asinh(Z)}.$$

In Figure 8 some typical allowed and not allowed interval arguments Z are drawn with solid and dashed lines respectively. On the two branch cuts the function values are defined by limits using directions that are characterized by the respective double arrows.

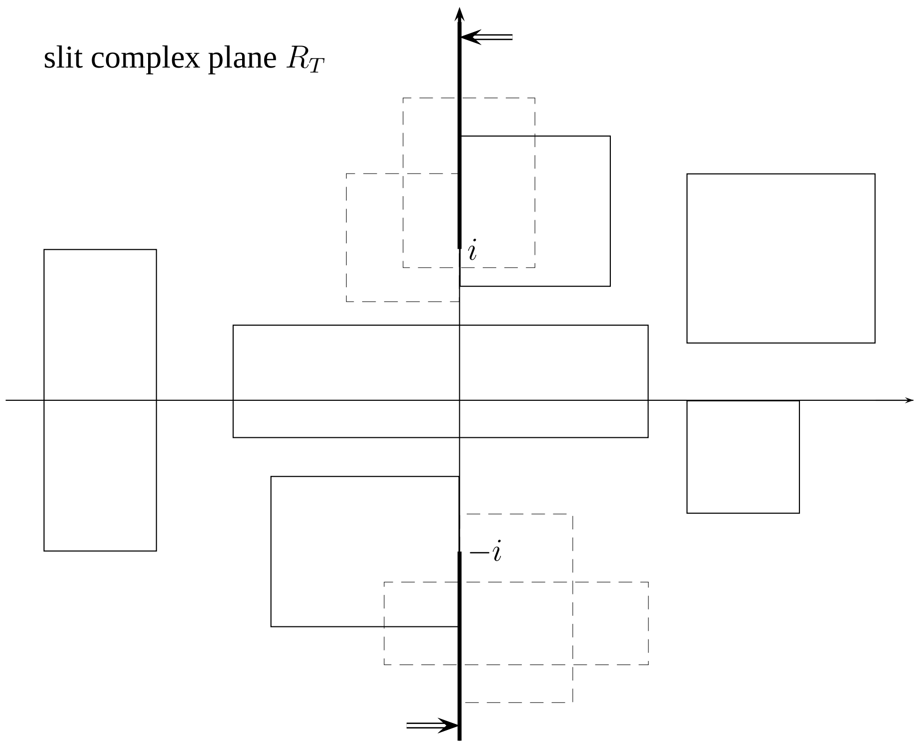

Figure 8: Allowed and not allowed complex arguments Z for asinh(Z).

The inclusion function is implemented in C-XSC as

$$\texttt{asinh}(Z) = i \cdot \texttt{asin}(-i \cdot Z),$$

and the necessary multiplications with i are error-free operations.

3.2.9 The Inverse Hyperbolic Cosine

For $z \in R_C := \mathbb{C} - (-\infty, +1)$ the principal value of the inverse hyperbolic cosine is defined as

$$\text{arcosh}(z) := \begin{cases} i \cdot \arccos(z), & \Im\{\arccos(z)\} < 0, \\ \text{arcosh}(x), & \Im\{z\} = 0, \\ -i \cdot \arccos(z), & \Im\{\arccos(z)\} > 0. \end{cases}$$

For $Z \subset R_C$ the inclusion function for the inverse hyperbolic cosine is defined as

$$\text{arcosh}(Z) := \square\{\text{arcosh}(z) \mid z \in Z\},$$

and for $Z \subset R_C$ the C-XSC function

```
cinterval acosh(const cinterval& Z);
```

calculates a rectangular complex interval `acosh(Z)` including the interval $\text{arcosh}(Z)$, that is

$$\text{arcosh}(Z) \subseteq \texttt{acosh(Z)}, \quad Z \subset R_C.$$

The rectangular complex intervals Z may not touch or intersect the branch cut given by $(-\infty, +1)$.

Replacing `exp(Z)` by `acosh(Z)` in the sample program on page 11, then with $Z =$ `([-1,-1],[1E-200,1E-200])` the program output is

```
 acosh(Z) = ([9.999999999999997E-101,1.000000000000001E-100],
             [3.141592653589792E+000,3.141592653589794E+000]),
```

and with $Z =$ `([-1,-1],[-1E-200,-1E-200])` we get

```
acosh(Z) = ([+9.999999999999997E-101,+1.000000000000001E-100],
            [-3.141592653589794E+000,-3.141592653589792E+000]).
```

If we consider $Z =$ `([-1,-1],[1E-200,1])`, then for $\text{arcosh}(Z)$ our sample program calculates the following inclusion

```
acosh(Z) = ([9.999999999999997E-101,1.061275061905038E+000],
            [2.237035759287405E+000,3.141592653589794E+000]).
```

3.2.10 The Inverse Tangent

With $z = x + iy$ the principal branch of the inverse tangent function is

$$\arctan(z) := \frac{1}{2i} \cdot \text{Ln}\frac{1+iz}{1-iz}, \quad z \in R_A := \mathbb{C} - \{iy \,\big|\, |y| \geq 1\}.$$

Let $Z \subset R_A$ be any rectangular complex interval, then the inclusion function for $\arctan(z)$ is defined as

$$\arctan(Z) := \square\{\arctan(z) \mid z \in Z \subset R_A\}.$$

In C-XSC the inclusion function `atan`(Z) is declared as

```
cinterval atan(const cinterval& Z);
```

