Multivalued Elementary Functions in Computer-Algebra Systems

David J. Jeffrey

Department of Applied Mathematics
University of Western Ontario
djeffrey@uwo.ca

Abstract. An implementation (in Maple) of the multivalued elementary inverse functions is described. The new approach addresses the difference between the single-valued inverse function defined by computer systems and the multivalued function which represents the multiple solutions of the defining equation. The implementation takes an idea from complex analysis, namely the *branch* of an inverse function, and defines an index for each branch. The branch index then becomes an additional argument to the (new) function. A benefit of the new approach is that it helps with the general problem of correctly simplifying expressions containing multivalued functions.

1 Introduction

The manner in which computer-algebra systems handle multivalued functions, specifically the elementary inverse functions, has been the subject of extensive discussions over many years. See, for example, [5,6,8]. The discussion has centred on the best way to handle possible simplifications, such as

$$\sqrt{z^2} = z\ ? \quad \arcsin(\sin z) = z\ ? \quad \ln(e^z) = z\ ? \tag{1}$$

In the 1980s, errors resulting from the incorrect application of these transformations were common. Since then, systems have improved and now they usually avoid simplification errors, although the price paid is often that no simplification is made when it could be. For example, Maple 18 fails to simplify

$$\sqrt{1-z}\sqrt{1+z} - \sqrt{1-z^2}\ ,$$

even though it is zero for all $z \in \mathbb{C}$, see [2,8]. Here a new way of looking at such problemsis presented.

The discussion of possible treatments has been made difficult by the many different interpretations placed on the same symbols by different groups of mathematicians. Sorting through these interpretations, and assessing which ones are practical for computer algebra systems, has been an extended process. In this paper, we shall not revisit in any detail the many past contributions to the discussion, but summarize them and jump to the point of view taken here.

1.1 A Question of Values

One question which has been discussed at length concerns the number of values represented by function names. One influential point of view was expressed by Carathéodory, in his highly regarded book [4]. Considering the logarithm function, he addressed the equation

$$\ln z_1 z_2 = \ln z_1 + \ln z_2 \ , \tag{2}$$

for complex z_1, z_2. He commented [4, pp. 259–260]:

> The equation merely states that the sum of one of the (infinitely many) logarithms of z_1 and one of the (infinitely many) logarithms of z_2 can be found among the (infinitely many) logarithms of $z_1 z_2$, and conversely every logarithm of $z_1 z_2$ can be represented as a sum of this kind (with a suitable choice of $\ln z_1$ and $\ln z_2$).

In this statement, Carathéodory first sounds as though he thinks of $\ln z_1$ as a symbol standing for a set of values, but then for the purposes of forming an equation he prefers to select one value from the set. Whatever the exact mental image he had, the one point that is clear is that $\ln z_1$ does not have a unique value, which is in strong contrast to every computer system. Every computer system will accept a specific value for z_1 and return a unique $\ln z_1$.

The reference book edited by Abramowitz & Stegun [1, Chap 4] is another authoritative source, as is its successor [15]. They both define, to take one example, the solution of $\tan t = z$ to be $t = \mathrm{Arctan}\, z = \arctan z + k\pi$. When listing properties, they both give the equation

$$\mathrm{Arctan}(z_1) + \mathrm{Arctan}(z_2) = \mathrm{Arctan}\,\frac{z_1 + z_2}{1 - z_1 z_2} \ . \tag{3}$$

For $z_1 = z_2 = \sqrt{3}$, we have $\mathrm{Arctan}\,\sqrt{3} + \mathrm{Arctan}\,\sqrt{3} = \mathrm{Arctan}(-\sqrt{3})$. For computer users, this is confusing, because their systems return values $\arctan\sqrt{3} = \pi/3$ and $\arctan(-\sqrt{3}) = -\pi/3$, and most users do not see the difference between Arctan and arctan. (Below, a new form of (3) is given.) By comparing the Abramowitz & Stegun definition with the statement of Carathéodory, we can see that as far as equations are concerned, both sets of authors favour an interpretation based on interactively selecting one value from a set of possible ones.

