Function evaluation on branch cuts

Albert D. Rich and David J. Jeffrey
Soft Warehouse, Honolulu, Hawaii
Dept Applied Maths, U.W.O., London, Canada

SIGSAM Bulletin, June 1996, Issue 116, pp 25–27

1 Introduction

Once it is decided that a CAS will evaluate multivalued functions on their principal branches, questions arise concerning the branch definitions. The first questions concern the standardization of the positions of the branch cuts. These questions have largely been resolved between the various algebra systems and the numerical libraries, although not completely. In contrast to the computer systems, many mathematical textbooks are much further behind: for example, many popular textbooks still specify that the argument of a complex number lies between 0 and 2π. We do not intend to discuss these first questions here, however. Once the positions of the branch cuts have been fixed, a second set of questions arises concerning the evaluation of functions on their branch cuts.

In [2], Kahan considered the closure problem from several points of view and discussed different possible solutions. One of his proposals was a principle called counter clockwise continuity (CCC) for the determination of the closure of the elementary functions. To determine closure for any branch, one imagines circling the branch point counterclockwise (anticlockwise in the British hemisphere) and the closure is on the side one arrives at by this process. Thus, one decides $\arctan(5i/3) = \frac{1}{2}\pi + i\ln 2$ and $\arcsin(5/4) = \frac{1}{2}\pi - i\ln 2$. This convention has not been followed by all systems. In particular, Derive defines inverse tangent using clockwise continuity (CC), and therefore obtains $\arctan(5i/3) = -\frac{1}{2}\pi + i\ln 2$. There are many other cases within the current CAS of non-CCC closures.

Examining the reasoning behind the Derive selection of the closure of arctangent introduces the reasons for many of the departures from CCC in other systems. If both arcsine and arctangent are closed CCC, then the relation between them that is valid over the whole of $\mathbb{C}$ is

$$\begin{aligned}\arcsin z &= \arctan\frac{z}{\sqrt{1-z^2}} - \pi\mathcal{K}(-\ln(1+z)) \\ &\quad +\pi\mathcal{K}(-\ln(1-z)) \ , \qquad (1)\end{aligned}$$

where $\mathcal{K}$ is the unwinding number [1]. Another way to express this uses complex conjugation:

$$\arcsin z = \overline{\arctan\left(\overline{\frac{z}{\sqrt{1-z^2}}}\right)} \ , \qquad (2)$$

where $\overline{z}$ is the complex conjugate of z. In contrast, if arctangent is defined as clockwise continuous (CC) the relation is

$$\arcsin z = \arctan\frac{z}{\sqrt{1-z^2}} \ . \qquad (3)$$

This last relation is of prime importance within Derive, and overrides the importance of maintaining CCC.

Kahan recognized the importance of algebraic identities and their dependence upon closure definitions. For this reason, he also discussed the idea of a signed zero. We shall discuss both ideas here.

2 Closure by CCC

We start by considering whether CAS should follow Kahan in making all their functions CCC. The principles seem to be as follows.

1. Standards are good. We should like all our arctangents to simplify to the same thing on all systems.

2. CCC looked the best candidate for a standard. Before settling on CCC, Kahan had reviewed the implementations existing at the time of his article. As we have noted above, time has not established CCC as the standard, certainly within CAS, although it continues to exert a strong influence.

3. The standard allows the closure of new functions to be determined automatically. It should be noted that a finite branch cut cannot be CCC at both ends, unless it has a singularity in the cut. Therefore this principle will not always apply.

4. Any convention will allow some algebraic relations and invalidate others. This point has two consequences. First, one should not waste time searching

for the perfect scheme, because it does not exist; second, selecting one relation as the basis for criticizing a particular closure scheme will not be conclusive, because supporters of other schemes will always be able to find similar relations that are favoured by their schemes.

Given these principles, why is CCC not universally used? The main reason is probably that algebraic relations and numerical relations are not all equally important. For different areas of mathematics, and even for different CAS, the algebraic relations can be roughly ranked in order of importance. (This ranking may be conscious or unconscious.) The scheme that gives the best form to the relations at the top of the list is the scheme selected. It may be that no universal order of priorities can be agreed on, in which case we shall have to live with different definitions in different areas of mathematics.

