Fortran Utilities for Evaluating Expressions

The module 'evaluate' contains routines used to evaluate a mathematical expression contained in a character string. For example, the string ′cos((a+b)^2+1.6*c)′ is an allowable expression. This 'evaluate' module is contained in the file **evalmod.f90**. The module can be obtained from the website http://www.gbenthien.net/strings/index.html. To use the routines in this module the user needs to add the statement

use evaluate

to the top of the program. The main routines are described below.

1. SUBROUTINE EVALEXPR(*expr*, *value*)

   This routine evaluates the mathematical expression contained in the string *expr* and puts the resulting value in *value*. The mathematical expression can contain previously defined variables (see subroutine defparam), numbers (e.g., 1.5 or 1.5e-6), arithmetic operators (+, -, *, /, ^), and a number of mathematical functions (*sin, cos, tan, sqrt, exp, log, ln, abs, ang, real, imag, conjg, complex*). The expression can also use nested levels of parentheses for grouping. There are two predefined variables that are available to the user — the constant *pi*=3.14159265358979 and the imaginary unit *i*. The output variable *value* can be an integer, real number, or complex number (single or double precision). All computations are performed in double precision complex. Complex numbers can be entered as *a+i*b* (assuming that the user has not redefined *i*) or as *complex(a,b).*

2. SUBROUTINE DEFPARAM(*symbol*, *expr*) or DEFPARAM(*symbol, value*)

   This routine evaluates the expression contained in the string *expr* and assigns the value to a variable with the name given by the string *symbol*. Variable names must begin with a letter and can have no more than 24 characters. In addition, variable names can not contain any of the characters +, -, *, /, ^, (,), or a comma. The input *value* can be an integer, real number, or a complex number (single or double precision).

3. SUBROUTINE EVALEQN(*eqn*)

   This subroutine evaluates the right-hand-side of the equation contained in the string *eqn* and assigns the value to the variable name on the left-hand-side of *eqn*.

4. SUBROUTINE GETPARAM(*symbol*, *value*)

   This routine returns the number *value* associated with the variable name *symbol*. The output value can be an integer, a real number, or a complex number (single or double precision).

After execution of any of the above routines the global variable *ierr* contains error information. If *ierr* is zero, then there were no errors. Other possible values for *ierr* are

1 Expression empty
2 Parentheses don't match
3 Number string does not correspond to a valid number
4 Undefined symbol
5 Less than two operands for binary operation
6 No operand for unary plus or minus operators
7 No argument(s) for function
8 Zero or negative real argument for logarithm
9 Negative real argument for square root
10 Division by zero
11 Improper symbol format
12 Missing operator
13 Undefined function
14 Argument of tangent function a multiple of pi/2

Error messages are also written to the screen.