Riemann surfaces give a very pictorial way of seeing multi-valuedness [16,7], but a question remains whether they can be used computationally [13]. To discuss these approaches in detail will deflect attention from the implementation here. Therefore, now that alternative approaches have been noted, they will be set aside.

Here, an inverse function will have a single value [13]. Further, that single value will be determined by the arguments to the function and not by the context in which it finds itself.

2 A New Treatment of Inverse Functions

The basis of the new implementation is notation introduced in [11]. To the standard function $\ln z$, a subscript is added:

$$\ln_k z = \ln z + 2\pi i k \ .$$

Here the function $\ln z$ denotes the principal value of logarithm, which is the single-valued function with imaginary part $-\pi < \Im \ln z \leq \pi$. This is the function currently implemented in Maple, Mathematica, Matlab and other systems. In contrast, $\ln_k z$ denotes the kth branch of logarithm. With this notation, the statement above of Carathéodory can be restated unambiguously as

$$\exists k, m, n \in \mathbb{Z},\ \text{such that}\ \ln_k z_1 z_2 = \ln_m z_1 + \ln_n z_2 \ .$$

His "and conversely" statement is actually a stronger statement. He states

$$\forall k \in \mathbb{Z}, \exists m, n \in \mathbb{Z},\ \text{such that}\ \ln_k z_1 z_2 = \ln_m z_1 + \ln_n z_2 \ .$$

In the light of his converse statement, Carathéodory's first statement could be interpreted as meaning

$$\forall m, n \in \mathbb{Z}, \exists k \in \mathbb{Z},\ \text{such that}\ \ln_m z_1 + \ln_n z_2 = \ln_k z_1 z_2 \ .$$

I think the English statement does not support this interpretation, but it may be supported by the original German. In any event, it shows the greater conciseness of branch notation.

The principal of denoting explicitly the branch of a multivalued function will be extended here to all the elementary multivalued functions. In order for the new treatment to be smoothly implemented in Maple, a system of notation is needed that can co-exist with the built-in functions of Maple.

2.1 Notation for Inverses

The built-in functions for which we shall be implementing branched replacements are

– `log(z)`,
– `arcsin(z)`, `arccos(z)`, `arctan(z)`,
– `arcsinh(z)`, `arccosh(z)`, `arctanh(z)`,
– fractional powers $z^{1/n}$.

Rather than risk confusion by trying to modify the actions of these names within Maple, we shall leave the built-in functions untouched and work with independent, clearly defined and unambiguous notation for the branched functions.

The model we follow is to adapt the notation `invfunc` used in Maple; Mathematica has a similar construction `InverseFunction`. The most direct presentation is simply to display the definitions, with source code.

2.2 Subscripts in Maple

A subscript on a function f, as in $f_k(z)$, is really an additional argument to the function, except that instead of placing it in parentheses, as in $f(k,z)$, we choose subscripting. In Maple, however, the programming is quite different in the two cases. Thus $f(k,z)$ is coded as

```
f:= proc (k,z) ... end
```

and the k and z can be used in the procedure without further programming. A subscripted function, however, is written as `f[k](z)`, and is an 'indexed name'. The procedure is now coded as

```
f:= proc (z) ... end
```

and inside the procedure there is a variable available to the program called `procname`. If the procedure has been called with an indexed name, then this is contained in `procname` and the index, i.e., the subscript, can be retrieved for use in the procedure by using the `op` function.

3 Particular Functions

In this section, the inverses of the elementary functions are defined in the new notation. The implementations use Maple's indexed names, and in Maple's 2-D printing, the indexes appear as subscripts.

3.1 Inverse Sine

The principal branch of the inverse sine function is denoted in Maple by `arcsin`. Using this, we define the branched inverse sine by

$$\mathrm{invsin}_0\, z = \arcsin z \;, \tag{4}$$

$$\mathrm{invsin}_k\, z = (-1)^k\, \mathrm{invsin}_0\, z + k\pi \;. \tag{5}$$

The principal branch now has the equivalent representation $\mathrm{invsin}_0\, z = \mathrm{invsin}\, z = \arcsin z$. It has real part between $-\pi/2$ and $\pi/2$. Notice that the branches are spaced a distance π apart in accordance with the antiperiod[1] of sine, but the repeating unit is of length 2π in accord with the period of sine.