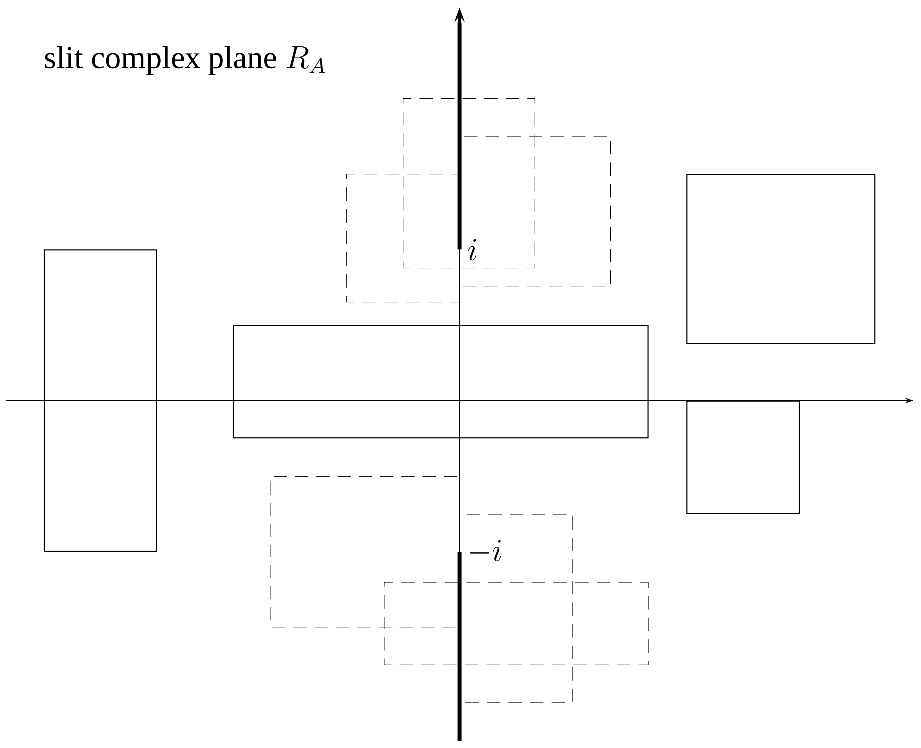

Figure 9: Allowed and not allowed complex arguments Z for arctan(Z).

The improved implementation is explained in [4]. For all $Z \subset R_A$ it holds

$$\arctan(Z) \subseteq \texttt{atan}(Z).$$

In Figure 9 some typical allowed and not allowed interval arguments Z are drawn with solid and dashed lines respectively.

Replacing `exp(Z)` by `atan(Z)` in the sample program on page 11, then with $Z1$ = `([1E+300,E+300],[1,1])` and $Z2$ = `([1E-300,E-300],[1,1])` the program output is[5]

```
atan(Z1) = ([1.570796326794896E+000,1.570796326794897E+000],
            [0.000000000000000E+000,4.940656458412466E-324]),

atan(Z2) = ([7.853981633974481E-001,7.853981633974486E-001],
            [3.457343375393865E+002,3.457343375393873E+002]).
```

3.2.11 The Inverse Hyperbolic Tangent

With $z \in R_I := \mathbb{C} - \{(-\infty, -1] \cup [+1, +\infty)\}$ the principal branch of the inverse hyperbolic tangent can be defined as

$$\text{artanh}(z) := -i \cdot \arctan(iz), \quad z \in R_I.$$

[5] 10^{300} and 10^{-300} are no machine numbers, hence $Z1$ and $Z2$ are no point intervals.

Let $Z \subset R_I$ be any rectangular complex interval, then the inclusion function for artanh(z) is

$$\text{artanh}(Z) := \Box\{\text{artanh}(z) \,|\, z \in Z \subset R_I\}.$$

In Figure 10 some typical allowed and not allowed interval arguments Z are drawn with solid and dashed lines respectively.

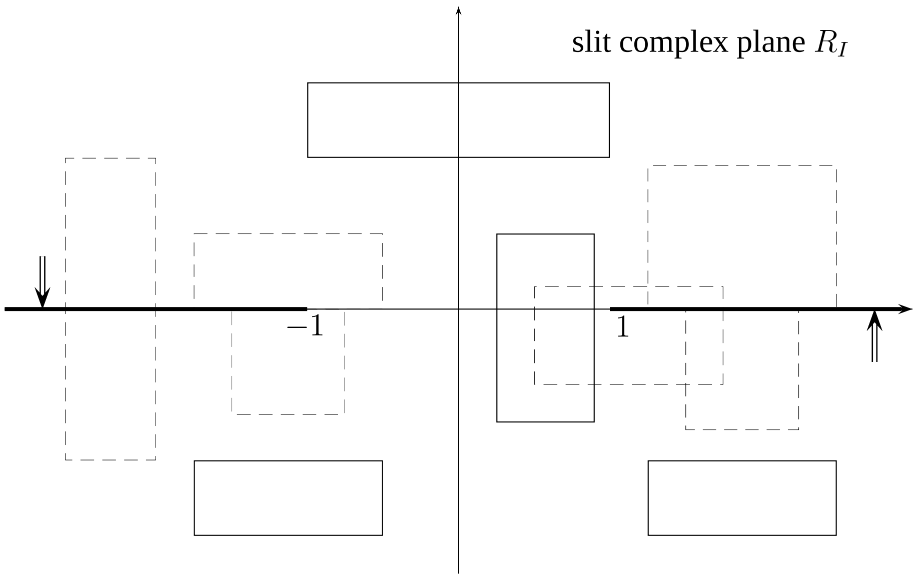

Figure 10: Allowed and not allowed complex arguments Z for artanh$(Z.)$

The two end points $(-1, 0)$ and $(+1, 0)$ belong to the branch cuts, hence the function artanh(z) is analytic in R_I.
Replacing `exp(Z)` by `atanh(Z)` in the sample program on page 11, then with
$Z1$ = `([1,1],[1E+200,1E+200])` and $Z2$ = `([-1,-1],[-1E+100,-1E+100])` the program output is

```
atanh(Z1)=([0.000000000000000E+000,4.940656458412466E-324],
           [1.570796326794896E+000,1.570796326794897E+000]),

atanh(Z2)=([-1.000000000000001E-200,-9.999999999999995E-201],
           [-1.570796326794897E+000,-1.570796326794896E+000]).
```

