

UNIVERSITÄT KARLSRUHE
FAKULTÄT FÜR MATHEMATIK

**Complex Standard Functions and their
Implementation in the CoStLy Library**

M. Neher

Preprint Nr. 04/18

Eingegangen am 28.10.2004

D-76128 KARLSRUHE

Complex Standard Functions and their Implementation in the CoStLy Library

M. Neher

Institut für Angewandte Mathematik, Universität Karlsruhe,
76128 Karlsruhe, Germany

Abstract

The practical calculation of range bounds for some complex standard functions is addressed in this paper. The functions under consideration are root and power functions, the exponential, trigonometric and hyperbolic functions, and their inverse functions. For such a function f and a given rectangular complex interval Z , some interval W is computed that contains all function values of f in Z . This is done by expressing the real and the imaginary part of f as compositions of real standard functions and then estimating the ranges of these compositions. In most cases, the inclusions are optimal, such that W is the smallest rectangular interval containing the range of f .

The algorithms presented in this paper have been implemented in a C++ class library called CoStLy (Complex Standard Functions Library), which is distributed under the conditions of the GNU General Public License.

MSC Subject Classifications: 30-04, 65G30, 65K05.

Key Words: Complex standard functions, complex interval arithmetic, enclosure methods, mathematical software.

1 Introduction

Today, most problems in applied mathematics or engineering sciences are solved with the help of computers, using floating-point calculations. These, however, are subject to roundoff errors, so that instead of a precise result, an approximation is computed. The accumulated roundoff errors can become quite large, and they are not easily bounded by analytic methods. For bounding these errors rigorously, and thus restoring mathematical correctness in computations with finite arithmetic, floating-point interval arithmetic [12] has been found to be particularly useful. In floating-point interval arithmetic, all calculations are performed on intervals with machine representable bounds instead of floating-point numbers. The rules of interval arithmetic as described in [2, 8, 16, 18] are extended by outward rounding, such that the exact result of any arithmetic operation is automatically enclosed in a floating-point interval, including roundoff errors.

Over the last decades, interval arithmetic has been implemented in many different programming environments, such as the programming language extensions Pascal-XSC [10], C-XSC [9], Fortran-XSC [23], the beam physics code COSY INFINITY [3], the Matlab toolbox INTLAB [22], filib++ [13, 14], MPFI [21], or the Maple package intpakX [7]. Besides the basic arithmetic operations, these software packages also offer validated computations for the usual set of real standard functions. If f is such a function and X a floating-point interval in the domain of f for which the range is bounded by machine representable numbers, then some floating-point interval W that contains the range is calculated.

For the validated computation of complex standard functions, Bühler [6] implemented the algorithms that were presented in [4, 5, 11] in a Pascal-XSC function library. INTLAB includes subroutines for complex standard functions using circular arithmetic. IntpakX also contains procedures for circular complex interval arithmetic, but apart from the evaluation of complex polynomials, only the exponential function is implemented. To the author's knowledge, the other packages mentioned currently do not supply validated complex functions.

Unfortunately, the development and maintenance of Pascal-XSC has stopped. The compilers needed for Pascal-XSC are outdated [24], so that the availability and implementability on today's computers has become complicated and the future development of Pascal-XSC is unclear. Presently, INTLAB seems to be the sole library for the rigorous calculation of complex functions that is widely available. It requires a valid Matlab license, however, so that INTLAB is not a truly free software, even though it is distributed free of charge for private use and for purely academic purposes.

Apart from INTLAB, C-XSC is probably the most advanced of the above libraries. It contains structured interval data types, real interval functions and even the complex interval data type with the basic arithmetic operations (including optimal division) between rectangular complex intervals, but no complex standard functions. It is distributed freely under the GNU public license and runs on many different platforms. This wide availability was the main reason for building CoStLy on C-XSC. CoStLy also works with the flib++ library, which is less comprehensive than C-XSC and for some of its functions exhibits a larger amount of overestimation than C-XSC, but which is generally faster than the C-XSC library.

CoStLy has been developed as a new C++ library for the validated computation of function values and of ranges of complex standard functions. Some of the implemented function have been derived (with modification) from the Pascal-XSC code presented in [6], some functions are new.

For readers who are not familiar with interval computations, the necessary fundamentals for understanding this paper are outlined in section 2. In section 3, complex inclusion functions and their implementation in software are discussed. The CoStLy library is presented in section 4. The paper closes with the presentation of numerical examples, including a comparison with INTLAB.

2 Notation and Preliminary Remarks

2.1 Standard Functions

The following set of standard functions is considered in this paper:

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

where $\text{power}(z, n)$ is the power function for integer exponents and $\text{pow}(z, p)$ the power function for real or complex exponents.

2.2 Complex Numbers and Sets

Complex numbers will be denoted by $z = x + iy$, where $x = \text{Re } z$, $y = \text{Im } z$. For a complex analytic function f we frequently write

$$f(z) = u(x, y) + w(x, y),$$

where $u(x, y) = \text{Re } f(z)$, $v(x, y) = \text{Im } f(z)$. 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 $\mathbb{C} - (-\infty, 0) = \mathbb{C}^- \cup 0$ is denoted by \mathbb{C}_0^- .

2.3 Interval Arithmetic

Interval Arithmetic as described in [2, 8, 16, 18] is a powerful tool for validated computations. In interval arithmetic, operations between intervals are employed to calculate guaranteed bounds for continuous problems with a finite number of basic arithmetic operations.

2.3.1 Real Interval Arithmetic

A compact real interval X is defined by a pair of two real numbers \underline{x} , \bar{x} with $\underline{x} \leq \bar{x}$: $X = [\underline{x}, \bar{x}]$. A real number x is identified with a point interval $X = [x, x]$. The set of all compact real intervals is denoted by \mathbb{IR} . Throughout this paper, intervals are denoted by capital letters.

$$w(X) := \bar{x} - \underline{x}$$

is called the *width* of X .

The basic arithmetic operations between real intervals are defined by

$$A \bullet B := \{a \bullet b \mid a \in A, b \in B\}, \quad \bullet \in \{+, -, \cdot, /\},$$

provided that $0 \notin B$ in the case of division. For $\bullet \in \{+, -, \cdot, /\}$, $C := A \bullet B$ is an interval, and the bounds of C may be calculated as [2, p. 2]

$$A + B = [\underline{a} + \underline{b}, \bar{a} + \bar{b}], \quad (1)$$

$$A - B = [\underline{a} - \bar{b}, \bar{a} - \underline{b}], \quad (2)$$

$$A \cdot B = [\min\{\underline{a}\underline{b}, \underline{a}\bar{b}, \bar{a}\underline{b}, \bar{a}\bar{b}\}, \max\{\underline{a}\underline{b}, \underline{a}\bar{b}, \bar{a}\underline{b}, \bar{a}\bar{b}\}], \quad (3)$$

$$A / B = A \cdot [1 / \bar{b}, 1 / \underline{b}]. \quad (4)$$

2.3.2 Rectangular Complex Interval Arithmetic

A rectangular complex interval Z is defined by a pair of two real intervals X and Y :

$$Z = X + \imath Y, \quad Z = \{z = x + \imath y \mid x \in X, y \in Y\}.$$

The set of all complex rectangular intervals is denoted by \mathbb{IC} . For a bounded subset M of \mathbb{C} , the *interval hull* $\square M$ of M is the smallest rectangular interval that contains M . We have

$$\square M = [\inf_{z \in M} \operatorname{Re} z, \sup_{z \in M} \operatorname{Re} z] + \imath [\inf_{z \in M} \operatorname{Im} z, \sup_{z \in M} \operatorname{Im} z].$$

Addition and multiplication of rectangular complex intervals are given by the following definition ([2, Definition 5.3]): Let $Z_1 = X_1 + \imath Y_1$ and $Z_2 = X_2 + \imath Y_2$ be two complex intervals, then

$$\begin{aligned} Z_1 \pm Z_2 &:= X_1 \pm X_2 + \imath(Y_1 \pm Y_2), \\ Z_1 \cdot Z_2 &:= X_1 Y_1 - X_2 Y_2 + \imath(X_1 Y_2 + X_2 Y_1). \end{aligned} \quad (5)$$

In general, we have

$$Z_1 \cdot Z_2 = \square\{z_1 \cdot z_2 \mid z_1 \in Z_1, z_2 \in Z_2\} \supseteq \{z_1 \cdot z_2 \mid z_1 \in Z_1, z_2 \in Z_2\}.$$

An algorithm for calculating

$$\square\{z_1/z_2 \mid z_1 \in Z_1, z_2 \in Z_2\}$$

has been given in [15].