The Maple code for the function is

```
invsin := proc (z::algebraic) local branch;
             if nargs <> 1 then
                error "Expecting 1 argument, got", nargs ;
```

[1] An antiperiodic function is one for which $\exists\alpha$ such that $f(z+\alpha) = -f(z)$, and α is then the antiperiod. This is a special case of a quasi-periodic function [14], namely one for which $\exists\alpha,\beta$ such that $f(z+\alpha) = \beta f(z)$.

```
          elif type(procname, 'indexed') then
              branch := op(procname);
              branch*Pi+(-1)^branch*arcsin(z);
          else arcsin(z);
          end if;
      end proc;
```

The `nargs` function counts the number of arguments supplied by the user, and although here the code is restricted to 1 argument, one could allow the branch number to be passed as an argument instead of as a subscript. Note that the code is not 'industrial strength', and in particular the branch is not tested for being an integer. Since the code is exploratory, it relies on the user being sensible. Examples of its use appear below.

3.2 Inverse Cosine

The principal branch has real part between 0 and π, and this is easiest achieved by setting $\operatorname{invcos}_k z = \operatorname{invsin}_{k+1} z - \pi/2$. The code is

```
invcos := proc (z::algebraic) local branch;
          if nargs <> 1 then
              error "Expecting 1 argument, got", nargs ;
          elif type(procname, 'indexed') then
              branch := op(procname);
              invsin[branch+1](z)-Pi/2;
          else arccos(z);
          end if;
      end proc;
```

3.3 Inverse Tangent

The principal branch has real part from $-\pi/2$ to $\pi/2$, and the kth branch is $\operatorname{invtan}_k z = \operatorname{invtan} z + k\pi$. As code:

```
invtan := proc (z::algebraic) local branch;
          if nargs <> 1 then
              error "Expecting 1 argument, got", nargs ;
          elif type(procname, 'indexed') then
              branch := op(procname);
              branch*Pi+arctan(z);
          else arctan(z);
          end if;
      end proc;
```

The two-argument inverse tangent function has been implemented in many computer languages. It is a synonym for arg, in that $\arg(x + iy) = \arctan(y, x)$ for $x, y \in \mathbb{R}$. It can be described using the branches of invtan as

$$\arctan(y, x) = \operatorname{invtan}_k(y/x) ,$$

where $k = H(-x)\operatorname{sgn} y$, and H is the Heaviside step function. For x small this is inaccurate, when using invcot is better.

3.4 The Logarithm

The logarithm is the inverse of the exponential function, and therefore our convention would suggest implementing the branched version using `invexp`. This, however, seems too radical for acceptance, so we use `loge` instead. Another possibility might seem to be `Log`, but this is unsatisfactory because textbooks cannot agree on the definition of Log. Also Mathematica uses `Log[x]` as its standard log function, and may in the future have its own branch implementation.

```
loge := proc (z::algebraic) local branch;
          if nargs <> 1 then
               error "Expecting 1 argument, got", nargs ;
          elif type(procname, 'indexed') then
               branch := op(procname);
               ln(z) + 2*Pi*I*branch;
          else ln(z);
          end if;
        end proc;
```

3.5 Inverse Hyperbolic Functions

A common point of contention in notation for inverse hyperbolic functions is whether to write `arcsinh` or `arsinh`, and similarly for the other functions. The point of the debate being that the geometrical interpretation of inverse sinh is an area, not an arc. Maple and Mathematica use the former notation to the chagrin of more enlightened authors [3,9] who prefer the latter. They argue that `arc` should not be merely a synonym for inverse. The convention here allows us to avoid this argument by using the `inv` prefix. We use the Russian abbreviations for the primary functions to save typing. Thus we define in the obvious way `invsh[k](z), invch[k](z), invth[k](z)`. We save space by not listing them.