3.2.12 The Inverse Cotangent

With $z \in R_C := \mathbb{C} - \{iy \,|\, |y| \leq 1\}$ the principal branch of the inverse cotangent can be defined as

$$\text{arccot}(z) := \arctan(1/z), \quad z \in R_C.$$

Let $Z \subset R_C$ be any rectangular complex interval, then the inclusion function for arccot(z) is

$$\text{arccot}(Z) := \Box\{\text{arccot}(z) \,|\, z \in Z \subset R_C\}.$$

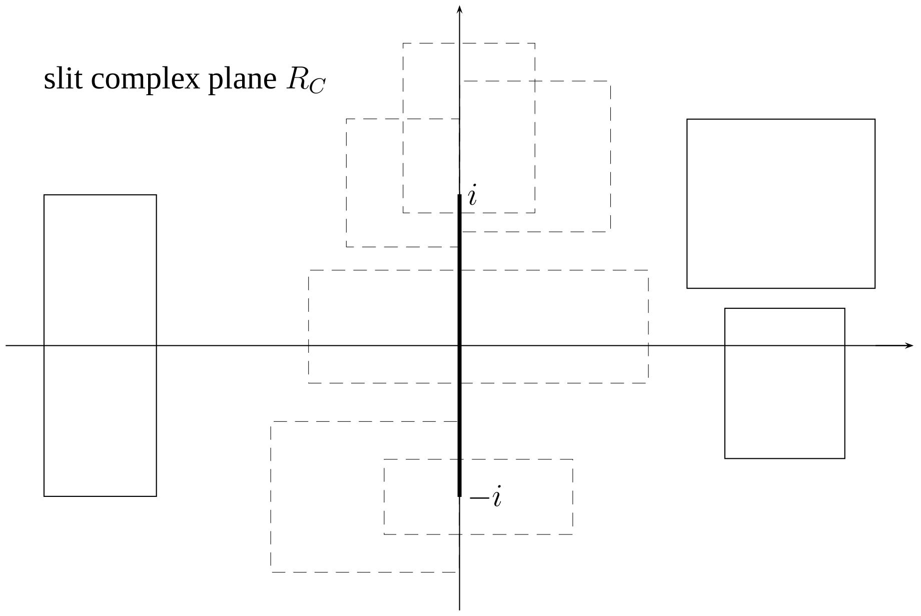

Figure 11: Allowed and not allowed complex arguments Z for arccot(Z).

Even though the code of the arccot procedure is very similar to the code of arctan, it has been implemented independently. Merely calling $\arctan(1/Z)$ would have simplified the programming, but also would have caused considerable overestimation in the result.
Replacing `exp(Z)` by `acot(Z)` in the program on page 11, then with

$$Z1 = \texttt{([1E+300,1E+300],[100,100])}$$
$$Z2 = \texttt{([1E+300,1E+300],[1E+300,1E+300])}$$

the program output is

```
acot(Z1) = ([+9.999999999999996E-301,+1.000000000000001E-300],
            [-4.940656458412466E-324,-0.000000000000000E+000]),

acot(Z2) = ([+4.999999999999997E-301,+5.000000000000003E-301],
            [-5.000000000000003E-301,-4.999999999999997E-301]).
```

3.2.13 The Inverse Hyperbolic Cotangent

With $z \in R_H := \mathbb{C} - \{(-1, +1)\}$ the principal branch of the inverse hyperbolic cotangent can be defined as

$$\text{arcoth}(z) := i \cdot \text{arccot}(iz), \quad z \in R_H.$$

For any $Z \subset R_H$ the inclusion function is

$$\text{arcoth}(Z) := \square\{\text{arcoth}(z) \mid z \in Z \subset R_H\}.$$

In C-XSC the inclusion function `acoth`(Z) is declared as

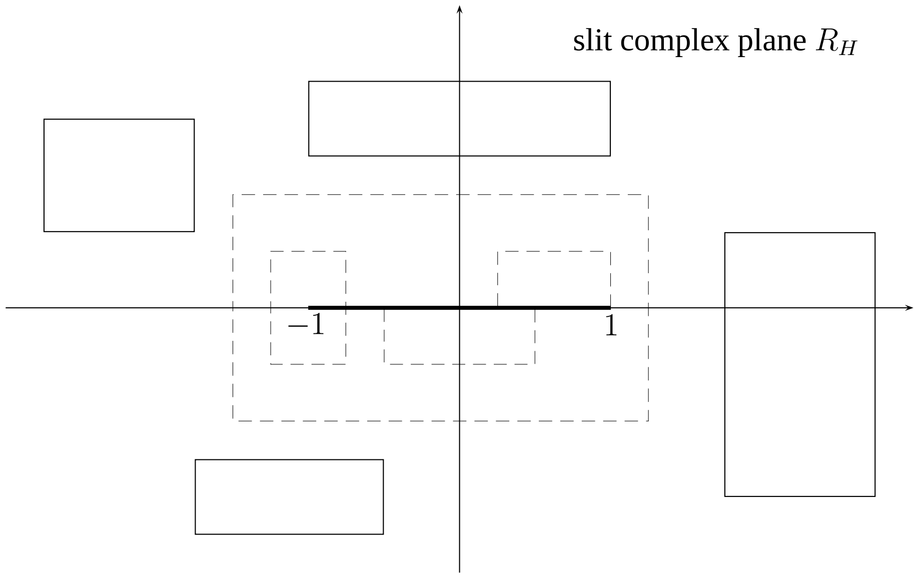

Figure 12: Allowed and not allowed complex arguments Z for $\operatorname{arcoth}(Z.)$

```
cinterval acoth(const cinterval& Z);
```

and it holds $\operatorname{arcoth}(Z) \subseteq$ `acoth`(Z) for all $Z \subset R_H$.