Therefore, if agreement is ever to be reached, a comprehensive list of algebraic identities would have to be drawn up, showing the forms of the identities under different closures. The equations above constitute an example of one entry in this list. The CAS would have to agree on an order of importance and then agree on the scheme that put that list in its best form. If this does not lead to CCC for every function, then one potential advantage of CCC will be lost, namely, the ability to predict closure in the absence of tables of properties. However, even with CCC, one cannot avoid tables of properties. Few people would be able to quote equation (1) without referring to a table. Users of algebra systems spend most of their time working with functions, rather than defining them, so simple consequences should take priority over simple definitions.

3 Signed zero

Kahan made a more radical proposal also [2]. The IEEE standard for floating point arithmetic contains separate encodings for $+0$ and -0. Therefore it is possible to evaluate differently $\arctan(5i/3+0) =$CCC-RESULT and $\arctan(5i/3-0) =$CC-RESULT.

Para-conjugation is defined to be $z^* = -\overline{z}$. A function $f(z)$ is said to have para-conjugate symmetry if $f(z^*) = f(z)^*$, which is to say it is symmetric about the imaginary axis. A function has *near* para-conjugate symmetry if $f(z^*) = f(z)^*$, provided $\Re(z) \neq 0$. Likewise, conjugate symmetry means $f(\overline{z}) = \overline{f(z)}$ and *near* conjugate symmetry means $f(\overline{z}) = \overline{f(z)}$, provided $\Im(z) \neq 0$. For example, because square root has a branch cut along the negative real axis, it has only near conjugate symmetry, and this under any closure scheme. However with Kahan's signed zero, we can gain full conjugate symmetry, because $\overline{\sqrt{-1+0i}} = \sqrt{-1-0i} = -i$.

The problem with a signed zero is the interpretation of an unqualified point on the branch cut. Thus if a user types $\arctan(5i/3)$, what does this mean? The system would have to assume $\arctan(5i/3 \pm 0)$ and return $\pm\frac{1}{2}\pi + i\ln 2$. It is this observation that leads us to a variation on Kahan's proposal. For each function having a branch cut, we avoid the question of closure by interpreting arguments on the cut as being indeterminate as to side, and returning both values. Provided the branch cuts are selected so that the indeterminacy manifests itself as an ambiguity in sign, then the system could return simplifications using some implementation of the $\pm$ symbol.

4 A proposal

We make the proposal that multivalued functions do not return unique values for arguments that lie on their branch cuts. In DERIVE, the idea that a function need not return a unique value has already been used in the implementation of the signum function. Consider the following ways of defining $\operatorname{sgn}(0)$.

1. Define $\operatorname{sgn}(0)$ to be 1 or -1. If this is done, the identity $\operatorname{sgn}(-z) = -\operatorname{sgn}(z)$ is not true at $z = 0$. On the other hand, the identity $|\operatorname{sgn} z| = 1$ is obeyed.

2. Define $\operatorname{sgn}(0)$ to be 0. In this case the anti-symmetry relation is valid but $|\operatorname{sgn} z| \neq 1$.

3. Let $\operatorname{sgn}(0)$ be an unspecified point on the unit circle of the complex plane. Both of the above identities are now retained. Notice that we are not proposing that $\operatorname{sgn}(0)$ be undefined or remain unevaluated. Having a natural representation for an arbitrary point on the unit circle turns out to be quite useful, for example, the solution of the equation $|z| = 1$ is $\operatorname{sgn}(0)$.

Reasoning similar to that just given for the signum singularity leads us to conclude that multivalued functions should not be given unique values along their branch cuts, but instead the functions should simplify to an arbitrary element of the set consisting of the values of the function on either side of the cut. In this way, identities are not invalidated by the assignment of unique values on the branch cuts. Consider as an example the logarithm function. The identity that is lost by specifying values on the branch cut is

$$\ln \overline{z} = \overline{\ln z} \ . \tag{4}$$

Enforcing CCC for logarithm and trying to retain the identity leads to the contradiction

$$\pi i = \ln(-1) = \overline{\ln(-1)} = \overline{\pi i} = -\pi i \ . \tag{5}$$

Thus we propose that $\ln(-1)$ simplify to $\pm\pi i$ instead of πi. Notice that we are not proposing to return a set of values, elementary functions do not return sets. Rather, the value returned can be operated on by mathematical operators the same way numbers can: for example, $(\pm i\pi)^2$ simplifies to $-\pi^2$.

The following table gives, for each of the multivalued elementary functions, the position of the branch cut, in DERIVE's implementation. Other CAS may place some of the branch cuts in different places. For example, the branch cuts of acoth were recently changed in Maple. The branch cuts are specified using $z = x + iy$. The table also contains for each function a relation that is valid on the whole complex plane except on the branch cuts, if a particular value is selected.

