Qucs

Reference Manual

Measurement Expressions Reference Manual

Gunther Kraut

Introduction

This manual describes the measurement expressions available in "Qucs", the "Quite Universal Circuit Simulator".

Measurement expressions come into play whenever the results of a "Qucs" simulation run need post processing. Examples would be the conversion of a simulated voltage waveform from volts to dBV, the root mean square value of that waveform or the determination of the peak voltage. The "Qucs" measurement functions offer a rich set of data manipulation tools.

If you are not familiar with the way how to enter those formulas, please refer to chapter *"Using Measurement Expressions"*, which points out the possibilities to create and change measurement expressions. Also the data types supported are specified here. Chapter *"Functions Syntax and Overview"* introduces the basic syntax of functions and a categorical list of all functions available. The core of the document, a detailed compilation of all "Qucs" functions divided into different categories, is presented in chapter *"Math Functions"* and chapter *"Electronics Functions"*. Finally, the *Index* contains an alphabetical list of all functions.

Using Measurement Expressions

The chapter describes the usage of mathematical expressions for post processing simulation data in "Qucs", how to enter formulas and modifying them. It gives a brief description of the overall syntax of those expressions.

Entering Measurement Expressions

Measurement expressions generate new datasets by function or operator driven evaluation of simulation results. Those new datasets are accessible in the data display tab after simulation. The related equations can be entered into the schematic editor by the following means:

- Using the equation icon in the "Tools" bar (see fig. 1)
- Using menu item "Insert" → "Insert equation"

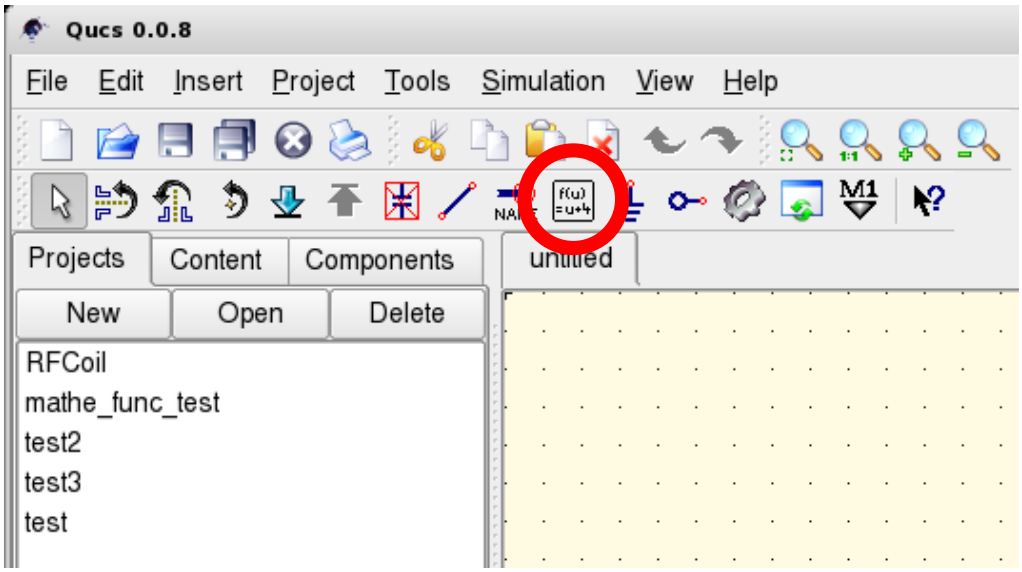

Figure 1: Entering a new measurement expression via equation icon

You can now place the equation symbol by mouse click anywhere in the schematic. Each mouse click creates a new equation instance each consisting of a variable number of measurement expressions. Press the Esc key if you do not like further equations.

Another option is to select an existing equation, copy it (either by menu item "Edit" → "Copy" or by Ctrl + C [1]) and paste it (either by menu item "Edit" → "Paste" or by Ctrl + V).

After having successfully created an equation instance, you are now able to modify it.

Changing Measurement Expressions

For sake of simplicity we assume that you have just generated a new equation - if you like to change an existing, more complicated equation the following steps are the same.

Thus, the excerpt of your schematic surface looks like that in fig. 2.

[1] Ctrl + C means that you have to press the Ctrl key and the C key simultaneously.