Replacing `exp(Z)` by `acoth(Z)` in the program on page 11, then with

$Z1 =$ `([1,1],[1E-300,1E-300])`
$Z2 =$ `([1,1],[1E+100,1E+100])`

the program output is

```
acoth(Z1) = ([+3.457343375393865E+002,+3.457343375393873E+002],
             [-7.853981633974484E-001,-7.853981633974482E-001]),

acoth(Z2) = ([+9.999999999999995E-201,+1.000000000000001E-200],
             [-1.000000000000001E-100,-9.999999999999997E-101]).
```

4 Applications

4.1 Evaluation of Complex Expressions

In this section we give some examples how to calculate guaranteed enclosures of complex expressions, composed of the complex standard functions. The diameters of the interval arguments of these functions should be sufficiently small to ovoid the well-known overestimations of the results.

Example 1.

To include

$$w := \arctan \left\{ 3 \left(\frac{1+i}{1-i} \right)^2 - 2 \left(\frac{1-i}{1+i} \right)^3 \right\},$$

we use the simple C-XSC program

```
#include <iostream>        // for cout
#include "cimath.hpp"      // for atan()

using namespace cxsc;
using namespace std;

int main()
{
    cinterval W,Z1,Z2;

    Z1 = cinterval(complex(1,1));
    Z2 = cinterval(complex(1,-1));
    W = atan( 3*power(Z1/Z2,2) - 2*power(Z2/Z1,3) );
    cout << SetPrecision(15,15)
         << Scientific << "W = " << W << endl;
}
```

The above program delivers the output

```
   W = ([-1.338972522294496E+000,-1.338972522294491E+000],
        [-1.469466662255313E-001,-1.469466662255280E-001])
```

and W is a guaranteed enclosure of the exact value

$$w = \arctan(-3 - 2i) \in W.$$

Example 2.

To include

$$w := \ln\left\{(1 + 2^{-200}i)^5\right\}$$

we use the simple C-XSC program

```
#include <iostream>         // for cout
#include "cimath.hpp"       // for Ln()

using namespace cxsc;
using namespace std;

int main()
{
    cinterval W1,W2,Z;

    Z = cinterval(complex(1,comp(0.5,-199)));
    // point interval Z = [1,1]+i*[2^(-200),2^(-200)]

    W1 = Ln( power(Z,5) );
    W2 = 5 * Ln(Z);
    cout << SetPrecision(15,15) << Scientific
         << "W1 = " << W1 << endl
         << "W2 = " << W2 << endl;
}
```

The above program delivers the output

```
W1 = ([-2.664535259100383E-015,2.220446049250314E-015],
      [3.111507638930558E-060,3.111507638930587E-060])

W2 = ([9.681479787123287E-121,9.681479787123310E-121],
      [3.111507638930570E-060,3.111507638930571E-060]).
```

With the assignment `W1 = Ln( power(Z,5) )` the inclusion of the real part $\Re\{w\} = \frac{5}{2} \cdot \ln(1 + 2^{-400})$ is almost useless, because here we first calculate the inclusion of $(1 + 2^{-200}i)^5$, which is a thick interval including the zero $z_0 = 1$ of the complex logarithm function, and after that the logarithm function is called with this interval argument. Hence the real part of `W1` is only a rough inclusion of $\Re\{w\}$.

On the other hand, with the assignment `W2 = 5 * Ln(Z)` the logarithm function is called with the complex point interval $Z = 1 + i \cdot 2^{-200}$, and although this interval Z lies rather close to the zero $z_0 = 1$, the logarithm function `Ln()` encloses $\ln(1 + 2^{-200}i)$ fairly good. Hence `W2` is a pretty well enclosure of w.

The conclusion of this example is to choose the appropriate equivalent expression for getting tight enclosures of the particular expression value.

Example 3.

For real arguments $a > 0$ and $b > 0$ it holds

(14) $$\ln(a \cdot b) = \ln(a) + \ln(b).$$

In this example we show that (14) is not valid for complex arguments a and b, if `arg`(a) + `arg`$(b) > \pi$. With $a = -1 + i$ and $b = i$ the C-XSC program

```
#include <iostream>         // for cout
#include "cimath.hpp"       // for Ln()

using namespace cxsc;
using namespace std;

int main()
{
    cinterval A,B,W1,W2;

    A = cinterval(complex(-1,1));
    B = cinterval(complex(0,1));

    W1 = Ln(A*B);
    W2 = Ln(A) + Ln(B);
    cout << SetPrecision(15,15) << Scientific
         << "W1 = " << W1 << endl
         << "W2 = " << W2 << endl;
}
```

delivers the output

```
W1 = ([3.465735902799723E-001,3.465735902799731E-001],
      [-2.356194490192348E+000,-2.356194490192343E+000])

W2 = ([3.465735902799723E-001,3.465735902799731E-001],
      [3.926990816987239E+000,3.926990816987245E+000])
```

and obviously `W1` and `W2` differ with regard to their imaginary parts.