3.6 Fractional Powers

The principal branch of $z^{1/n}$ is defined by $\exp(\frac{1}{n}\ln z)$, and replacing $\ln z$ by $\ln_k z$ gives the branched function. The standard notation for roots and fractional powers does not leave an obvious place for the branch label, and most obvious names are already used by Maple or Mathematica. We use the name `invpw`, meaning inverse (integer) power. The Maple code defines `invpw[k](z,n)`, where the subscript is the branch, as usual, while the fractional power is $1/n$. Thus it is modelled on the Maple `surd` function. Unlike the other inverse functions, there are only n distinct values, but we allow k to be any integer.

Since square root is so common, it is coded separately as `invsq[k](z)`, and it can be displayed in traditional notation as $(-1)^k\sqrt{z}$.

4 Applications

We now demonstrate some uses of the new notation.

4.1 Plotting

With the new functions, we can easily plot branches. Figure 1 shows plots produced by the Maple commands

```
> plot([invsin[-1](x),invsin(x),invsin[1](x)],x=-1 .. 1,
    linestyle=[2,1,3]);
> plot([invtan[-1](x),invtan(x),invtan[1](x),invtan[2](x)],
     x=-5..5, discont = true, linestyle = [2, 1, 3, 4]);
```

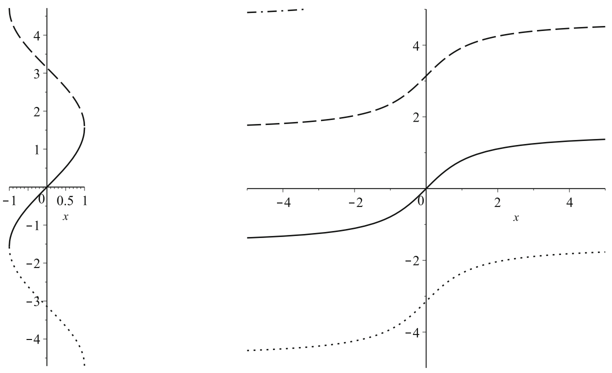

Fig. 1. The branches of inverse sine and inverse tangent plotted taking advantage of branch notation

4.2 Identities

In order to express identities containing inverse functions correctly, we need the unwinding number,

$$\mathcal{K}(z) = \left\lceil \frac{z-\pi}{2\pi} \right\rceil ,$$

defined in [5] (rather than in [6] where the sign is different). Note that the unwinding number is a built-in function in Maple, called `unwindK`. This immediately gives us

$$\ln_k e^z = z - 2\pi i \mathcal{K}(z) + 2\pi i k \ . \tag{6}$$

Note the special case $\ln_{\mathcal{K}(z)} e^z = z$.

Consider an identity one might see in a traditional treatment:

$$\cos x = \sqrt{1 - \sin^2 x}\ , \tag{7}$$

where the author would add "and the branch of the root is chosen appropriately". Using the branched root, we write the more precise

$$\cos x = \mathrm{invsq}[\mathcal{K}(2ix)](1 - \sin^2 x) = (-1)^{\mathcal{K}(2ix)}\sqrt{1 - \sin^2 x}\ . \tag{8}$$

We can contrast the two approaches in Maple with the command

```
> plot([ sqrt(1-sin(x)^2), invsq[unwindK(2*x*I)](1-sin(x)^2)],
       x = -7 .. 7, linestyle = [2, 1]);
```

The resulting plot is given in figure 2.

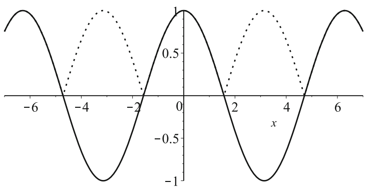

Fig. 2. The graph of $\sqrt{1 - \sin^2 x}$ using branch notation for square root

We return to the Abramowitz and Stegun [1] 'identity' (3). The branch problems with this equation are neatly displayed by the Maple command

```
> plot3d([arctan(x)+arctan(y), arctan((x+y)/(1-x*y))],
x = -2 .. 2, y = -2 .. 2, orientation = [-45, 45, 0])
```

The more precise identity is

$$\mathrm{invtan}(x) + \mathrm{invtan}(y) = \mathrm{invtan}_k \frac{x+y}{1-xy},\ \text{where } k = H(xy-1)\,\mathrm{sgn}(x)\ , \tag{9}$$

and H is the Heaviside step. A more complicated example from [1] is their identity for $\mathrm{Arcsin}\, x + \mathrm{Arcsin}\, y$, which becomes

$$\mathrm{invsin}\, x + \mathrm{invsin}\, y = \mathrm{invsin}[k]\left(x\sqrt{1-y^2} + y\sqrt{1-x^2}\right)\ , \tag{10}$$
$$k = H(x^2 + y^2 - 1)(\mathrm{sgn}\, x + \mathrm{sgn}\, y)/2\ .$$

Here the branch of invsin is allowed to vary, but there might be another formula which includes variable branches of square root.