For $Z \in \mathbb{IC}$, the real *interval of absolute values* of Z is denoted by

$$\operatorname{abs}(Z) := \{|z| : z \in Z\}.$$

2.3.3 Floating Point Interval Arithmetic

A floating point interval is defined by a pair of floating point numbers instead of real numbers. Rigor of the computation is restored by enclosing real or complex constants into floating point intervals and by performing all calculations with directed rounding according to the rules of interval arithmetic [12].

2.4 Inclusion Functions

For $D \subseteq \mathbb{C}$, the range $\{f(z) : z \in D\}$ of a function $f : D \rightarrow \mathbb{C}$ is denoted by $\operatorname{Rg}(f, D)$. An *inclusion function* F of a given function $f : \mathbb{C} \rightarrow \mathbb{C}$ is an interval function $F : \mathbb{IC} \rightarrow \mathbb{IC}$ that encloses the range of f on all intervals $Z \subseteq D$:

$$F(Z) \supseteq \operatorname{Rg}(f, Z) \quad \text{for all } Z \subseteq D.$$

If $F(Z) = \square \operatorname{Rg}(f, Z)$ holds for all $Z \subseteq D$, then F is called *inclusion optimal* or simply *optimal*. F is called *inclusion isotone*, if

$$Z_1 \subseteq Z_2 \Rightarrow F(Z_1) \subseteq F(Z_2)$$

holds for all $Z_1, Z_2 \subseteq D$.

The optimal inclusion function for a continuous real function on $D \subseteq \mathbb{R}$ is

$$F(X) := [\min_{x \in X} f(x), \max_{x \in X} f(x)] \quad \text{for all } X \subseteq D. \quad (6)$$

In general, the computation of $\min_{x \in X} f(x)$ and $\max_{x \in X} f(x)$ is impractical or even impossible. For the real standard functions in S_F , however, the monotonicity properties and locations of the local maxima and minima are well-known. For each $f \in S_F$ and arbitrary X in the domain of f , the range $\text{Rg}(f, X)$ is computable from function values of f at a few specific points in X , by evaluating f at the endpoints of X and by checking for local or absolute extremal values in X [4, 5, 11].

Optimal inclusion functions for complex standard functions are given in [4, 5, 11]. An alternative concept using circular arithmetic is presented in [19] and the literature cited therein. In [20], centered forms are discussed for the construction of real and complex inclusion functions (restricted to polynomials and rational functions in the complex case). The complex mean value form for analytic functions is proposed in [17].

3 Inclusion Functions for Complex Standard Functions and their Implementation in CoStLy

3.1 Design of Complex Inclusion Functions

The construction of inclusion functions in this paper is closely connected with the implementation of these functions in a software library. Unfortunately, restrictions that come with a rigorous function definition sometimes conflict with the expectation of software users that a computer program should always deliver some result. 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. Then $\sqrt{-1}$ is undefined and calling the `sqrt()` procedure with -1 must terminate execution (which in our experience annoys most users). On the other hand, if analyticity is dropped, should `sqrt(-1)` then return $+i$, $-i$, a list containing both values, or an interval containing both values? Each choice affects the mathematical definition of an appropriate inclusion function, long before the implementation.

For a multi-valued function f , various types of inclusion functions may be equally justified:

- an inclusion function F_p for some specific single-valued branch f_p of f (which usually implies restricting f to a subdomain of \mathbb{C}),
- an inclusion function F_C which is defined for all $Z \in \mathbb{C}$, but which includes function values of different branches of f , depending on the location of Z in the complex plane,
- an inclusion function F for f such that $F(Z)$ contains all function values of f on Z .

Since our goal is developing software for validated methods, we must guarantee that the range enclosures produced by our computer program are valid by all means. Hence, we will mainly develop

- inclusion functions for principal branches of multi-valued functions,
- inclusion functions enclosing all function values of a given multi-valued function, provided that the set of all values is bounded.

Our inclusion functions will usually not allow branch cuts. If Z contains a branch cut of f , then the inclusion function F is undefined on Z . Calling the CoStLy procedure for $F(Z)$ will result in program abortion. This strict concept is only abandoned for three exceptions: The argument function and root functions at the origin (where we believe that dropping analyticity is not harmful), and the logarithm for negative reals (see the discussion in Section 3.4.2).

3.2 General Aspects of the Implementation

Usually, the function $f(z) = u(x, y) + v(x, y)$ under consideration will be analytic on the given interval Z , so that the extremal values of u and v lie on the boundary of Z . For determining these values, it suffices to consider the function values in the corner points of Z and at the intersection points of the boundary of Z with extremal curves of f . These intersection points are always determined with a few

evaluations of real standard functions, such that no infinite operations like iteration or integration are involved. Throughout, the analysis is elementary but tedious. Hence we do not present the details of the respective algorithms for calculating the extremal values. They are described at great length in the CoStLy source code.

The CoStLy library for complex standard functions is based on either one of the C-XSC library or the filib++ library. Both libraries supply real and complex interval data types, the basic arithmetic operations for real and complex intervals and highly accurate inclusion functions for all real standard functions contained in S_F (but no complex standard functions). When developing inclusion functions for complex functions, we assume that (highly accurate bounds for) ranges of real standard functions are available wherever such bounds are required.

C++ allows the definition of functions and operators. Hence, the code for implementing an inclusion function is almost identical to its mathematical notation. For example,

$$\text{Rg}(\exp, X) \cdot \text{Rg}(\cos, Y) + \imath \text{Rg}(\exp, X) \cdot \text{Rg}(\sin, Y) \quad (7)$$

is an inclusion function for the complex exponential function $e^z = e^x \cos y + \imath e^x \sin y$. In the CoStLy source code for the exponential function,

```
CInterval CoStLy::exp( const CInterval& z )
{
    Interval rez = exp( z.re() ), imz = z.im();
    return CInterval( rez * cos( imz ), rez * sin( imz ) );
},
```

the ranges are replaced by the procedures `exp`, `cos`, and `sin` for real standard functions from the C-XSC or filib++ libraries. If an inclusion function contains constants such as π or $\sqrt{2}$, which are not representable exactly as floating point numbers, then these terms are enclosed by floating point intervals. Apart from these obvious modifications, the CoStLy procedures are in one-to-one agreement with the inclusion functions that are presented in the following.

3.3 Single-valued Functions

Complex functions fall into two categories: single-valued functions, such as the exponential function e^z , and multi-valued relations, like roots or logarithms. For all single-valued functions discussed in this paper, it is possible to calculate $\square \text{Rg}(f, Z)$ (and thus construct an optimal inclusion function) with finitely many basic arithmetic operations and function values of real standard 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.

3.3.1 Separable Functions

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

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

For a separable function

$$f(z) = u(x, y) + \imath v(x, y) = g_1(x) \cdot h_1(y) + \imath g_2(x) \cdot h_2(y), \quad (8)$$

the optimal inclusion function is obtained from range bounds for the real functions that appear on the right hand side of (8) as

$$F(Z) := \square \text{Rg}(f, Z) = \text{Rg}(g_1, X) \cdot \text{Rg}(h_1, Y) + i \text{Rg}(g_2, X) \cdot \text{Rg}(h_2, Y).$$

The latter formula has been implemented in CoStLy for the separable functions above, calling the respective procedures for real standard functions as supplied by the libraries C-XSC or filib++.

3.3.2 The Square Function

Similar to the separable case, the optimal inclusion function for z^2 is given by

$$Z^2 := X^2 - Y^2 + 2iX \cdot Y,$$

where

$$T^2 := \{t^2 : t \in T\} \quad \text{for } T \in \mathbb{IR} \quad (9)$$

is used. Both C-XSC and filib++ contain a procedure `sq(T)` which implements (9) for real intervals.

3.3.3 Power Function for Integer Exponents

The power function z^n is single-valued for $n \in \mathbb{Z}$. For $n = -1, 0, 1, 2$, inclusion functions for z^n are implemented as $1/Z$ (if $Z \not\cong 0$), 1 , Z , and Z^2 , respectively. In the following analysis, which appears to be new, we assume that $n \leq -2$ or $n \geq 3$ holds and that $0 \notin Z$ if $n < 0$.

