Functions

P!\K/AVOP|Un0X

COLLABORATORS

	TITLE : Functions	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	P!\K/AVOP|Un0X	June 8, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

Functions

1.1 Functions

Functions :

Click here for the main documentation.

Laplace offers a lot (really? ;-) internal functions. Call them with the parameters seperated by commas embedded in () brackets, e.g.

sin(pi)

exp(1)

parameter : ... means that the parameter must evaluate to one of the given object types.

result : ... means that the result is of the given object types. If there are more than one type, each type coresponds to a type in the parameter.

abs(expr)

acos(expr)

acosh(expr)

acot(expr)

addrows(matrix,row1,row2,expr)

asin(expr)

asinh(expr)

atan(expr)

atanh(expr)

combine(list)

const(list)

cos(expr)

cosh(expr)

cot(expr)

debug(string)

det(expr)

diff(function,variable)

dispose(list)

disposeall()

do(list)

eq(expr,expr)

eqleft(eq)

eqright(eq)

eval(expr)

exp(expr)

fak(expr)

include(path)

inv(expr)

ln(expr)

log(expr)

matrix([list],[list],..)

multrow(matrix,row1,expr)

neg(expr)

rang(expr)

sign(expr)

sin(expr)

sinh(expr)

solve(expr)

spur(expr)

sqrt(expr)

swaprows(matrix,row1,row2)

tan(expr)

tanh(expr)

taylor(function,a,grade)

trans(expr)

umatrix(size)

vector(list)

window(reference)

1.2 Trigonometric functions

sin(expression) :

cos(expression) :

tan(expression) :

cot(expression) :

asin(expression) :

acos(expression) :

atan(expression) :

acot(expression) :

sinh(expression) :

cosh(expression) :

tanh(expression) :

asinh(expression) :

acosh(expression) :

atanh(expression) :

expression : real, equation

result : real, equation

Calculate those well known trigonometric functions.

1.3 Functions - abs

abs(expression) :

expression : real, vector, equation

result : real, vector, equation

Real : Returns the absolute of the real.

Vector : Returns the absolute of the vector, which is defined as sqrt(a1ˆ2 + a2ˆ2 + ...) where a1, a2 are the components of the vector. (Physically this is the length of the vector.)

1.4 Functions - addrows

addrows(matrix,r1,r2,expression) :

matrix : matrix (must be variable or parameter)

r1 : real (should be integer)

r2 : real (should be integer)

expression : real

result : matrix

This will copy the input matrix, except that row r1 will be replace by (row r1)+expression*(row r2)

1.5 Functions - combine

combine(list) :

result : matrix

List is a list of object names of vectors or matrices (may be mixed).

This will create a matrix that is a combinition of the input vectors/matrices. E.g.

combine(matrix([1,2],[3,4]),vector(a,b))

/1 3 a\

\2 4 b/

1.6 Functions - const

const(list) :

List is a list of object names seperated by commas, where name is name[:type] the object name with an optional type specifier :

r, real - real object (default)

v, vec, vector - vector object, any dimension

v[d], vec[d], vector[d] - vector object, dimension d

m, mat, matrix - matrix object, any size

m[r,c], mat[r,c], matrix[r,c] - matrix object, size r*s

const() will create the named objects.

1.7 Functions - debug

debug(string) :

This will add the string to the debug list.

1.8 Functions - det

det(expression) :

expression : matrix, equation

result : real, equation

Returns the determinant of the matrix.

1.9 Functions - diff

diff(function,variable) :

This will calculate the first derivation of the given function.

There a different kinds of syntax for this command :

1) Calculate the derivation of an already defined function of one variable :

diff(functionname) - e.g.

f(x)=x^2

diff(f)

2) Calculate the (partial) derivation of an already defined function of one or more variables :

diff(functionname,variable) - e.g.

f(x,y)=x^2+y^3

diff(f , x)

diff(f , y)

3) Calculate the (partial) derivation of an already definied function of two or more variables, setting some variable to constant values :

diff(functionname(parameterlist),variable) - e.g.

f(x,y)=x^2+y^3*x

diff(f(x,1) , x)

g(x)=diff(f(x,y) , y)

4) Calculate the (partial) derivation of an explicit given function of one or more variables :

diff((expression),variable) - e.g.

diff((x^2*sin(x)) , x)

Don’t forget the brackets, otherwise Laplace cannot realize, that this is not a reference.

Usually a function has free variables x, y or z. If you want to other variable names, you have to declare them as constants before using diff(), or use diff() inside of a function definition. E.g.

diff((urps^2*sin(urps)),urps) -> Error : undefined reference.

use instead :

const(urps)

diff((urps^2*sin(urps)) , urps)

or

f(urps)=diff((urps^2*sin(urps)) , urps)

1.10 Functions - dispose

dispose(list) :

This will dispose the previously defined objects. List is a list of object names seperated by commas.