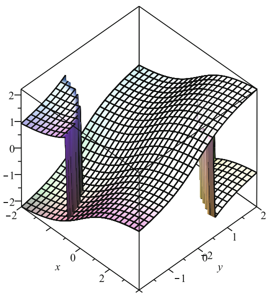

Fig. 3. A plot of the sum of two inverse tangents and the usual formula for their sum

As a final identity, we consider formula (4.4.39) in [1].

$$\operatorname{Arctan}(x+iy) = k\pi + \frac{1}{2}\arctan\frac{2x}{1-x^2-y^2} + \frac{i}{4}\ln\frac{x^2+(y+1)^2}{x^2+(y-1)^2} \ .$$

To turn this identity into something that computer-algebra systems can use, one should decide what to do with k. This can be replaced by

$$\operatorname{invtan}_k(x+iy) = \frac{1}{2}\operatorname{invtan}_n\frac{2x}{1-x^2-y^2} + \frac{i}{4}\ln\frac{x^2+(y+1)^2}{x^2+(y-1)^2} \ ,$$

where $n = 2k + \operatorname{sgn}(x)H(x^2+y^2-1)$.

4.3 Calculus

Calculating the derivative of an inverse function is a standard topic in calculus. The results in the textbooks are restricted to the principal branches of the functions. It is possible, however, to generalize results to any branch. For example

$$\frac{d}{dx}\operatorname{invsin}_k x = \frac{1}{\cos(\operatorname{invsin}_k x)} = \frac{(-1)^k}{\sqrt{1-x^2}} \ .$$

Integration by substitution is a well-known application of inverse functions. A specific difficulty has been the application of the substitution $u = \tan\frac{1}{2}x$ in integrals such as

$$\int\frac{3\,dx}{5-4\cos x} = \int\frac{6\,du}{1+9u^2} = 2\arctan(3\tan\tfrac{1}{2}x) \ . \qquad (11)$$

The right-hand side is discontinuous, as has been pointed out in [12,10]. The correction to the usual integration formula [12] can be rewritten in the new notation as

$$\int\frac{3\,dx}{5-4\cos x} = 2\operatorname{invtan}_{\mathcal{K}(ix)}(3\tan\tfrac{1}{2}x) \ . \qquad (12)$$

The contrast is illustrated in figure 4 by the plot

```
> plot([ 2*invtan[unwindK(I*x)](3*tan((1/2)*x)),
     2*arctan(3*tan((1/2)*x))], x=-3..9,linestyle=[2,1],
        discont=true);
```

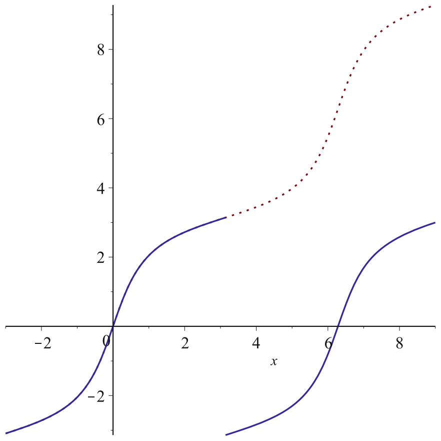

Fig. 4. A graph of the discontinuous and continuous integrals discussed in (11) and (12)

5 Conclusions

The focus here has been on the implementation of multivalued inverse functions in a computer-algebra system. The development of the notation, and of tools such as the unwinding number, has been motivated by the idea that traditional treatments of multivalued functions are not precise enough. Too often, decisions on branch choices are avoided by texts, the avoidance being often covered by phrases such as “taking an appropriate branch”. The notation here allows one to state precisely which branch of a function should be used, and the notation also reminds one that such choices are important.

