1.1 Introduction to package fsimp

Documentation is under construction!

The `fsimp` package provides a 'fullsimp' simplification function for expressions. It uses the simplification routines in core Maxima, but combines them with an algorithm that allows expressions to grow and shrink while trying different simplification routines. It always chooses the smallest expression as the most simple. The user has some control over the algorithm and routines.

1.2 Functions and Variables for fsimp

`fsdebug` [Variable]

Setting the variable `fsdebug` to 1 will allow the user to see the fsimp function operations, list selection of simplification routines, etc.

`fsimp (expr,`[*simp1,simp2,...*]) [Function]

The `fullsimp` function takes an input expression `expr` and attempts to find a simpler form using the builtin simplification routines in core Maxima. These include: `[resimplify, expand, combine, radcan, ratsimp, rootscontract, xthru, multthru, factor, sqrtdenest, triglist,exptlist, loglist].` Additional trigonometric simplifications are applied as well. The full list of simplifications is shown in the variable `simplist` which will be shown if the variable `fsdebug` is set to 1.

There are two ways to manipulate the simplifications in `simplist`. First, the option arguments to `fullsimp`, `simp1, simp2, etc.` are simplification routines the user wants *excluded* from `simplist`.

Second, the user may create a *list* called `fs_custom_simp`, and place any user defined simplification routines. For example the code

```
fs_custom_simp:[lambda([q], block([algebraic : true], ratsimp(q)))]
```

will create a simplification routine where the variable `algebraic` is set to `true`, while the `ratsimp` simplification is applied.

In general, `fullsimp` tries to find the smallest equivalent expression based on the Common Lisp `conssize` and string size of an expression.

The user may manipulate the order of simplifications by judicious use of the exclusion option and the custom simplification option.

```
(%i1) load("fsimp.mac")$
(%i2) fullsimp((1+cos(t))/sin(t));
                                              t
(%o2)                                   cot(-)
                                              2
(%i3) fullsimp(cos(x)+%i*sin(x));
                                          %i x
(%o3)                                   %e
```

If you wish to exclude trigonometric simplifications,

```
(%i1) load("fsimp.mac")$
(%i2) fullsimp((1+cos(t))/sin(t),fstrigsimp);
                                        t
(%o2)                               cot(-)
                                        2
```

Appendix A Function and Variable index