Function	Branch cut	Symmetry relation
ln :	$x < 0, y = 0$	$\ln \overline{z} = \overline{\ln z}$
asin :	$\|x\| > 1, y = 0$	$\operatorname{asin} \overline{z} = \overline{\operatorname{asin} z}$
acos :	$\|x\| > 1, y = 0$	$\operatorname{acos} \overline{z} = \overline{\operatorname{acos} z}$
atan :	$\|y\| > 1, x = 0$	$\operatorname{atan} z^* = (\operatorname{atan} z)^*$
acot :	$\|y\| > 1, x = 0$	$\operatorname{acot} z^* = \pi + (\operatorname{acot} z)^*$
asec :	$\|x\| < 1, y = 0$	$\operatorname{asec} \overline{z} = \overline{\operatorname{asec} z}$
acsc :	$\|x\| < 1, y = 0$	$\operatorname{acsc} \overline{z} = \overline{\operatorname{acsc} z}$
atanh :	$\|x\| > 1, y = 0$	$\operatorname{atanh} \overline{z} = \overline{\operatorname{atanh} z}$
acoth :	$\|x\| < 1, y = 0$	$\operatorname{acoth} \overline{z} = \overline{\operatorname{acoth} z}$
asinh :	$\|y\| > 1, x = 0$	$\operatorname{asinh} z^* = (\operatorname{asinh} z)^*$
acosh :	$x < 1, y = 0$	$\operatorname{acosh} \overline{z} = \overline{\operatorname{acosh} z}$
asech :	$x < 0, x > 1, y = 0$	$\operatorname{asech} \overline{z} = \overline{\operatorname{asech} z}$
acsch :	$\|y\| < 1, x = 0$	$\operatorname{acsch} z^* = (\operatorname{acsch} z)^*$
nth root :	$x < 0, y = 0$	$(\overline{z})^{1/n} = \overline{z^{1/n}}$

The ramifications on a CAS that systematically implemented this proposal are significant, but we think beneficial. For example, the last line of the above table means that $\sqrt{-1}$ should simplify to $\pm i$ instead of i. Otherwise, a CAS should not simplify $\sqrt{\overline{z}} - \overline{\sqrt{z}}$ to 0, even though z could be real and negative.

Although our reasoning may be valid, we realize that simplifying $\sqrt{-1}$ to $\pm i$ would have the mathematics community howling. After all, i is widely defined as being $\sqrt{-1}$. The problem comes from trying to define i in terms of elementary functions of real numbers. Gauss defined complex numbers as pairs of real numbers (x, y) that obey various rules. One consequence of these rules is $(0, 1)^2 = (-1, 0)$. In such a system, since $\sqrt{-1}$ does not serve as the definition for i, it can be allowed to simplify to $\pm i$.

Computer algebra systems can easily store expressions that represent arbitrary elements of a set, such as $\pm i$. However, strictly numerical programs must return a number or an error. No one wants their scientific pocket calculator to return an error for arcsin(2). Thus, adopting a convention for assigning a unique value along branch cuts may well be appropriate for such programs. As is well known, there is a tension between symbolic simplification and numerical computation. Thus most CASs already have two ways of evaluating mathematical expressions: exact (i.e. symbolic) mode and approximate (i.e. numeric) mode. Therefore, we propose that $\sqrt{-1}$ should simplify to $\pm i$ and it should approximate to i. What would happen internally is that $\sqrt{-1}$ would simplify to $\pm i$, which is stored as $(\pm 1)i$ and then ± 1 would approximate to 1.

The following table summarizes the consequences of systematically using this algorithm on branch cuts of various inverse elementary functions:

Expression	Simplifies to	Approximates to
$\sqrt{-1}$	$\pm i$	i
$\ln(-1)$	$\pm\pi i$	$3.1415...i$
$\arctan(5i/3)$	$\pm\pi/2 + i\ln 2$	$1.5707... + 0.6931...i$
$\arcsin(5/4)$	$\pi/2 \pm i\ln 2$	$1.5707... + 0.6931...i$

References

[1] R.M.Corless and D.J.Jeffrey, "Editor's corner:The unwinding number", this BULLETIN, pp.28–35.

[2] W. Kahan,"Branch cuts for complex elementary functions", in *The State of the Art in Numerical Analysis:* Proceedings of the Joint IMA/SIAM Conference on the State of the Art in Numerical Analysis, University of Birmingham, April 14-18, 1986, M. J. D. Powell and A. Iserles, Eds, Oxford University Press.