The power function z^n is analytic in the entire complex plane, so that the extremal values of $u(x, y) = \text{Re}(z^n)$ and $v(x, y) = \text{Im}(z^n)$ lie on the boundary of Z . Turning to polar coordinates (r, φ) , we get

$$z^n = \underbrace{r^n \cos(n\varphi)}_{u(x,y)} + i \underbrace{r^n \sin(n\varphi)}_{v(x,y)}. \quad (10)$$

Using the chain rule, the Cauchy-Riemann equations and the well-known relations

$$r_x = \cos \varphi, \quad r_y = \sin \varphi, \quad \varphi_x = \frac{-\sin \varphi}{r}, \quad \varphi_y = \frac{\cos \varphi}{r},$$

the partial derivatives of u and v can be written as follows:

$$\begin{aligned} u_x &= nr^{n-1} \cos(n\varphi)r_x - nr^n \sin(n\varphi)\varphi_x \\ &= nr^{n-1} (\cos(n\varphi) \cos \varphi + \sin(n\varphi) \sin \varphi) = nr^{n-1} \cos((n-1)\varphi) = v_y, \\ u_y &= nr^{n-1} (\cos(n\varphi) \sin \varphi - \sin(n\varphi) \cos \varphi) = -nr^{n-1} \sin((n-1)\varphi) = -v_x. \end{aligned}$$

u_x and v_y vanish for $\varphi = \varphi_k = \frac{(2k-1)\pi}{2(n-1)}$, $k = -n+2, -n+3, \dots, n-1$, u_y and v_x vanish for $\varphi = \psi_k = \frac{k\pi}{n-1}$, $k = -n+2, -n+3, \dots, n-1$. Due to the sign changes of the partial derivatives, local extremal values of u and v occur at the intersections of horizontal or vertical boundaries of Z with the rays $\varphi = \varphi_k$ and $\varphi = \psi_k$, as illustrated in Fig. 1.

If Z does not intersect any of the above rays, then the extremal values of u and v are attained in the corner points of Z . Otherwise, for each horizontal or vertical boundary of Z , function values of u and v in at most 2 consecutive ray intersections are sufficient for determining the absolute extremal values of u and v . The locations of the relevant ray intersections depend on the sign of n and the position of Z in the complex plane. We omit the details of the elementary, but tedious procedure for determining the relevant intersection points.

Recapitulating, we have shown that the range of z^n on some interval Z can be calculated from a small number of real standard function evaluations, for arbitrary $n \in \mathbb{Z}$ (where $0 \notin Z$ is assumed if $n < 0$).

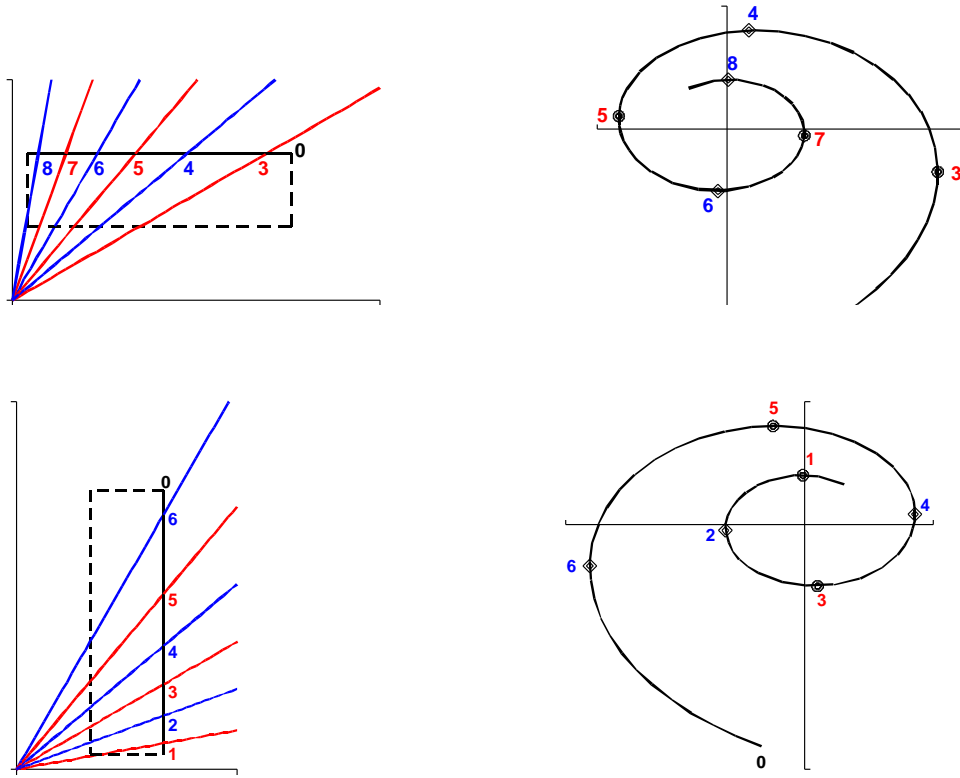


Figure 1: The image of a straight line under the mapping z^n circulates around the origin (right). The character of the extremal point at an intersection of the line with a ray $\varphi = \varphi_k$ or $\varphi = \psi_k$ is depending on the orientation of the line and on the modulus of k with respect to 4.

3.3.4 The Tangent Function

The real and imaginary parts of the tangent are given by [1, 4.3.57]

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

If Z contains no poles, then the tangent is analytic in $Z = X + iY$ and the extremal values of $u(x, y)$ and $v(x, y)$ lie on the boundary of Z . For the range of $u(x, y)$ on a vertical boundary, we have

$$\text{Rg}(u, \tilde{x} + iY) = \frac{\sin(2\tilde{x})}{\cos(2\tilde{x}) + \text{Rg}(\cosh, 2 \cdot Y)}, \quad \tilde{x} \in \{\underline{x}, \bar{x}\}. \quad (11)$$

The expression on the right hand side of (11) already includes the function values in the corner points of Z . For calculating the extremal values of u on Z , any intersections of the horizontal boundaries of Z with the extremal curves $|\tan x| = |\coth y|$ of the tangent function must be determined. For each horizontal boundary, there are at most two intersections. The corresponding extremal value is

$$\pm \frac{1}{\sinh(2\tilde{y})}, \quad \tilde{y} \in \{\underline{y}, \bar{y}\}, \quad (12)$$

which can be calculated without computing the missing coordinate of the intersection point. Details of this procedure, such as determining the correct sign in (12), were given in [4, 5].

Likewise, the range of $v(x, y)$ is bounded by enclosing the function values on the vertical boundaries as

$$\text{Rg}(v, (X, \tilde{y})) = \frac{\sinh(2\tilde{y})}{\text{Rg}(\cos, 2 \cdot X) + \cosh(2\tilde{y})}, \quad \tilde{y} \in \{\underline{y}, \bar{y}\}$$

and by checking for intersections of the extremal curves with the horizontal boundaries of Z . If such an intersection occurs then the extremal value is

$$-\frac{1}{|\sin(2\tilde{x})|}, \quad \tilde{x} \in \{\underline{x}, \bar{x}\}$$

(cf. [4, 5]). Again, the extremal value can be calculated without computing the missing coordinate of the intersection point.

3.3.5 $\cot z$, $\tanh z$, $\coth z$

These functions are defined via the complex tangent function. In particular, we have [1, 4.5.9, 4.5.12]

$$\cot z = \tan\left(\frac{\pi}{2} - z\right), \quad \tanh z = -\imath \tan(\imath z), \quad \coth z = \imath \tan\left(\frac{\pi}{2} - \imath z\right),$$

from which we obtain the inclusion functions

$$\cot Z := \tan\left(\frac{\pi}{2} - Z\right), \quad \tanh Z := -\imath \tan(\imath Z), \quad \coth Z := \imath \tan\left(\frac{\pi}{2} - \imath Z\right).$$

Because multiplication with \imath is an exact operation for floating point intervals, these functions are implemented in CoStLy by calling the procedure for \tan .

3.4 Multi-valued Functions

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

3.4.1 Argument Functions

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

$$x = r \cos \varphi, \quad y = r \sin \varphi, \tag{13}$$

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 of a complex number is not uniquely defined. If $z \in \mathbb{C}^-$, then the principal value of the argument function is the unique polar angle $\text{Arg } z \in (-\pi, \pi)$ that fulfills (13).