In **Example 2** we used the formula

$$\ln(z^n) = n \cdot \ln(z), \tag{15}$$

which of course was valid for $z = 1 + 2^{-200}i$ and $n = 5$, however it is well known that (15) is not true for any z to the power of n. So be careful by simplifying complex expressions to get tight enclosures of particular expression values.

4.2 Verified Root Enclosures of Complex Expressions

We consider complex expressions composed of complex standard functions and the arithmetic operations. Finding the roots of such expressions is a non trivial problem in numerical mathematics. Most algorithms deliver only approximations of the exact zeros without any or with only weak statements concerning the accuracy.

In this section, we describe an algorithm that computes verified enclosures of such roots starting with an appropriate approximation for each of them. The used algorithm is based on the following Theorem 1, [6, 19], where $\underline{\cup}$ denotes the convex hull of two complex sets, $A \overset{\circ}{\subset} B$

means that A is fully contained in the interior of B, and $[S]$ is the interval hull of a complex set S.

Theorem 1.

Let Z be a complex interval, $z \in \mathbb{C}$, $0 \neq c \in \mathbb{C}$, and let $f : \mathbb{C} \to \mathbb{C}$ be an analytic function in the convex region $R \supset z \underline{\cup} (z + Z)$. If

$$-c \cdot f(z) + [(1 - c \cdot f'(z \underline{\cup} (z + Z))) \cdot Z] \overset{\circ}{\subset} Z \tag{16}$$

holds then there exists a unique zero z_0 of $f(z)$, with $z_0 \in z + Z$.

The proof is given on page 44 (see also [19]). In practice the constant $c \neq 0$ is set to $c = 1/f'(z)$.

In this section $F(Z)$ denotes the interval function, associated with $f(z)$. For example, if $f(z) = \sin(z^2 + z + 1)$, then the interval function is defined by[6] $F(Z) = \sin(Z^2 + Z + 1)$. In other words, the variable z must be substituted by the interval Z.

In the following algorithm it is not necessary to know initial sets including the roots, on the contrary we can start with any complex number ζ_0, which in general should lie sufficiently close to the root z_0.

$\oplus$, $\ominus$, $\odot$, $\oslash$ denote machine interval operations with complex machine intervals. Let $Zi := [\zeta_0]$ be the point interval including the complex start value ζ_0 and let $c := 1/\mathsf{Sup}(F'(Zi)) \neq 0$. Then with the point interval $C := [c]$ on the machine we have to verify

$$-C \odot F(Zi) \oplus (1 \ominus C \odot F'(Zi \obar (Zi \oplus Z))) \odot Z \overset{\circ}{\subset} Z, \tag{17}$$

where the left-hand side must lie in the interior of Z, and with (17) the inclusion relation (16) holds more than ever.

In the following example we calculate $F'(Z)$ by symbolic differentiation. This sometimes tedious procedure can be replaced by using an automatic differentiation, which delivers both function values, $F(Z)$ and $F'(Z)$ (see [11, 5]).

Example 1. The following example is taken from [19]:

$$f(z) = \arctan\big((z - a) \cdot \ln(z^2 - 5 \cdot z + 8 + i) \big)$$

$$f'(z) = \frac{\ln(z^2 - 5 \cdot z + 8 + i) + \dfrac{(z - a) \cdot (2 \cdot z - 5)}{z^2 - 5 \cdot z + 8 + i}}{1 + (z - a)^2 \cdot (\ln(z^2 - 5 \cdot z + 8 + i))^2}$$

Setting $a = 4i$ the zeros of $f(z)$ are

$$z_0 = 2 + i, \qquad z_1 = 3 - i, \qquad z_2 = 4i.$$

With appropriate approximations the following C-XSC program calculates guaranteed enclosures of the roots z_j.

[6]It should be noted that in this section $F(Z)$ of course is an inclusion function, yet not the optimal one.

```
/************************************************************/
/*  Program to enclose a simple zero of a complex expression  */
/*  composed of complex standard functions and arithmetic     */
/*  operations. Choose a complex start value sufficiently     */
/*  close to the zero!                                        */
/************************************************************/

#include <iostream>        // for cout
#include "cimath.hpp"      // for Ln(), ...

using namespace cxsc;
using namespace std;

// The zeros of function F() are to be enclosed.
complex a(0,4);  // a = 0 +4i, parameter of F().

cinterval F(const cinterval& Z)
{
    cinterval res;
    res = atan( (Z-a)*Ln(sqr(Z)-5*Z+complex(8,1)) );
    return res;
};

//  First derivative of F():
cinterval DERIV(const cinterval& Z)
{
    cinterval U,res;
    U = sqr(Z) - 5*Z + complex(8,1);
    res = (Ln(U)+(Z-a)*(2*Z-5)/U)/(1+sqr(Z-a)*sqr(Ln(U)));
    return res;
};

int main()
{
    cinterval Y,Z0,Zb,Z,D,CH,Zi,C;
    complex zeta0;
    int kmax(12),incl,k; // kmax: Maximum number of iteration steps
    real eps=0.125;

    while (1) {
        cout << "Complex zero approximation (x,y) = ? ";
        cin >> zeta0;
        Zi = cinterval(zeta0);

        for (int p=1; p<=5; p++)
            Zi = cinterval( Sup(Zi - F(Zi)/DERIV(Zi)) );
        cout << "Improved zero approximation: " << Sup(Zi) << endl;
        C = cinterval(1/Sup(DERIV(Zi))); // C: point interval
        Y = -C*F(Zi);  Z = Y; // Y,C: initial values of iteration
        k=0;

        do {
            k++;
            Zb = Blow(Z,eps); // epsilon inflation of Z
            CH = Zi | Zi+Zb;  // CH: complex hull
            D = 1 - C*DERIV(CH);
            Z = Y + D*Zb;
            incl = in(Z,Zb);  // Inclusion test
        }     // kmax: Maximum number of iteration steps
        while ( (k<kmax) and (incl==0) );

        if (incl) {   // incl=1 <--> Inclusion verified
            Z0 = Zi + Z;
            cout << SetPrecision(15,15)
                 << Scientific << "Zero found in: " << endl;
            cout << Z0 << endl; }
        else cout << "No inclusion found" << endl;
    }

} // main
```