1.11 Functions - disposeall

disposeall() :

This will dispose all previously defined objects.

1.12 Functions - do

do(list) :

This is a synonym for an expression list. E.g.

{debug("Hello world !"), sin(2)} and

do(debug("Hello world !"), sin(2)) are equal.

1.13 Functions - eq

eq(expression, expression) :

expression : real, vector, matrix

result : equation

This will create an equation object.

1.14 Functions - eqleft

eqleft(equation) :

Get the left side of an equation.

1.15 Functions - eqright

eqright(equation) :

Get the right side of an equation.

1.16 Functions - eval

eval(expression) :

expression : real, vector, matrix, equation

result : depends on expression

This will evaluate the given expression. Parameter references are replaced, too.

1.17 Functions - exp

exp(expression) :

expression : real, equation

result : real, equation

Returns e^x of the real. Use this functions instead of entering e^x directly

1.18 Functions - fak

fak(expression) :

expression : real, equation

result : real, equation

Returns x! (factorial) of a whole reals. x! for x <= 0 is 0.

1.19 Functions - include

include(path) :

This will load and process a file from the directory Include.

See also Libraries.

1.20 Functions - inv

inv(expression) :

expression : real, matrix, equation

result : real, matrix, equation

Real : Return the invers of the real, which is simple 1/x

Matrix : Return the invers of the matrix, which is defined that m*inv(m) is the standart matrix.

1.21 Functions - ln/log

ln(expression) :

log(expression) :

expression : real, equation

result : real, equation

Returns the natural/common logarithm of the real.

1.22 Functions - matrix

matrix([list],[list],..) :

This will create a matrix object. [list] is a column of the matrix, and list is a list of expressions seperated by commas. The components must evaluate to real. E.g.

matrix([1,2,3],[4,5,6],[7,8,9])

```
/1 4 7\
|2 5 8|
\3 6 9/
```

1.23 Functions - multrow

multrow(matrix,r,expression) :

matrix : matrix (must be variable or parameter)

r : real (should be integer)

expression : real

result : matrix

This will copy the input matrix, except that row r will be replace by expression*(row r)

1.24 Functions - neg

neg(expression) :

expression : real, vector, matrix, equation

result : real, vector, matrix, equation

Returns the negative of the expression.

1.25 Functions - rang

rang(expression) :

expression : matrix, equation

result : real, equation

Returns the rang of the matrix. This is the number of non-zero rows of solve(expression).

1.26 Functions - sign

sign(expression) :

expression : real, equation

result : real, equation

Return 1 if expression is > 0

Return 0 if expression is = 0

Return -1 if expression is < 0

1.27 Functions - solve

solve(expression) :

expression : matrix

result : matrix

Returns the solution of the matrix. This is the result of basic matrix transformations to convert the left square of the matrix into a standart matrix. This usually make only sense for non-square matrices.

1.28 Functions - spur

spur(expression) :

expression : matrix, equation

result : real, equation

Returns the spur of the matrix.

1.29 Functions - sqrt

sqrt(expression) :

expression : real, equation

result : real, equation

Returns the square root of the expression.

1.30 Functions - swaprows

swaprows(matrix,r1,r2) :

matrix : matrix (must be variable or parameter)

r1 : real (should be integer)

r2 : real (should be integer)

result : matrix

This will copy the input matrix, except that row r1 and row r2 are swapped.

1.31 Functions - taylor

taylor(function,a,grade) :

This will calculate a taylor approximation for the given function of one variable at a of the given grade. If you omit the grade, a default of 2 will be used.

There a different kinds of syntax for this command :

1) Taylor approximation for an already defined function of one variable :

taylor(functionname, a, grade) - e.g.

f(x)=sin(x)

taylor(f, 0, 2)

2) Taylor approximation for an explicit given function of one variables :

taylor((expression)(var), a, grade) - e.g.

taylor((sin(x))(x) , 0, 2)

1.32 Functions - trans

trans(expression) :

expression : matrix, equation

result : matrix, equation

Returns the transposition of the matrix.

1.33 Functions - umatrix

umatrix(size) :

result : matrix

This will create a standart matrix of the given size.

1.34 Functions - vector

vector(list) :

This will create a vector object. list is a list of expressions seperated by commas. The components must evaluate to real. E.g.

vector(1,2,3)

/1\

|2|

\3/

1.35 Functions - window

window(name) :

This will open a window, displaying the named object.