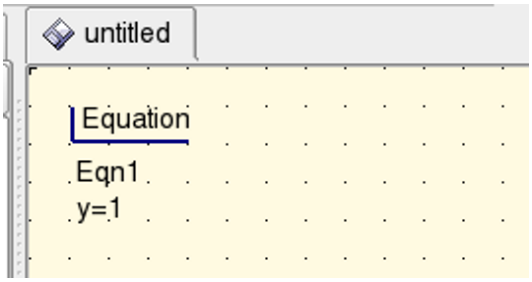

Figure 2: Newly created equation

You can now manipulate the current name of the equation instance. Simply click onto "Eqn1", which becomes highlighted. Then type in a new name for it and finalise your inputs with the Enter key.

After that, you can enter a new equation. Again, click onto "y=1". Only the "1" is marked, and you can enter a new expression there. Please use the variables, operators and constants described in chapter *"Syntax of Measurement Expressions"*. Note that you can also refer to results (dependents) of other equations. But how to change the name of the current dependent "y"? Right click onto the equation, and a context menu opens. Select the first item called "Edit properties". A sub window appears, which should look like the one in fig. 3. The alternative for entering equations is to double click onto the equation.

You can now change the name of the dependent, the equation itself (which is "1" in the example shown) and the name of the equation. If you do not want the result to be exported into the data display tab, but temporarily need it for further calculations, select "no" in the "Export value" cell.

Syntax of Measurement Expressions

Function names, variable names, and constant names are all case sensitive in measurement expressions - it is distinguished between lowercase and uppercase letters such as 'a' and 'A'.

In functions, commas are used to separate arguments.

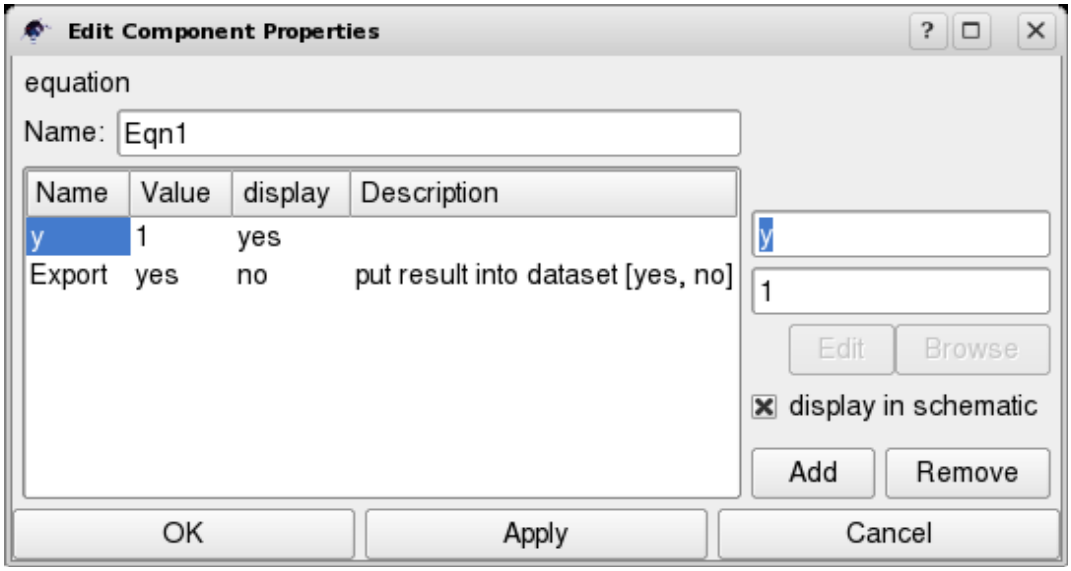

Figure 3: Editing equation properties

Variable Names

User defined variable names consist of a letter, followed by any number of letters, digits, or underscores.

The syntax of variable names created by the ”Qucs” simulator is as specified in table 1. Please note that all voltages and currents in “Qucs” are peak values except the noise voltages and currents which are RMS values at 1Hz bandwidth.

