Multivalued Elementary Functions in Computer-Algebra Systems

David J. Jeffrey

djeffre @ o.ca

Abstract. ... *branch* ...

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.

G.A. Aranda-Corral et al. (Eds.): AISC 2014, LNAI 8884, pp. 157–167, 2014.

1.1 A Question of Values

One question which has been discussed at length concerns the number of val-

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 indepen-

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 $f(k, z$

```
            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 $\mathrm{invcos}_k z = \mathrm{invsin}_{k+1} z - \pi/2$. 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 $\mathrm{invtan}_k z = \mathrm{invtan}\, z + k\pi$. 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

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

4 Applications

We now demonstrate some uses of the new notation.

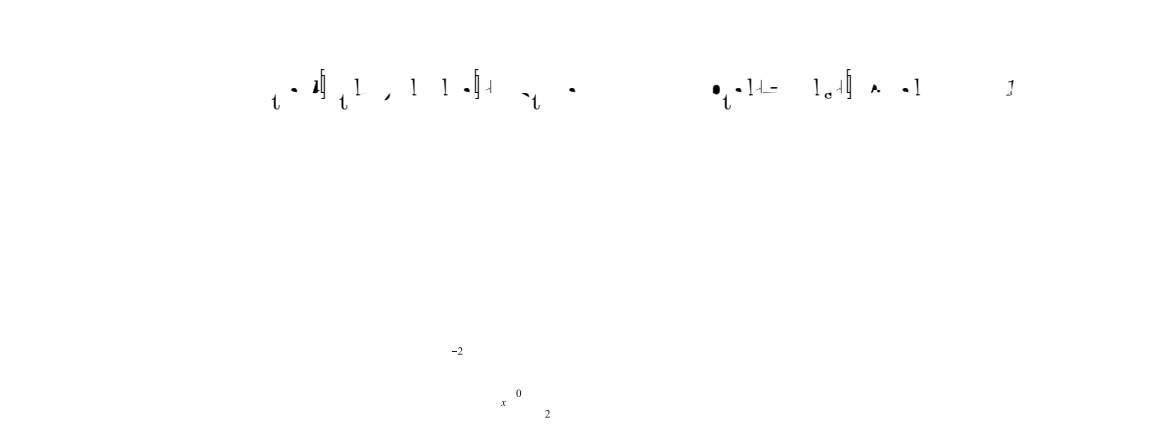
4.1 Plotting

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]);
```

Consider an identity one might see in a traditional treatment:

$$\cos x = \sqrt{1 - \sin}$$



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);
```

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