The argument function is often used to transform polar coordinates into Cartesian coordinates and vice versa. In this context, the value of $\arg 0$ is arbitrary, since

$$0 = 0 \cdot (\cos \varphi + \imath \sin \varphi)$$

for any $\varphi \in \mathbb{R}$. To include intervals that contain the origin, but do not intersect the negative real axis, we define the interval argument function as

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

If $0 \neq Z_1 \subseteq Z_2 \subseteq \mathbb{C}_0^-$, then $\text{Arg } Z_1 \subseteq \text{Arg } Z_2$, so that both optimality of the range enclosure and inclusion isotonicity are maintained for all intervals in the domain of $\text{Arg } Z$, save for $Z = 0$. For example, if $Z_1 = 0$, $Z_2 = \text{Arg}(0 + \imath[0, 1])$, then $Z_1 \subseteq Z_2$ holds, but $\text{Arg } Z_1 = 0 \notin \frac{\pi}{2} = \text{Arg } Z_2$.

A frequent use of the multi-valued argument function is the inclusion of a rectangular interval $Z = X + \imath Y$ into a polar interval $(R, \Phi) = ([r, \bar{r}], [\underline{\varphi}, \bar{\varphi}])$ (a sector of an annulus) such that

$$Z \subseteq (R, \Phi) \tag{14}$$

holds. The optimal inclusion (that is, the inclusion with minimal overestimation) is

$$R := \text{abs}(Z), \quad \Phi := \text{Arg } Z, \quad (15)$$

provided that $\text{Arg } Z$ is defined. This implies that Z must not intersect the negative real axes, 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 inclusion function for the multi-valued argument function $\arg Z$, that will be used in (15) instead of $\text{Arg } Z$.

For minimal overestimation in (14), the following properties are required:

- (i) $Z \subseteq (\text{abs}(Z), \arg Z)$ for all $Z \in \mathbb{IC}$
- (ii) for every $Z \in \mathbb{IC}$, there is no interval Φ such that

$$Z \subseteq (\text{abs}(Z), \Phi) \quad \text{and} \quad w(\Phi) < w(\arg Z).$$

(i) and (ii) are fulfilled for the following inclusion function for the argument function.

$$\arg Z = \begin{cases} 0, & Z = 0, \\ \pi, & \underline{x} < 0, \bar{x} \leq 0, \underline{y} = \bar{y} = 0, \\ [0, \pi], & \underline{x} < 0 < \bar{x}, \underline{y} = \bar{y} = 0, \\ \square\{\widetilde{\arg} z : z \in Z\}, & \underline{x} < 0, \bar{x} \leq 0, \underline{y} < 0 < \bar{y}, \\ \square\{\text{Arg } z : z \in Z \cap C^-\}, & \text{otherwise,} \end{cases}$$

where $Z = [\underline{x}, \bar{x}] + i[\underline{y}, \bar{y}]$ and

$$\widetilde{\arg} z := \begin{cases} \text{Arg } z, & \text{Re } z \leq 0, \text{Im } z > 0, \\ \pi, & \text{Re } z \leq 0, \text{Im } z = 0, \\ \text{Arg } z + 2\pi, & \text{Re } z \leq 0, \text{Im } z < 0. \end{cases}$$

$\arg Z$ has the following properties:

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

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

then we have

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

Even though $Z_1 \subset Z_2$ holds, the function values computed by the inclusion function $\arg Z$ have an empty intersection.

- The optimal inclusion of a rectangular interval into a polar interval is given by

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

3.4.2 Natural Logarithm

The principal branch of the natural logarithm is given by

$$\text{Ln } z = \ln_{\mathbb{R}} |z| + \imath \text{Arg } z,$$

where $\ln_{\mathbb{R}}$ denotes the real natural logarithm. For $Z \subset \mathbb{C}^-$, we define $\text{Ln } Z$ as

$$\text{Ln } Z := \ln(\text{abs}(Z)) + \imath \text{Arg } Z = \square\{\text{Ln } z : z \in Z\}.$$

The corresponding CoStLy subroutine throws an exception (which provokes program termination, unless it is caught by the calling procedure) when it is called with an argument Z that intersects the negative real axes.

An inclusion function for the logarithm that is also defined for negative reals is implemented in CoStLy as

$$\ln Z := \ln(\text{abs}(Z)) + \imath \arg Z.$$

A warning to potential users of this procedure seems appropriate here. Calling $\ln(Z)$ for a list of intervals $Z_k \not\ni 0$ that cover the unit circle will return an interval W_k for each Z_k . Thus CoStLy could be abused to calculate an upper bound on $|\ln z|$ on the unit circle, from which it is a small step to falsely conclude (referring to Cauchy's estimate) that the logarithm would be analytic in the unit disc. If $\text{Ln}(Z)$ is used instead of $\ln(Z)$ this misconception is averted, because at least one interval Z_k in the list contains negative real numbers, for which the program will abort execution and issue a warning message. Hence we advise 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.

3.4.3 Root Functions

The principal branch of the square root function \sqrt{z} is defined as

$$\text{Sqrt } z = e^{\frac{1}{2}\text{Ln } z}, \quad z \in \mathbb{C}^-.$$

Since $\lim_{\substack{z \rightarrow 0 \\ z \in \mathbb{C}^-}} \text{Sqrt } z = 0$, a single-valued continuous square root function on \mathbb{C}_0^- is obtained by letting

$\text{Sqrt } 0 := 0$. We define the interval function $\text{Sqrt } Z$ as follows:

$$\text{Sqrt } Z = \begin{cases} \text{Sqrt } z, & Z = z \in \mathbb{C}^-, \\ 0, & Z = 0, \\ \square\{\text{Sqrt } z : z \in Z\}, & Z \subset \mathbb{C}_0^-, \\ \text{undefined}, & Z \cap \mathbb{R}^- \neq \emptyset. \end{cases}$$

$\text{Sqrt}(Z, n)$, the inclusion function for the principal branch of the n -th root function,

$$\text{Sqrt}(z, n) = e^{\frac{1}{n}\text{Ln } z}, \quad n \in \mathbb{N}, \quad z \in \mathbb{C}^-,$$

is defined in the same manner.

Turning to polar coordinates as in the analysis of the power function, we have

$$z^{\frac{1}{n}} = \underbrace{r^{\frac{1}{n}} \cos\left(\frac{\varphi}{n}\right)}_{u(x,y)} + \imath \underbrace{r^{\frac{1}{n}} \sin\left(\frac{\varphi}{n}\right)}_{v(x,y)},$$

so that

$$\begin{aligned} u_x &= \frac{1}{n} r^{\frac{1}{n}-1} \cos\left(\frac{\varphi}{n}\right) r_x - \frac{1}{n} r^{\frac{1}{n}} \sin\left(\frac{\varphi}{n}\right) \varphi_x \\ &= \frac{1}{n} r^{\frac{1}{n}-1} \left(\cos\left(\frac{\varphi}{n}\right) \cos \varphi + \sin\left(\frac{\varphi}{n}\right) \sin \varphi \right) = \frac{1}{n} r^{\frac{1}{n}-1} \cos\left(\left(1 - \frac{1}{n}\right)\varphi\right) = v_y, \\ u_y &= \frac{1}{n} r^{\frac{1}{n}-1} \sin\left(\left(1 - \frac{1}{n}\right)\varphi\right) = -v_x. \end{aligned}$$

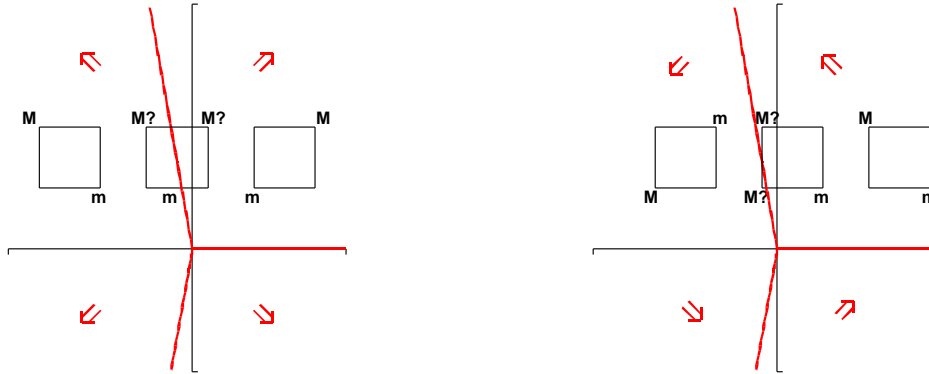


Figure 2: Monotonicity properties of $u(x, y) = \operatorname{Re} \sqrt[n]{z}$ (left) and $v(x, y) = \operatorname{Im} \sqrt[n]{z}$ (right). The arrows indicate the directions in which u and v increase. For example, the arrow pointing to the upward right in the left picture shows that $u(x, y)$ increases strictly with x and y in the region $0 \leq \varphi \leq \frac{n\pi}{2n-2}$. M and m denote the locations of the global maximum and minimum of u and v on the respective interval.

u_x and v_y vanish for $\varphi = \pm \frac{n\pi}{2n-2}$; u_y and v_x vanish for $\varphi = 0$. If Z intersects neither of the rays $\varphi = \pm \frac{n\pi}{2n-2}$ nor the positive real axes, then the extremal values of u and v lie in the corner points of Z . Otherwise, intersections of the boundary of Z with these rays must be considered as well. The monotonicity properties of u and v are used to determine the locations of the extremal values of u and v for a given interval Z , as shown in Fig. 2.