Numbers

Numbers are written in conventional decimal way, with an optional decimal point between the digits. For powers of ten, the familiar scientific notation with an ’e’ is used. In this way, ’1.234e6’ is an example for the real floating point number 1234000. Imaginary numbers can be entered by a multiplication factor ’i’ or ’j’ (see also table 3). An example would be ’1+2*i’ or - if you want to leave out the multiplication sign - ’1+i2’.
Beside the scientific ’e’ notation the following number suffixes can be used (see table 2):

Variable Name	Description
nodename.V	DC voltage at node *nodename*
name.I	DC current through circuit component *name*
nodename.v	AC voltage at node *nodename*
name.i	AC current through circuit component *name*
nodename.vn	AC noise voltage at node *nodename*
name.in	AC noise current through circuit component *name*
nodename.Vt	Transient voltage at node *nodename*
name.It	Transient current through circuit component *name*
name.OP	*name* = component name, OP = operating point (device dependent), e.g. D1.Id
S[x,y]	S-parameter, e.g. S[1,1]
Rn	equivalent noise resistance
Sopt	optimal reflection coefficient for minimum noise
Fmin	minimum noise figure
F	noise figure
nodename.Vb	Harmonic balance voltage at node *nodename*

Table 1: Syntax of simulator generated variable names

Vectors and Matrices

You can enter vectors and matrices manually by enclosing columns and rows into brackets. Columns are separated by commas, rows by semicolons. A valid matrix entry in a measurement expression would be 'A=[1,2;3,4]', defining the matrix $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$. The notation 'y=[1,2,3,4]' configures the vector $y = \begin{pmatrix} 1 & 2 & 3 & 4 \end{pmatrix}$. You get access to components of matrices and vectors by writing its name followed by brackets. Inside of the latter ranges (see table 6) or indices, separated by commas, define the extract you desire. Examples are 'y=M', accessing the whole matrix M, 'y=M[2,3]', extracting the value of the second row and third column of M, or 'y=M[:,3]', obtaining the complete third column.

Built-in Constants

The constants which can be used within measurement expressions are given in table 3.

Suffix	Name	Value	Suffix	Name	Value
E	exa	1E+18	m	milli	1E-3
P	peta	1E+15	u	micro	1E-6
T	tera	1E+12	n	nano	1E-9
G	giga	1E+9	p	pico	1E-12
M	mega	1E+6	f	femto	1E-15
k	kilo	1E+3	a	atto	1E-18

Table 2: Number Suffixes

Constant	Description	Value
e	Euler's constant	2.718282
i , j	Imaginary unit $\left(\sqrt{-1}\right)$	i1
kB	Boltzmann's constant	1.380658e23 J/K
pi	π	3.141593

Table 3: Built-in Constants

Operators

Operator Precedence Expressions are evaluated in the standard way, meaning from left to right, unless there are parentheses. The priority of operators is also handled familiarly, thus for example multiplication has precedence to addition. Tables 4 and 5 specify sorted lists of all operators, the topmost having highest priority. Operators on the same line have the same precedence.

Ranges The general nomenclature of ranges is displayed in table 6. It shows one-dimensional ranges, whereas also n-dimensional ranges are possible, if you consider nested sweeps.

Post Processing of Simulation Data by Expressions

After a simulation has run the results are stored in datasets. Usually, such a dataset is a vector or a matrix, but may also be a real or complex scalar. For transient analysis, this dataset contains voltage or current information over time, for Harmonic Balance it contains amplitudes at dedicated frequencies, while for S-parameter analysis a vector of matrices (thus matrices in dependency of frequency)

Operator	Description	Example
()	Parentheses, function call	`y=max(v)`
^	Exponentiation	`y=3^4`
*	Multiplication	`y=3*4`
/	Division	`y=3/4`
%	Modulo	`y=4%3`
+	Addition	`y=3+4`
-	Subtraction	`y=3-4`
:	Range operator	`y=v[3:12]`
+,-	Unary plus, unary minus	`y=+x z=-y`

Table 4: Arithmetic Operator Priorities

is returned. In further generalisation the components of vectors and matrices consist of complex numbers.

Additionally, datasets can be generated by using expressions. As an example the linspace() function shall be named, which creates a vector of linearly spaced elements.