Results:

```
Complex zero approximation (x,y) = ? (1.9,0.9)
Improved zero approximation:
  (2.000000007788749E+000,1.000000008855897E+000)
Zero found in:
([1.999999999999997E+000,2.000000000000003E+000],
 [9.999999999999975E-001,1.000000000000003E+000]).

Complex zero approximation (x,y) = ? (2.95,-1.05)
Improved zero approximation: ( 3.000000, -1.000000)
Zero found in:
([2.999999999999998E+000,3.000000000000002E+000],
 [-1.000000000000002E+000,-9.999999999999987E-001])

Complex zero approximation (x,y) = ? (0.1,3.9)
Improved zero approximation:
   (3.197344634236622E-036,4.000000000000001E+000)
Zero found in:
([-3.257083639014695E-030,3.147356704712094E-030],
 [3.999999999999999E+000,4.000000000000001E+000]).
```

The algorithm works well with sufficiently good approximations to the roots. Hence in the `for` loop of the program the input approximation is additionally improved using Newton's method. Of course, if convergence fails, then (17) cannot be fulfilled. On the other hand, if the inclusion relation (17) is valid, then a calculated wide interval $Z_0 \ni z_0$ can probably be improved by restarting the process and choosing supremum of the computed interval as a new approximation to the zero. We demonstrate this idea with the following

Example 2.

$$f(z) = \sin(z), \qquad f'(z) = \cos(z).$$

The roots are: $z_k = k \cdot \pi + 0i,\ k \in \mathbb{Z}$. We start the program with $\zeta_0 = 2 + i/2$.

```
Complex zero approximation (x,y) = ? (2,0.5)
Improved zero approximation:
(3.141592653589794E+000,4.911384887033602E-025)
Zero found in:
([3.141592653589793E+000,3.141592653589794E+000],
 [-1.653038930843842E-039,1.561203434685851E-039]).
```

With $\zeta_0 = (3.141592653589793E + 000, 1.561203434685851E - 039$ we get

```
Complex zero approximation (x,y) = ?
(3.141592653589793,1.561203434685851E-039)
Improved zero approximation:
(3.141592653589794E+000,3.248565551764031E-112)
Zero found in:
([3.141592653589793E+000,3.141592653589794E+000],
 [-1.061792541045360E-126,1.061792541045360E-126]).
```

Proceeding in the same way, the imaginary part of the zero $z_0 = \pi + 0 \cdot i$ can be included almost optimally in the IEEE system.

A Optimal Inclusion and Range of a Function

Let us consider the simple analytic function $f(z) = z^2$, with $z = x + i \cdot y \in \mathbb{C}$ and a rectangular complex interval $Z = X + i \cdot Y = [-2, 1] + i \cdot [-1, +1] \in \mathbb{IC}$. As we can see in Figure 13, the range Rg$(f, Z) := \{f(z) \,|\, z \in Z\}$ itself is not a rectangular interval, however the optimal inclusion function

$$F(Z) := \text{sqr}(Z) = [-1, 4] + i \cdot [-4, +4]$$

is a rectangle touching the range Rg(f, Z) in four points.

With $f(z) = z^2 = (x^2 - y^2) + i \cdot 2xy = u(x, y) + i \cdot v(x, y)$ the reader will easily proof that in the w-plane the contour of the range Rg(f, Z) is given by appropriate parts of the following two parabolas

$$v = \pm 2 \cdot \sqrt{1 + u}, \qquad v = \pm 4 \cdot \sqrt{4 - u}.$$

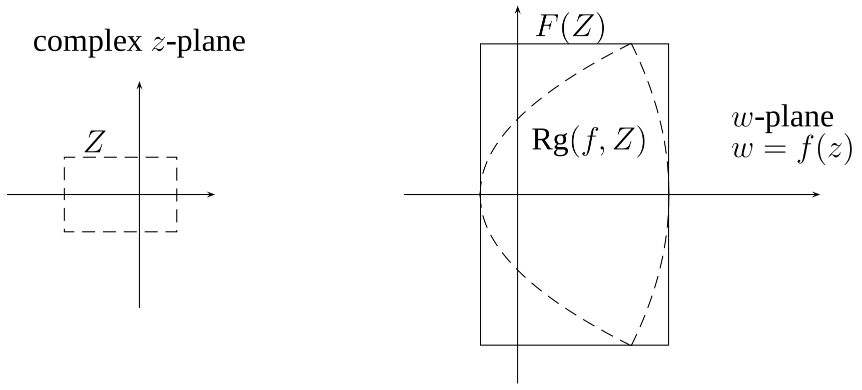

Figure 13: Range $\mathrm{Rg}(f, Z)$ and inclusion function $F(Z)$ for $f(z) = z^2$.

For example, in the z-plane the left parallel to the y-axis of Z is

$$\begin{aligned} z &= -2 + i \cdot t, \qquad t \in [-1, 1], \\ z^2 &= (4 - t^2) - i \cdot 4t = u(t) + i \cdot v(t), \\ \rightsquigarrow \quad \frac{v^2}{16} &= 4 - u, \qquad v = \pm 4 \cdot \sqrt{4 - u}. \end{aligned}$$

In this context it is important to observe that

$$F_1(Z) := Z \cdot Z = [-3, 5] + i \cdot [-4, 4] \supset F(Z) = \mathrm{sqr}(Z)$$

is just as well an inclusion function, yet not the optimal one.