While $\operatorname{Sqrt}(Z, n)$ calculates precise range bounds with only a few real function evaluations, it has the disadvantage that it is undefined if Z contains negative real numbers. For applications where roots of negative real numbers occur, we implemented an alternative n -th root function in CoStLy. To maintain a precise function definition, it is implemented as a set function $\operatorname{sqrt_all}(Z, n)$, which computes a list of n intervals that contain all n -th roots of Z :

$$\operatorname{sqrt_all}(Z, n) \supseteq \{w : w^n = z, z \in Z\}.$$

For a sufficiently small interval $Z \in \mathbb{C} - \{0\}$, the optimal interval enclosure for all n -th roots consists of n distinct intervals. While these could be computed using arguments as in the calculation of the principal value $\operatorname{Sqrt}(Z, n)$, such a procedure would be computationally expensive, especially for large values of n . Hence, we used the following strategy: Z is enclosed into the polar interval $(R, \Phi) = ([\underline{r}, \bar{r}], [\underline{\varphi}, \bar{\varphi}]) = (\operatorname{abs}(Z), \operatorname{arg}(Z))$. All n -th roots of Z are then contained in the union $\cup_{k=0}^{n-1} (R_k, \Phi_k)$, where

$$R_k = [\underline{r}^{(1/n)}, \bar{r}^{(1/n)}], \quad \Phi_k = (\Phi + 2k\pi)/n, \quad k = 0, 1, \dots, n-1.$$

Finally, (R_k, Φ_k) is enclosed into a rectangular interval W_k , such that $\operatorname{sqrt_all}(Z, n) := \cup_{k=0}^{n-1} W_k$ (Fig. 3).

3.5 Power Functions for Real and Complex Exponents

For $z \neq 0$, z^p is defined as the set of all numbers

$$z^p := e^{p \ln z} = e^{p \operatorname{Ln} z + i p \varphi}, \quad \varphi = \operatorname{arg} z = \operatorname{Arg} z + 2k\pi, \quad k \in \mathbb{Z}.$$

If $z \in \mathbb{C}^-$, then the principal value of z^p is $e^{p \operatorname{Ln} z}$. The inclusion function for the principal value of z^p is implemented in CoStLy as

$$\operatorname{Pow}(Z, P) := \exp(P \cdot \operatorname{Ln}(Z)) \supseteq \square\{e^{p \operatorname{Ln} z} : z \in Z, p \in P\},$$

where Z and P may be complex intervals. The overestimation is moderate for small Z and P .

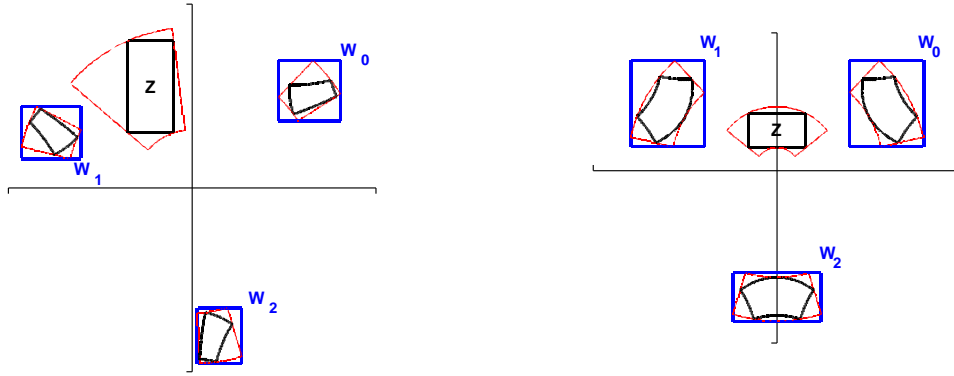


Figure 3: Inclusion sets for $\text{sqrt_all}(Z, 3)$ for two different intervals Z .

Additionally, a procedure called $\text{pow_all}(Z, P)$ for enclosing the set

$$S := \{e^{p \ln z} : z \in Z, p \in P\}$$

of *all* powers is implemented in CoStLy. Z may be complex, but P must be a real interval, because $e^{p \ln z}$ is unbounded if p is an exponent with a nonzero imaginary part. Three cases are considered for the computation of $\text{pow_all}(Z, P)$, $P = [p, \bar{p}]$.

Case 1: $0 \notin Z$.

If $p \in \mathbb{R} - \mathbb{Q}$, $z \neq 0$ and $\varphi = \arg(z)$, then $z^p = e^{p \ln |z|} \cdot e^{p i \varphi}$ represents infinitely many numbers on the circle with radius $e^{p \ln |z|}$ around the origin. If $Z \in \mathbb{IC}$, $0 \notin Z$ and $P \in \mathbb{IR}$ then all solutions

$$e^{p \ln |z|} \cdot e^{p i \varphi}, \quad z \in Z, p \in P$$

lie on an annulus which is determined by two radii, the inner radius r_1 and the outer radius r_2 :

$$r_1 = e^{\inf(P \ln |Z|)}, \quad r_2 = e^{\sup(P \ln |Z|)}.$$

Unfortunately, rectangular intervals are rather unsuitable for enclosing an annulus. It could be enclosed into the circumscribed square, but

- if $r_2 - r_1$ is small then it is desirable to have an enclosure of the solution set with less overestimation,
- for subsequent calculations it may be advantageous for the origin to be excluded from the interval enclosure of the annulus.

For these reasons, $\text{pow_all}(Z, P)$ is implemented in CoStLy as follows: The annulus is enclosed into the union of four distinct rectangular complex intervals. The border lengths of the inscribed and of the circumscribed squares are

$$d_1 = \frac{\sqrt{2}}{2} r_1, \quad d_2 = r_2.$$

These values are used to compute four complex intervals R_1, R_2, R_3 and R_4 each covering a quarter of the annulus (Fig. 4).

Remark: Special cases such as $P \in \mathbb{Z}$ or $P \in \mathbb{Q}$ are not handled separately by the CoStLy procedure for $\text{pow_all}(Z, P)$. For example, $1^2 = 1$, but $\text{pow_all}(1, 2)$ returns a list of four rectangles covering the unit circle.

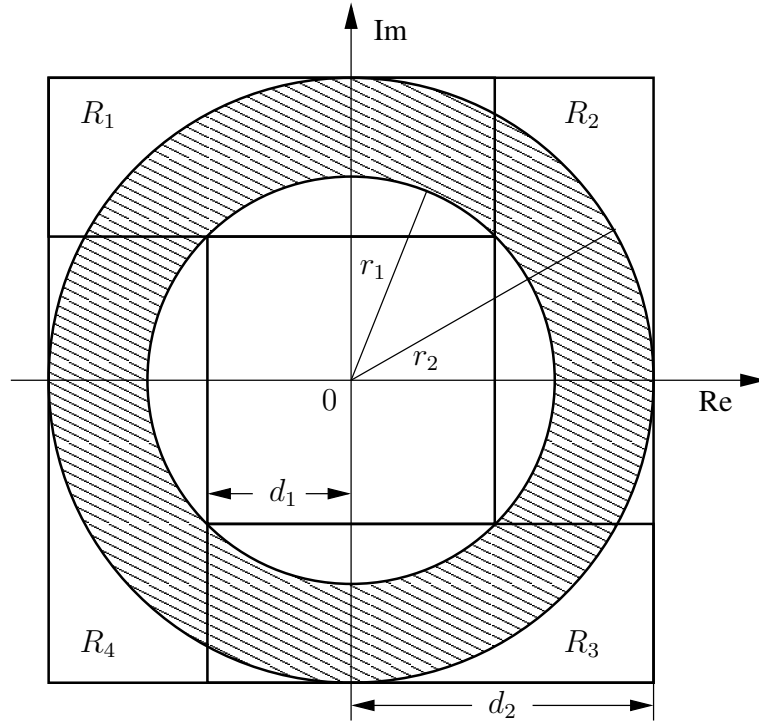


Figure 4: Inclusion of an annulus in 4 intervals.