Operator	Description	Example
()	Parentheses	`a=(x||y)&&z`
!	Negation	`z=!x`
? :	Abbreviation for conditional expression "if x then y else z"	`a=x?y:z`
&&	And	`z=x&&y`
||	Or	`z=x||y`
^^	Exclusive Or	`z=x^^y`
==	Equal	`z=x==y`
!=	Not equal	`z=x!=y`
<	Less than	`z=x<y`
<=	Less than or equal	`z=x<=y`
>	Larger than	`z=x>y`
>=	Larger than or equal	`z=x>=y`

Table 5: Logical Operator Priorities

Syntax	Explanation
m:n	Range from index m to index n
:n	Range up to index n
m:	Range starting from index m
:	No range limitations

Table 6: Range definition

Functions Syntax and Overview

This chapter introduces the basic syntax of the function descriptions and contains a categorical list of all available functions.

Functions Reference Format

"Qucs" provides a rich set of functions, which can be used to generate and display new datasets by function based evaluation of simulation results. Beside a large number of mathematical standard functions such as square root (sqrt), exponential function (exp), absolute value (abs), functions especially useful for calculation and transformation of electronic values are implemented. Examples for the latter would be the conversion from Watts to dBm, the generation of noise circles in an amplifier

design, or the conversion from S-parameters to Y-parameters.

Functions Reference Format

In the subsequent two chapters, each function is described using the following structure:

<Function Name>

Outlines briefly the functionality of the function.

Syntax

Defines the general syntax of this function.

Arguments

Name, type, definition range and whether the argument is optional, are tabulated here. In case of an optional parameter the default value is specified. "Type" is a list defining the arguments allowed and may contain the following symbols:

Symbol	Description
$\mathbb{R}$	Real number
$\mathbb{C}$	Complex number
$\mathbb{R}^n$	Vector consisting of n real elements
$\mathbb{C}^n$	Vector consisting of n complex elements
$\mathbb{R}^{m\times n}$	Real matrix consisting of m rows and n columns
$\mathbb{C}^{m\times n}$	Complex matrix consisting of m rows and n columns
$\mathbb{R}^{m\times n\times p}$	Vector of p real $m \times n$ matrices
$\mathbb{C}^{m\times n\times p}$	Vector of p complex $m \times n$ matrices

"Definition range" specifies the allowed range. Each range is introduced by a bracket, either "[" or "]", meaning that the following start value of the range is either included or excluded. The start value is separated from the end value by a comma. Then the end value follows, finished by a bracket again, either "[" or "]".

The first bracket mentioned means “excluding the end value”, the second means “including”.

If a range is given for a complex number, this specifies the real or imaginary value of that number. If a range is given for a real or complex vector or matrix, this specifies the real or imaginary value of each element of that vector or matrix. The symbols mean “includes listed value” and “excludes listed value”.

Description

Gives a more detailed description on what the function does and what it returns. In case some background knowledge is presented.

Examples

Shows an application of the function by one or several simple examples.

See also

Shows links to related functions. A mouse click onto the desired link leads to an immediate jump to that function.

Functions Listed by Category

This compilation shows all “Qucs” functions sorted by category (an alphabetical list is given in the appendix). Please click on the desired function to go to its detailed description.

Math Functions

Vectors and Matrices: Creation

eye()	...	Creates n x n identity matrix
linspace()	...	Creates a real vector with linearly spaced components
logspace()	...	Creates a real vector with logarithmically spaced components

Vectors and Matrices: Basic Matrix Functions

adjoint()	...	Adjoint matrix
array()	...	Read out single elements
det()	...	Determinant of a matrix
inverse()	...	Matrix inverse
transpose()	...	Matrix transpose
length()	...	Length of a vector

Elementary Mathematical Functions: Basic Real and Complex Functions

abs()	...	Absolute value
angle()	...	Phase angle in radians of a complex number. Synonym for "arg"
arg()	...	Phase angle in radians of a complex number
conj()	...	Conjugate of a complex number
deg2rad()	...	Converts phase from degrees into radians
hypot()	...	Euclidean distance function
imag()	...	Imaginary value of a complex number
mag()	...	Magnitude of a complex number
norm()	...	Square of the absolute value of a vector
phase()	...	Phase angle in degrees of a complex number
polar()	...	Transform from polar coordinates into complex number
rad2deg()	...	Converts phase from degrees into radians
real()	...	Real value of a complex number
signum()	...	Signum function
sign()	...	Sign function
sqr()	...	Square of a number
sqrt()	...	Square root
unwrap()	...	Unwraps a phase vector in radians