B Proof of Theorem 1.

First some remarks and definitions:

- $z = x + i \cdot y \in \mathbb{C}$.
  In this section we define $|z| := |x| + |y|$, with $|z| \geq \sqrt{x^2 + y^2}$.
- $S \subset \mathbb{C}$ is an arbitrary subset of $\mathbb{C}$.
  In this section we define $|S| := \max\limits_{z \in S}\{|z|\}$,
  and $[\,S\,]$ denotes the interval hull of S.
- With $S_1 \subset \mathbb{C}$ and $S_2 \subset \mathbb{C}$ the convex hull is denoted by $S_1 \underline{\cup} S_2$.
- The diameter of a real interval $X = [\underline{x}, \overline{x}]$ is defined by $\mathrm{d}(X) := \overline{x} - \underline{x}$.
- The diameter of a complex interval $Z = X + i \cdot Y$ is defined by $\mathrm{d}(Z) := \mathrm{d}(X) + \mathrm{d}(Y)$.
  It can easily be shown that $\mathrm{d}(z + Z) = \mathrm{d}(Z)$.
- $S \cdot Z := \{w \mid w = s \cdot z;\ s \in S, z \in Z\}$.

With the above notations it holds[7]

$$\text{(18)} \qquad \mathrm{d}([\,S \cdot Z\,]) \geq |S| \cdot \mathrm{d}(Z).$$

Theorem 1. was formulated on page 40.

Let Z be a complex interval, $z \in \mathbb{C}$, $0 \neq c \in \mathbb{C}$, and let $f : \mathbb{C} \to \mathbb{C}$ be an analytic function in the convex region $R \supset z \underline{\cup} (z + Z)$, then with

$$\text{(19)} \qquad -c \cdot f(z) + [(1 - c \cdot f'(z \underline{\cup} (z + Z))) \cdot Z] \overset{\circ}{\subset} Z$$

there exists a unique zero z_0 of $f(z)$, with $z_0 \in z + Z$.

Proof (see also [19]):

Let $f : \mathbb{C} \to \mathbb{C}$ be an analytic function in R. For the range $\mathrm{h}(z+Z)$ of the analytic help function

$$\text{(20)} \qquad \mathrm{h}(z) := z - c \cdot f(z)$$

it holds the inclusion relation, [6, Theorem 5., page 35]

$$\text{(21)} \qquad \mathrm{h}(z + Z) \subset \mathrm{h}(z) + [\,\mathrm{h}'(z \underline{\cup} (z + Z)) \cdot Z\,].$$

With the definition of $\mathrm{h}(z)$ we find

$$\text{(22)} \qquad \mathrm{h}(z + Z) \subset z - c \cdot f(z) + [\,(1 - c \cdot f'(z \underline{\cup} (z + Z))) \cdot Z\,],$$

and together with (19) we get the inclusion relation

$$\text{(23)} \qquad \mathrm{h}(z + Z) \overset{\circ}{\subset} z + Z.$$

So with Brouwer's fixed-point theorem the help function $\mathrm{h}(z)$ has at least one fixed-point $z_0 \in z + Z$, that implies $\mathrm{h}(z_0) = z_0$, and with the definition of $\mathrm{h}(z)$ it follows that $f(z)$ has at least one zero $z_0 \in z + Z$.

In the next step it will be shown that this zero $z_0 \in z + Z$ is even unique. The inclusion relation (19) implies

$$z - c \cdot f(z) + [(1 - c \cdot f'(z \underline{\cup} (z + Z))) \cdot Z] \overset{\circ}{\subset} Z + z,$$

and for the diameter d we get the estimations

$$\begin{array}{rcl} \mathrm{d}(z + Z) & > & \mathrm{d}(z - c \cdot f(z) + [(1 - c \cdot f'(z \underline{\cup} (z + Z))) \cdot Z]) \\ & = & \mathrm{d}([(1 - c \cdot f'(z \underline{\cup} (z + Z))) \cdot Z]) \\ & \geq & |1 - c \cdot f'(z \underline{\cup} (z + Z))| \cdot \mathrm{d}(Z). \end{array}$$

The equal sign is substantiated by a simple displacement in the z-plane, generated by $z - c \cdot f(z)$, and the last line follows with (18). If in Theorem 1. the condition (19) is fulfilled, then Z cannot

[7] It should be noted that (18) is a generalization of $\mathrm{d}(A \cdot B) \geq |A| \cdot \mathrm{d}(B)$, where A and B are complex intervals, [2, (15), page 72].

be a point interval and so $\mathrm{d}(Z)$ must be positive. Hence $\mathrm{d}(z+Z) = \mathrm{d}(Z) > 0$, and, with the above estimation, we get

$$(24) \qquad |1 - c \cdot f'(z \underline{\cup} (z+Z))| < 1 \quad \text{resp.} \quad |\mathrm{h}'(z \underline{\cup} (z+Z))| < 1.$$

So in $z + Z \subset z\underline{\cup}(z+Z)$ the function $\mathrm{h}(z)$ is a contractive mapping. Then, with Banach's fixed-point theorem, there exists exactly one fixed-point z_0. Hence

$$z_0 \in z + Z, \quad \text{and} \quad \mathrm{h}(z_0) = z_0.$$

With (24) particularly we have

$$c \neq 0 \quad \text{and} \quad 0 \notin f'(z \underline{\cup} (z+Z)).$$

Hence, with the definition of $\mathrm{h}(z)$ it follows that the unique fixed-point z_0 of $\mathrm{h}(z)$ is a unique simple root of $f(z)$ ∎

References

[1] Abramowitz, M. and Stegun, I. *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables.* National Bureau of Standards, Washington, 1964.