Case 2: $0 \in Z$, $p > 0$.

The annulus of Case 1 is now a disc (which shrinks to the origin if $Z = 0$). `pow_all(Z, P)` computes the circumscribed square of this disc.

Case 3: $0 \in Z$, $p \leq 0$.

0^p is undefined for $p \leq 0$. The CoStLy procedure for `pow_all(Z, P)` throws an exception if $0 \in Z$ and $p \leq 0$ hold.

3.6 Inverse Trigonometric and Inverse Hyperbolic Functions

If f is any trigonometric or hyperbolic standard function in S_F and z a complex number in the range of f , then the set

$$\{w : f(w) = z\}$$

is unbounded. Hence it is impossible to enclose *all* values $f^{-1}(z)$ into a compact interval. For this reason, the inverse trigonometric or inverse hyperbolic functions are implemented as single-valued principal values on proper subsets of \mathbb{C} .

3.6.1 Asin

The image of the strip $S : -\frac{\pi}{2} < \operatorname{Re} z < \frac{\pi}{2}$ under the sine function is $R_S := \mathbb{C} - \{(-\infty, -1] \cup [1, \infty)\}$. The sine function is one to one in S . If possible, we would like to enlarge the domain of our inverse sine function beyond R_S . On the boundary of S , we have

$$\sin\left(-\frac{\pi}{2} + iy\right) = -\cosh y, \quad \sin\left(\frac{\pi}{2} + iy\right) = \cosh y, \quad (16)$$

from which we derive the bijection

$$\sin : S \cup \left\{-\frac{\pi}{2}, \frac{\pi}{2}\right\} \rightarrow R_S \cup \{-1, 1\}$$

with its continuous inverse function

$$\text{Asin} : R_S \cup \{-1, 1\} \rightarrow S \cup \{-\frac{\pi}{2}, \frac{\pi}{2}\}.$$

Because of

$$\sin(\pm\frac{\pi}{2} + iy) = \sin(\pm\frac{\pi}{2} - iy),$$

there is no continuous inverse sine function on a larger domain.

For a given interval $Z \subset R_S \cup \{-1, 1\}$, we define the inclusion function

$$\text{Asin } Z := \square\{\text{Asin } z : z \in Z\}.$$

It is implemented in CoStLy as discussed in [11]. Apart from small roundoff errors, which are enclosed in the result, the precise range bounds are calculated. INTLAB, for example, uses the logarithmic representation of Asin given by

$$\text{Asin } z = -i\text{Ln}(iz + \text{Sqrt}(1 - z^2)), \quad z \in \mathbb{C} - \{(-\infty, -1) \cup (1, \infty)\}$$

to compute Asin Z as

$$\text{Asin } Z := -i\text{Ln}(iZ + \text{Sqrt}(1 - Z^2)),$$

which shows considerable overestimation unless Z is a small interval.

3.6.2 Acos

The inverse cosine functions follows from the identity $\cos(z) = -\sin(z - \frac{\pi}{2})$, which yields

$$\text{Acos } z = \frac{\pi}{2} - \text{Asin } z.$$

Acos z maps $\mathbb{C} - \{(-\infty, -1) \cup (1, \infty)\}$ onto $0 < \text{Re } z < \pi \cup \{0, \pi\}$. Its inclusion function is implemented in CoStLy as

$$\text{Acos } Z = \frac{\pi}{2} - \text{Asin } Z.$$

3.6.3 Asinh

The function

$$\sinh z = \sinh x \cos y + i \cosh x \sin y$$

is one to one in the horizontal strip $T : -\frac{\pi}{2} < \text{Im } z < \frac{\pi}{2}$. Its boundary values are $\pm i \cosh x$ for $z = x \pm i\frac{\pi}{2}$. The range of $\sinh z$ in T is $R_T := \mathbb{C} - \{(-\infty, -1]i \cup [1, \infty)i\}$. The domain of a continuous inverse hyperbolic sine function can be extended beyond R_T by including the points $\pm i$. Thus we obtain

$$\text{Asinh} : R_T \cup \{-i, i\} \rightarrow T \cup \{-\frac{\pi}{2}i, \frac{\pi}{2}i\}.$$

The corresponding inclusion function is implemented in CoStLy as

$$\text{Asinh } Z = i\text{Asin}(-iZ).$$

3.6.4 Acosh

The construction of an inclusion functions for the inverse hyperbolic cosine is more involved. Using $\cosh(iz) = \cos z$, a single-valued branch of the inverse hyperbolic cosine is given by

$$\text{acosh } z := i\text{Acos } z = i(\frac{\pi}{2} - \text{Asin}(iz)),$$

mapping (most of) the complex plane to the strip $0 < \text{Im } z < \pi$. However, continuity implies that the intervals $(-\infty, 1]$ and $[1, \infty)$ cannot be included in the domain of acosh, since

$$\cosh x = \cosh(-x) = \cosh(x + i\pi) = \cosh(-x + i\pi).$$

To retain the range of the real hyperbolic cosine in the domain of the inverse function, we use the symmetry condition $\cosh(-z) = \cosh z$ to map $\mathbb{C} - (-\infty, 1)$ via $\text{Acos } z$ onto the union of the origin and the semi-strip $\text{Re } z > 0, -\pi < \text{Im } z < \pi$. Thus we obtain the principal value

$$\text{Acosh } z := \begin{cases} \imath \text{Acos } z, & \text{Im}(\text{Acos } z) < 0, \\ \text{acosh}(x), & \text{Im}(z) = 0, \\ -\imath \text{Acos } z, & \text{Im}(\text{Acos } z) > 0. \end{cases}$$

For $Z \subset \mathbb{C} - (-\infty, 1)$, the inclusion function for the inverse cosine is implemented in CoStLy as

$$\text{Acosh } Z := \square\{\text{Acosh } z : z \in Z\}.$$

3.6.5 Atan

The principal branch of the inverse tangent function is

$$\text{Atan } z = \frac{1}{2\imath} \text{Ln} \frac{1 + \imath z}{1 - \imath z}, \quad z \in \mathbb{C} - \{\imath y : |y| \geq 1\}.$$

Calculating sharp range bounds for Atan was discussed in [5, 11] by considering continuation at branch cuts. The analysis is simplified considerably when the inclusion function is restricted to a principal value. We refer to the CoStLy source code for the details.

The inclusion function for Atan is implemented in CoStLy as

$$\text{Atan } Z := \square\{\text{Atan } z : z \in Z\}.$$

The obtained enclosures are generally much more accurate than calculating

$$\frac{1}{2\imath} \text{Ln} \frac{1 + \imath Z}{1 - \imath Z},$$

as it is for example done in INTLAB.

3.6.6 Acot, Atanh, Acoth

For $z \in \mathbb{C} - \{\imath y : |y| \leq 1\}$, the principal value of Acot is

$$\text{Acot } z := \text{Atan} \frac{1}{z}.$$

Its inclusion function is implemented in CoStLy as

$$\text{Acot } Z := \square\{\text{Atan} \frac{1}{z} : z \in Z\}.$$

Even though the code of the Acot procedure is very similar to the code of Atan , it has been implemented independently. Merely calling $\text{Atan}(1/Z)$ would have simplified the programming, but also would have caused considerable overestimation in the result.