After branch information is added to existing equations, they are typically longer than before. This means that people looking for elegance rather than strictness will find little benefit in the new approach and notation. Looking back at Carathéodory’s discussion of (2), we can see exactly the desire for elegance of presentation bringing with it the cost of impreciseness.

Outside computer-algebra systems, the notation also offers benefits. For example, it should make the topic easier for students. We already teach students that $y = x^2$ implies $x = \pm\sqrt{y}$, and we teach calculus students that $dy/dx = 1$ implies $y = x + K$, where K is a constant. So solutions to equations in which arbitrary elements appear are already part of a student’s education. By using branch indexing, we can bring all the elementary inverse functions into a single pattern, and both students and computers are forced to confront branch choices explicitly.

There are many multivalued functions in mathematics, and here we have considered only the elementary functions. The principles developed here can be found already in Maple to varying degrees. The Lambert W function has been fully implemented using the same ideas of explicit branches as here. Maple's `RootOf` construction uses an index to specify different roots of an equation. Although there is a tendency to think of `RootOf` as specifying values rather than functions, there is no reason not to use it to define a function, although its generality will often make the branch structure of the defined function difficult to understand. The current approach is one of a number of possibilities for correct manipulation in a computer-algebra system. It fits together with the unwinding number approach happily and offers other ways of presenting and working with expressions. As with the unwinding number, there remains much scope for further development.

References

1. Abramowitz, M., Stegun, I.J.: Handbook of Mathematical Functions. Dover (1965)
2. Bradford, R.J., Corless, R.M., Davenport, J.H., Jeffrey, D.J., Watt, S.M.: Reasoning about the elementary functions of complex analysis. Annals of Mathematics and Artificial Intelligence 36, 303–318 (2002)
3. Bronshtein, I.N., Semendyayev, K.A., Musiol, G., Muehlig, H.: Handbook of Mathematics, 5th edn. Springer, Heidelberg (2007)
4. Carathéodory, C.: Theory of functions of a complex variable, 2nd edn. Chelsea, New York (1958)
5. Corless, R.M., Davenport, J.H., Jeffrey, D.J., Watt, S.M.: According to Abramowitz and Stegun. SIGSAM Bulletin 34, 58–65 (2000)
6. Corless, R.M., Jeffrey, D.J.: The unwinding number. Sigsam Bulletin 30(2), 28–35 (1996)
7. Corless, R.M., Jeffrey, D.J.: Elementary Riemann surfaces. Sigsam Bulletin 32(1), 11–17 (1998)
8. Davenport, J.H.: The challenges of multivalued "functions". In: Autexier, S., Calmet, J., Delahaye, D., Ion, P.D.F., Rideau, L., Rioboo, R., Sexton, A.P. (eds.) AISC 2010. LNCS, vol. 6167, pp. 1–12. Springer, Heidelberg (2010)
9. Gullberg, J.: Mathematics: From the Birth of Numbers. W. W. Norton & Company, New York (1997)
10. Jeffrey, D.J.: The importance of being continuous. Mathematics Magazine 67, 294–300 (1994)
11. Jeffrey, D.J., Hare, D.E.G., Corless, R.M.: Unwinding the branches of the Lambert W function. Mathematical Scientist 21, 1–7 (1996)
12. Jeffrey, D.J., Rich, A.D.: The evaluation of trigonometric integrals avoiding spurious discontinuities. ACM Trans. Math. Software 20, 124–135 (1994)
13. Jeffrey, D.J., Norman, A.C.: Not seeing the roots for the branches. SIGSAM Bulletin 38(3), 57–66 (2004)
14. Lawden, D.F.: Elliptic functions and applications. Springer (1989)
15. Lozier, D.W., Olver, F.W.J., Boisvert, R.F.: NIST Handbook of Mathematical Functions. Cambridge University Press (2010)
16. Trott, M.: Visualization of Riemann surfaces of algebraic functions. Mathematica in Education and Research 6, 15–36 (1997)