Elementary Mathematical Functions: Exponential and Logarithmic Functions

exp()	...	Exponential function
limexp()	...	Limited exponential function
log10()	...	Decimal logarithm
log2()	...	Binary logarithm
ln()	...	Natural logarithm (base e)

Elementary Mathematical Functions: Trigonometry

cos()	...	Cosine function
cosec()	...	Cosecant
cot()	...	Cotangent function
sec()	...	Secant
sin()	...	Sine function
tan()	...	Tangent function

Elementary Mathematical Functions: Inverse Trigonometric Functions

arccos()	...	Arc cosine (also known as "inverse cosine")
arccosec()	...	Arc cosecant (also known as "inverse cosecant")
arccot()	...	Arc cotangent
arcsec()	...	Arc secant (also known as "inverse secant")
arcsin()	...	Arc sine (also known as "inverse sine")
arctan()	...	Arc tangent (also known as "inverse tangent")

Elementary Mathematical Functions: Hyperbolic Functions

cosh()	...	Hyperbolic cosine
cosech()	...	Hyperbolic cosecant
coth()	...	Hyperbolic cotangent
sech()	...	Hyperbolic secant
sinh()	...	Hyperbolic sine
tanh()	...	Hyperbolic tangent

Elementary Mathematical Functions: Inverse Hyperbolic Functions

arcosh()	...	Hyperbolic area cosine
arcosech()	...	Hyperbolic area cosecant
arcoth()	...	Hyperbolic area cotangent
arsech()	...	Hyperbolic area secant
arsinh()	...	Hyperbolic area sine
artanh()	...	Hyperbolic area tangent

Elementary Mathematical Functions: Rounding

ceil()	...	Round to the next higher integer
fix()	...	Truncate decimal places from real number
floor()	...	Round to the next lower integer
round()	...	Round to nearest integer

Elementary Mathematical Functions: Special Mathematical Functions

besseli0()	...	Modified Bessel function of order zero
besselj()	...	Bessel function of n-th order
bessely()	...	Bessel function of second kind and n-th order
erf()	...	Error function
erfc()	...	Complementary error function
erfinv()	...	Inverse error function
erfcinv()	...	Inverse complementary error function
sinc()	...	Sinc function
step()	...	Step function

Data Analysis: Basic Statistics

avg()	...	Average of vector elements
cumavg()	...	Cumulative average of vector elements
max()	...	Maximum value
min()	...	Minimum value
rms()	...	Root Mean Square of vector elements
runavg()	...	Running average of vector elements
stddev()	...	Standard deviation of vector elements
variance()	...	Variance of vector elements
random()	...	Random number between 0.0 and 1.0
srandom()	...	Set seed for a new series of pseudo-random numbers

Data Analysis: Basic Operation

cumprod()	...	Cumulative product of vector elements
cumsum()	...	Cumulative sum of vector elements
interpolate()	...	Equidistant spline interpolation of data vector
prod()	...	Product of vector elements
sum()	...	Sum of vector elements
xvalue()	...	Returns x-value which is associated with the y-value nearest to a specified y-value in a given vector
yvalue()	...	Returns y-value of a given vector which is located nearest to the specified x-value

Data Analysis: Differentiation and Integration

ddx()	...	Differentiate mathematical expression with respect to a given variable
diff()	...	Differentiate vector with respect to another vector
integrate()	...	Integrate vector

Data Analysis: Signal Processing

dft()	...	Discrete Fourier Transform
fft()	...	Fast Fourier Transform
idft()	...	Inverse Discrete Fourier Transform
ifft()	...	Inverse Fast Fourier Transform
fftshift()	...	Move the frequency 0 to the center of the FFT vector
Time2Freq()	...	Interpreted Discrete Fourier Transform
Freq2Time()	...	Interpreted Inverse Discrete Fourier Transform
kbd()	...	Kaiser-Bessel derived window

Electronics Functions

Unit Conversion

dB()	...	dB value
dbm()	...	Convert voltage to power in dBm
dbm2w()	...	Convert power in dBm to power in Watts
w2dbm()	...	Convert power in Watts to power in dBm

