Math-c Documentation

Operators

For the use of operator in quaternions, check the quaternion section.

a + b

sum

a -> scalar,string or matrix

b -> scalar,string or matrix

a - b

subtraction

a -> scalar,string or matrix

b -> scalar,string or matrix

a .* b

a • b //the operator is available in some keyboards as (ALT+8)

multiplication for each element of the matrix a to the same position of the matrix b(element by element multiplication)

a -> scalar or matrix

b -> scalar or matrix

a * b

linear multiplication matrix

a -> scalar or matrix

b -> scalar or matrix

a / b

division

a -> scalar

b -> scalar

a ./ b

division for each element of the matrix a to the same position of the matrix b(element by element division)

a -> scalar or matrix

b -> scalar or matrix

a % b

modulus, returns the remainder of the division a/b (a an b can be float numbers)

a -> scalar

b -> scalar

a .^ b

power for each element of the matrix a to the same position of the matrix b(element by element power)

a -> scalar or matrix

b -> scalar or matrix

a ^ b

power a to b

a -> scalar

b -> scalar

!a

not a, of is a is 0 return 1, else return 0

a -> scalar

~a

bitwise not

a -> scalar

a˚ //the operator is available in some keyboards as (ALT+K)

Conjugate

a -> scalar, vector or matrix

a'

Transpose and conjugate if is a matrix, and for a scalar only conjugate

a -> scalar, vector or matrix

a || b

or compare

a -> scalar

b -> scalar

a >= b

equal or greater than compare

a -> scalar

b -> scalar

a <= b

equal or less than compare

a -> scalar

b -> scalar

a > b

greater than compare

a -> scalar

b -> scalar

a < b

less than compare

a -> scalar

b -> scalar

a == b

equal compare

a -> scalar,string or matrix

b -> scalar,string or matrix

a != b

not equal compare

a -> scalar,string or matrix

b -> scalar,string or matrix

a && b

and compare

a -> scalar

b -> scalar

y = and(a,b)

y= a & b

bitwise and

a -> integer value

b -> integer value.

returns

y -> bitwise and, if a or b are not integer, the decimals are truncated to do the operation.

y = or(a,b)

y= a | b

bitwise or

a -> integer value

b -> integer value.

returns

y -> bitwise or, if a or b are not integer, the decimals are truncated to do the operation.

y = xor(a,b)

y= a ^^ b

bitwise xor

a -> integer value

b -> integer value.

returns

y -> bitwise xor, if a or b are not integer, the decimals are truncated to do the operation.

a << b

shift a, b times to the left

a -> integer value

b -> integer value.

a >> b

shift a, b times to the right

a -> integer value

b -> integer value.

Merge operators

A <-> B

Join two matrices horizontally, must have the same number of rows.

examples:

```
>>> M = [7 5 4;2 4 5]

M = [7 5 4 ;

2 4 5]

>>> D = [8;9]

D = [8 ;

9]

>>> M <-> D

ans = [7 5 4 8 ;

2 4 5 9]

>>> a = 7;        //scalar and vector
>>> g = [1 9 4];

>>> a <-> g

ans = [7 1 9 4];
```

A <|> B

Join two matrices vertically, must have the same number of cols. B will be added to bottom A

examples:

```
>>> M = [7 5 4;2 4 5]

M = [7 5 4 ;

2 4 5]

>>> D = [2 9 6]

>>> M <-> D

ans = [7 5 4 ;

2 4 5 ;

2 9 6]

>>> a = 7;

>>> g = [5; 2; 9];        //scalar and vector

>>> a <|> g

ans = [7 ;

5

2;

9]
```

Assignment operators

y += x

Addition assignment, equal to y = y+x

y -= x

subtraction assignment, equal to y = y-x

y *= x

Linear multiplication assignment, equal to y = y*x

y •= x

Multiplication assignment, equal to y = y•x

y /= x

Division assignment, equal to y = y/x

y %= x

Module assignment, equal to y = y%x