[2] Alefeld, G. and Herzberger, J. *Introduction to Interval Computations.* Academic Press, New York, 1983.

[3] Bantle, A., Krämer, W. *Automatic Forward Error Analysis for Floating Point Algorithms,* Reliable Computing, Vol. 7, No. 4, pp. 321-340, 2001.

[4] Blomquist, F. *Anmerkungen zur Realisierung der komlexen Intervallfunktionen in der CoStLy-Bibliothek,* interner Bericht, 55 Seiten, WRSWT, Bergische Universität Wuppertal, 2005.

[5] Blomquist, F., Hofschuster, W., Krämer, W. *Real and Complex Taylor Arithmetic in C-XSC.* Preprint 2005/4, Universitt Wuppertal, 2005.

[6] Böhm, H. *Berechnung von Polynomnullstellen und Auswertung arithmetischer Ausdrücke mit garantierter Genauigkeit.* Dissertation, Universität Karlsruhe, 1983.

[7] Bohlender, G. *Floating-point computation of functions with maximum accuracy.* IEEE Trans. Comp. C-26 No. 7, 621–632, 1977.

[8] Braune, K. *Hochgenaue Standardfunktionen für reelle und komplexe Punkte und Intervalle in beliebigen Gleitpunktrastern.* Dissertation, Universität Karlsruhe, 1987.

[9] Braune K., Krämer, W. *High Accuracy Standard Functions for Real and Complex Intervals.* In Kaucher, E.; Kulisch, U. and Ullrich, Ch., editors, Computerarithmetic: Scientific Computation and Programming Languages, pages 81-114. Teubner, Stuttgart, 1987.

[10] Braune, K. *Standard Functions for Real and Complex Point and Interval Arguments with Dynamic Accuracy.* Computing Supplementum, 6:159-184, 1988.

[11] Hammer, R.; Hocks, M.; Kulisch, U.; Ratz, D. *C++ Toolbox for Verified Computing.* Springer, 1994.

[12] Higham, N. J. *Accuracy and Stability of Numerical Algorithms,* 2nd ed., SIAM, 2002.

[13] Hofschuster W., Krämer, W. *Ein rechnergestützter Fehlerkalkül mit Anwendungen auf ein genaues Tabellenverfahren,* Preprint Nr. 96/5, Institut für Wissenschaftliches Rechnen und Mathematische Modellbildung, Karlsruhe, 1996.

[14] Hofschuster W., Krämer, W. *A Computer Oriented Approach to Get Sharp Reliable Error Bounds,* Reliable Computing 3, pp. 239-248, 1997.

[15] Hofschuster, W. and Krämer, W. *FI_LIB, eine schnelle und portable Funktionsbibliothek für reelle Argumente und reelle Intervalle im IEEE-double-Format,* Preprint 98/7 des Instituts für Wissenschaftliches Rechnen und Mathematische Modellbildung (IWRMM) an der Universität Karlsruhe, 1998.

[16] Hofschuster, W. *Zur Berechnung von Funktionswerteinschließungen bei speziellen Funktionen der mathematischen Physik,* Dissertation, Universität Karlsruhe, 2000.

[17] Hofschuster, W., Krämer, W. *C-XSC – A C++ Class Library for Extended Scientific Computing.* In *Numerical Software with Result Verification*. R. Alt, A. Frommer, B. Kearfott, W. Luther (eds), Springer Lecture Notes in Computer Science, LNCS 2991, 2004.

[18] Klatte, R.; Kulisch, U.; Lawo, Ch.; Rauch, M. and Wiethoff, A. *C-XSC: A C++ class library for extended scientific computing.* Springer, Berlin, 1993.

[19] Krämer, W. *Inverse Standardfunktionen für reelle und komplexe Intervallargumente mit a priori Fehlerabschätzungen für beliebige Datenformate.* Dissertation, Universität Karlsruhe, 1987.

[20] Krämer, W. *Inverse Standard Functions for Real and Complex Point and Interval Arguments with Dynamic Accuracy.* Computing Supplementum, 6:185-212, 1988.

[21] Krämer, W. *A priori Worst Case Error Bounds for Floating-Point Computations,* IEEE Transactions on Computers, Vol. 47, No. 7, 1998.

[22] Kulisch, U. and Miranker, W. L. *Computer Arithmetic in Theory and Practice.* Academic Press, New York, 1981.

[23] Moore R. E. *Interval Analysis.* Prentice Hall, Englewood Cliffs, N.J., 1966.

[24] Neher, M. *The mean value form for complex analytic functions.* Computing, 67:255-268, 2001.

[25] Neher, M. *Complex Standard Functions and their Implementation in the CoStLy Library.* Preprint Nr. 04/18, Fakultät für Mathematik, Universität Karlsruhe, 2004.

[26] Ratschek, H. and Rokne, J. *Computer Methods for the Range of Functions.* Ellis Horwood Limited, Chichester, 1984.

[27] Wolff v. Gudenberg, J. *Einbettung allgemeiner Rechnerarithmetik in PASCAL mittels eines Operatorkonzeptes und Implementierung der Standardfunktionen mit optimaler Genauigkeit.* Dissertation, Universität Karlsruhe, 1980.

[28] Wolff v. Gudenberg, J. *Berechnung maximal genauer Standardfunktionen mit einfacher Mantissenlänge.* Elektronische Rechenanlagen. 26, H. 5, 230–238,1984.

[29] XSC website on programming languages for scientific computing with validation. `http://www.xsc.de` [October 2005].