Because multiplication with \imath is an exact operation for floating point intervals, the functions

$$\text{Atanh } z = -\imath \text{Atan}(\imath z)$$

and

$$\text{Acoth } z = \imath \text{Acot } z$$

have been implemented as

$$\text{Atanh } Z := -\imath \text{Atan}(\imath Z), \quad \text{Acoth } Z := \imath \text{Acot } Z.$$

4 Numerical Examples

4.1 The CoStLy C++ Class Library

CoStLy has been developed as a new C++ library for the validated computation of function values and of ranges of complex standard functions. It requires one of the following libraries for the validated computation of real functions: C-XSC [9] or filib++ [13, 14]. The C-XSC library is more comprehensive than filib++, but the latter is faster. Because filib++ does not contain the complex interval data type, this data type is also implemented in CoStLy (see files `cinterval.h` and `cinterval.cpp`).

The libraries C-XSC and filib++ are distributed under the terms of the GNU Library General Public License; CoStLy is distributed under the terms of the GNU General Public License. The software is currently available at the following sites:

C-XSC: <http://www.xsc.de>

filib++: <http://www.math.uni-wuppertal.de/wrswt/software/filib.html>

CoStLy: <http://www.uni-karlsruhe.de/~Markus.Neher/CoStLy.html>

Readme files for the installation of the software are also available on the respective websites. At the moment, C-XSC supports the following platforms:

PC with Linux and GNU C++ compiler gcc 2.95.2/2.95.3/3.2/3.3/3.3.2,

PC with Linux and Intel C++ compiler 7.1 and 8.0

SUN Solaris workstation with SUN Forte Developer 7 C++ 5.4

SUN Solaris workstation with GNU C++ compiler gcc 2.95.2/2.95.3/3.2/3.3/3.3.2

DEC alpha with Linux and GNU C++ compiler gcc 2.95.2

filib++ requires one of the GNU C++ compilers gcc 2.95.2 through 3.3.2, or the KAI C++ compiler. The filib++ macro library (which is used by CoStLy) is only supported on ix86 systems and requires the use of GNU make.

Disclaimer: CoStLy has been extensively tested and has been found to be reliable and accurate. Of course, even though it is software for validated computations, it is subject to the same potential errors as conventional software. The program is distributed in the hope that it will be useful, but without any warranty.

4.2 Numerical Examples

In the following, we show tables with selected function values for various functions contained in CoStLy. The current version of the C-XSC library was used in the calculations. The results from CoStLy are compared with those from INTLAB [22]. Both packages contain complex inclusion functions, but INTLAB and CoStLy have been developed for different purposes. INTLAB is a comprehensive and powerful Matlab toolbox with a focus on reliable and fast linear algebra routines. While most INTLAB procedures are highly accurate, the complex inclusion functions are implemented rather economically, without striving to compute optimal range bounds. Hence, it is no surprise that the range bounds computed with the specialized CoStLy library are usually better than the range bounds obtained from INTLAB.

INTLAB uses circular intervals (denoted by $\langle m, r \rangle$ in the following, where m is the midpoint of a disc and r its radius), so that a direct comparison of the results is not possible. We inserted square intervals as arguments for CoStLy in the test cases, to approximate the INTLAB discs. If $D = \langle m, r \rangle$ was used as argument in an INTLAB procedure, then the circumscribed square $Z = [\text{Re } m - r, \text{Re } m + r] + i[\text{Im } m - r, \text{Im } m + r]$ served as argument of the corresponding CoStLy function. For better readability, the results obtained with INTLAB are shown in inf/sup representation rather than in midpoint/radius notation.

f	INTLAB V5	CoStLy 1.0
exp	$[-5.97+4, 1.61+4] + \imath [-3.48+4, 4.10+4]$	$[-5.99+4, -3.37+3] + \imath [-4.54+4, 5.45+4]$
cos	$[-2.58+1, 8.86+0] + \imath [-1.19+1, 2.28+1]$	$[-2.74+1, 1.21-1] + \imath [-1.13+1, 2.73+1]$
sin	$[-2.28+1, 1.19+1] + \imath [-2.58+1, 8.90+0]$	$[-2.74+1, 1.13+1] + \imath [-2.73+1, 1.21-1]$
tan	$[\text{NaN}, \text{NaN}] + \imath [\text{NaN}, \text{NaN}]$	$[-2.69-2, 3.67-2] + \imath [9.64-1, 1.04+0]$
cosh	$[-2.99+4, 8.03+3] + \imath [-1.74+4, 2.05+4]$	$[-3.00+4, -1.68+3] + \imath [-2.27+4, 2.73+4]$
sinh	$[-2.99+4, 8.03+3] + \imath [-1.74+4, 2.05+4]$	$[-3.00+4, -1.68+3] + \imath [-2.27+4, 2.73+4]$
tanh	$[\text{NaN}, \text{NaN}] + \imath [\text{NaN}, \text{NaN}]$	$[9.99-1, 1.01+0] + \imath [-3.05-8, 3.05-8]$
ln	$[2.24+0, 2.45+0] + \imath [1.90-1, 3.93-1]$	$[2.22+0, 2.46+0] + \imath [1.79-1, 4.19-1]$
acos	$[1.91-1, 3.94-1] + \imath [-3.14+0, -2.90+1]$	$[1.80-1, 4.21-1] + \imath [-3.16+0, -2.91+0]$
asin	$[\text{NaN}, \text{NaN}] + \imath [\text{NaN}, \text{NaN}]$	$[1.15+0, 1.40+0] + \imath [2.91+0, 3.16+0]$
atan	$[1.46+0, 1.49+0] + \imath [1.81-2, 3.68-2]$	$[1.46+0, 1.50+0] + \imath [1.58-2, 4.10-2]$
acosh	$[2.93+0, 3.14+0] + \imath [1.91-1, 3.94-1]$	$[2.91+0, 3.16+0] + \imath [1.80-1, 4.21-1]$
asinh	$[2.93+0, 3.15+0] + \imath [1.84-1, 3.97-1]$	$[2.91+0, 3.16+0] + \imath [1.79-1, 4.17-1]$
asinh	$[-2.99+4, 8.03+3] + \imath [-1.74+4, 2.05+1]$	$[-3.00+4, -1.68+3] + \imath [-2.27+4, 2.73+4]$
atanh	$[8.34-2, 1.03-1] + \imath [1.53+0, 1.56+0]$	$[8.03-2, 1.07-1] + \imath [1.52+0, 1.56+0]$

Table 1: Function values for selected inclusion functions for
 $D = \langle 10 + 3\imath, 1 \rangle$ (INTLAB) vs. $Z = [9, 11] + \imath[2, 4]$ (CoStLy).

Example 1: Function values for $D = \langle 10 + 3\imath, 1 \rangle$, $Z = [9, 11] + \imath[2, 4]$.

Apart from roundoff errors, which do not affect the accuracy of the calculated bounds much, CoStLy computes the optimal range bounds for all functions tested. In contrast, some of the INTLAB functions suffer from overestimation. For example, the hyperbolic tangent is calculated in Intlab as

$$\tanh(D) := \frac{\sinh(D)}{\cosh(D)},$$

which is undefined here, because 0 is contained in the range bound that is calculated for $W_{\cosh}(D)$. The CoStLy evaluation of $\tanh(Z)$ shows, however, that \tanh is almost a constant function in D .

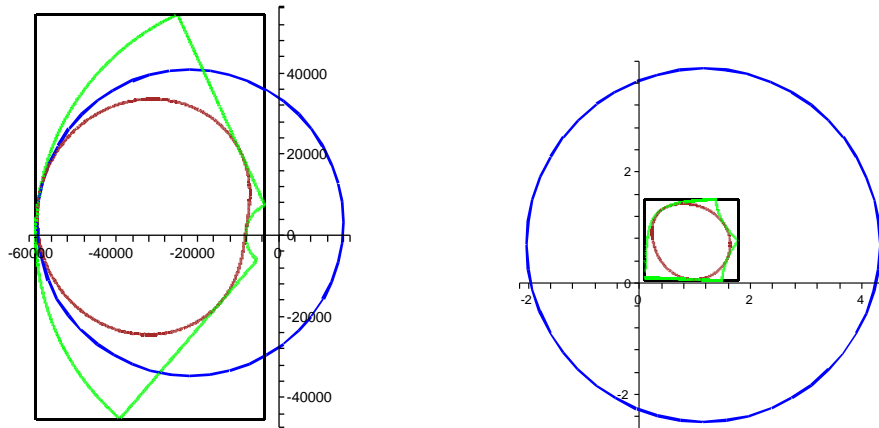


Figure 5: Ranges of the exponential function for $D = \langle 10 + 3\imath, 1 \rangle$, $Z = [9, 11] + \imath[2, 4]$ (left) and of asinh for $D = \langle 1.1 + 1.1\imath, 1 \rangle$, $Z = [0.1, 2.1] + \imath[0.1, 2.1]$ (right).

The kidney shaped sets are the respective ranges on D , the circles are the range bounds calculated by INTLAB. The remaining sets are the ranges on Z and the enclosing intervals as computed by CoStLy.

f	INTLAB V5	CoStLy 1.0
exp	$[-3.80+0, 6.53+0] + i[-2.49+0, 7.84+0]$	$[-4.13+0, 8.13+0] + i[1.10-1, 8.17+0]$
cos	$[-2.12+0, 3.63+0] + i[-4.06+0, 1.68+0]$	$[-2.10+0, 4.13+0] + i[-4.03+0, -9.99-3]$
sin	$[-1.38+0, 4.36+0] + i[-2.27+0, 3.48+0]$	$[1.00-1, 4.15+0] + i[-2.04+0, 4.01+0]$
tan	$[NaN, NaN] + i[NaN, NaN]$	$[-4.97+0, 4.97+0] + i[1.00-1, 1.01+1]$
cosh	$[-2.12+0, 3.63+0] + i[-1.68+0, 4.06+0]$	$[-2.10+0, 4.13+0] + i[9.99-3, 4.03+0]$
sinh	$[-2.27+0, 3.48+0] + i[-1.38+0, 4.36+0]$	$[-2.04+0, 4.01+0] + i[1.00-1, 4.15+0]$
tanh	$[NaN, NaN] + i[NaN, NaN]$	$[1.00-1, 1.01+1] + i[-4.97+0, 4.97+0]$
ln	$[-5.88-1, 1.48+0] + i[-2.45-1, 1.82+0]$	$[-1.96+0, 1.09+0] + i[4.75-2, 1.53+0]$
acos	$[-2.62-1, 2.04+0] + i[-2.30+0, -3.67-3]$	$[5.40-2, 1.53+0] + i[-1.79+0, -1.00-1]$
asin	$[NaN, NaN] + i[NaN, NaN]$	$[4.29-2, 1.52+0] + i[1.00-1, 1.79+0]$
atan	$[-2.38-1, 2.00+0] + i[-7.56-1, 1.48+0]$	$[1.00-1, 1.55+0] + i[1.84-2, 1.50+0]$
acosh	$[3.67-3, 2.30+0] + i[-2.62-1, 2.04+0]$	$[1.00-1, 1.79+0] + i[5.40-2, 1.53+0]$
asinh	$[-2.04+0, 4.34+0] + i[-2.50+0, 3.87+0]$	$[1.00-1, 1.79+0] + i[4.29-2, 1.52+0]$
atanh	$[4.30-2, 2.28+0] + i[-2.38-1, 2.00+0]$	$[1.84-2, 1.50+0] + i[1.00-1, 1.55+0]$

Table 2: Function values for selected inclusion functions for $D = \langle 1.1 + 1.1i, 1 \rangle$ (INTLAB) vs. $Z = [0.1, 2.1] + i[0.1, 2.1]$ (CoStLy).

Of all functions tested in INTLAB, the least amount of overestimation was observed for the exponential function and the logarithm. These are implemented using Taylor's circular centered forms [19, Chap. 2.2]. Nevertheless, even for the exponential function, the circle that is computed with INTLAB is not the smallest circle containing the range $W_{\exp}(D)$ (Fig. 5, left).

Example 2: Function values for $D = \langle 1.1 + 1.1i, 1 \rangle$, $Z = [0.1, 2.1] + i[0.1, 2.1]$.

Some of the functions considered have singularities at ± 1 or $\pm i$, and the boundaries of D and Z are close to these points. While CoStLy still calculates optimal bounds (except for negligible roundoff), some of the INTLAB routines show large overestimation of the result (which usually increases the closer D gets to a singularity). Observe that since $D \subset Z$, the range $W_f(D)$ is always contained in the range $W_f(Z)$, but the calculated range bounds do not follow this rule (Fig. 5, right).

Conclusion

We have presented optimal inclusion functions for the usual set of complex standard functions. These inclusion functions have been implemented in a new interval library for computing highly accurate range bounds. For some of the functions, several versions have been implemented to satisfy different needs in applications.

Future research will concentrate on the extension of the CoStLy library through the implementation of additional complex inclusion functions.

Acknowledgement

The author would like to thank Ingo Eble for help with the programming of the CoStLy library.

References

- [1] M. Abramowitz and I. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. National Bureau of Standards, Washington, 1964.
- [2] G. Alefeld and J. Herzberger. *Introduction to interval computations*. Academic Press, New York, 1983.
- [3] M. Berz. Cosy Infinity Version 8 reference manual. NSCL Technical Report MSUCL-1088, Michigan State University, 1998.
- [4] K. Braune. Standard functions for real and complex point and interval arguments with dynamic accuracy. *Computing Supplementum*, 6:159–184, 1988.
- [5] K. Braune and W. Krämer. High-accuracy standard functions for real and complex intervals. In E. Kaucher, U. Kulisch, and Ch. Ullrich, editors, *Computerarithmetic: Scientific computation and programming languages*, pages 81–114. Teubner, Stuttgart, 1987.
- [6] G. Bühler. *Standardfunktionen für komplexe Intervalle im 64 Bit IEEE Datenformat*. Diploma thesis, Universität Karlsruhe, 1993.
- [7] M. Grimmer. Interval arithmetic in Maple with intpakX. *PAMM*, 2:442–443, 2003.
- [8] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied interval analysis*. Springer, London, 2001.
- [9] R. Klatte, U. Kulisch, Ch. Lawo, M. Rauch, and A. Wiethoff. *C-XSC: A C++ class library for extended scientific computing*. Springer, Berlin, 1993.
- [10] R. Klatte, U. Kulisch, M. Neaga, D. Ratz, and Ch. Ullrich. *PASCAL-XSC – Language reference with examples*. Springer, Berlin, 1992.
- [11] W. Krämer. Inverse standard functions for real and complex point and interval arguments with dynamic accuracy. *Computing Supplementum*, 6:185–212, 1988.
- [12] U. Kulisch and W. L. Miranker. *Computer arithmetic in theory and practice*. Academic Press, New York, 1981.
- [13] M. Lerch, G. Tischler, and J. Wolff von Gudenberg. filib++ - Interval library specification and reference manual. Technical Report 279, Universität Würzburg, 2001.
- [14] M. Lerch, G. Tischler, J. Wolff von Gudenberg, W. Hofschuster, and W. Krämer. The interval library filib++ 2.0. Design, features and sample programs. Preprint 2001/4, Universität Wuppertal, Wissenschaftliches Rechnen/Softwaretechnologie, 2001.
- [15] R. Lohner and J. Wolff von Gudenberg. Complex interval division with maximum accuracy. In *Proc. 7th IEEE Symp. on Computer Arithmetic (ARITH 7) (Urbana, Illinois, 1985)*, pages 332–336, 1985.
- [16] R. E. Moore. *Interval analysis*. Prentice Hall, Englewood Cliffs, N.J., 1966.
- [17] M. Neher. The mean value form for complex analytic functions. *Computing*, 67:255–268, 2001.
- [18] A. Neumaier. *Interval methods for systems of equations*. Cambridge University Press, Cambridge, 1990.
- [19] M. S. Petković and L. D. Petković. *Complex interval arithmetic and its applications*. Wiley-VCH, Berlin, 1998.
- [20] H. Ratschek and J. Rokne. *Computer methods for the range of functions*. Ellis Horwood Limited, Chichester, 1984.
- [21] N. Revol and F. Rouillier. Motivations for an arbitrary precision interval arithmetic and the MPFI library. Research Report 2002-27, LIP, École Normale Supérieure de Lyon, France, 2002.
- [22] S. Rump. INTLAB – INTerval LABoratory. In T. Csendes, editor, *Developments in reliable computing*, pages 77–104. Kluwer, Dordrecht, 1999.

- [23] W. V. Walter. Fortran-XSC: A portable Fortran 90 module library for accurate and reliable scientific computing. In R. Albrecht, G. Alefeld, and H. J. Stetter, editors, *Validation Numerics – Theory and Applications*, pages 265–285. Computing Supplementum 9, 1993.
- [24] XSC website on programming languages for scientific computing with validation. <http://www.xsc.de> [October 2004].