Reflection Coefficients and VSWR

rtoswr()	...	Converts reflection coefficient to voltage standing wave ratio (VSWR)
rtoy()	...	Converts reflection coefficient to admittance
rtoz()	...	Converts reflection coefficient to impedance
ytor()	...	Converts admittance to reflection coefficient
ztor()	...	Converts impedance to reflection coefficient

N-Port Matrix Conversions

stos()	...	Converts S-parameter matrix to S-parameter matrix with different reference impedance(s)
stoy()	...	Converts S-parameter matrix to Y-parameter matrix
stoz()	...	Converts S-parameter matrix to Z-parameter matrix
twoport()	...	Converts a two-port matrix from one representation into another
ytos()	...	Converts Y-parameter matrix to S-parameter matrix
ytoz()	...	Converts Y-parameter matrix to Z-parameter matrix
ztos()	...	Converts Z-parameter matrix to S-parameter matrix
ztoy()	...	Converts Z-parameter matrix to Y-parameter matrix

Amplifiers

GaCircle()	...	Circle(s) with constant available power gain Ga in the source plane
GpCircle()	...	Circle(s) with constant operating power gain Gp in the load plane
Mu()	...	Mu stability factor of a two-port S-parameter matrix
Mu2()	...	Mu' stability factor of a two-port S-parameter matrix
NoiseCircle()	...	Generates circle(s) with constant Noise Figure(s)
PlotVs()	...	Returns a data item based upon vector or matrix vector with dependency on a given vector
Rollet()	...	Rollet stability factor of a two-port S-parameter matrix
StabCircleL()	...	Stability circle in the load plane
StabCircleS()	...	Stability circle in the source plane
StabFactor()	...	Stability factor of a two-port S-parameter matrix. Synonym for Rollet()
StabMeasure()	...	Stability measure B1 of a two-port S-parameter matrix
vt()	...	Thermal voltage for a given temperature in Kelvin

Math Functions

Vectors and Matrices

Creation

eye()

Creates n x n identity matrix.

Syntax

y=eye(n)

Arguments

Name	Type	Def. Range	Required
n	$\mathbb{N}$	$[1, +\infty[$	$\surd$

Description

This function creates the n x n identity matrix, that is

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & 0 & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}$$

Example

`y=eye(2)` returns

1	0
0	1

.

See also

linspace()

Creates a real vector with linearly spaced components.

Syntax

y=linspace(xs,xe,n)

Arguments

Name	Type	Def. Range	Required
xs	$\mathbb{R}$	$]-\infty,+\infty[$	$\surd$
xe	$\mathbb{R}$	$]-\infty,+\infty[$	$\surd$
n	$\mathbb{N}$	$[2,+\infty[$	$\surd$

Description

This function creates a real vector with n linearly spaced components. The first component is xs, the last one is xe.

Example

`y=linspace(1,2,3)` returns 1, 1.5, 2.

See also

logspace()

logspace()

Creates a real vector with logarithmically spaced components.

Syntax

y=logspace(xs,xe,n)

Arguments

Name	Type	Def. Range	Required
xs	$\mathbb{R}$	$]-\infty,+\infty[$	$\surd$
xe	$\mathbb{R}$	$]-\infty,+\infty[$	$\surd$
n	$\mathbb{N}$	$[2,+\infty[$	$\surd$

Description

This function creates a real vector with n logarithmically spaced components. The first component is xs, the last one is xe.

Example

`y=logspace(1,2,3)` returns 1, 1.41, 2.

See also

linspace()

Basic Matrix Functions

adjoint()

Adjoint matrix.

Syntax

Y=adjoint(X)

Arguments

Name	Type	Def. Range	Required
X	$\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the adjoint matrix Y of a matrix X:

$Y = X^H = (X^*)^T$, where X^* is the complex conjugate matrix of X and X^T is the transposed of the matrix X.

Example

`X=eye(2)*(3+i)` returns

3+j1	0
0	3+j1

. Then,

`Y=adjoint(X)` returns

3-j1	0
0	3-j1

.

See also

transpose(), conj()

array()

Read out single elements.

Syntax

The “array()” function is an implicit command. Thus normally the respective first expression (”preferred”) is used.

Syntax	Preferred	Alternative	Preferred	Alternative
1	y=VM[i,j]	y=array(VM,i,j)		
2	y=M[i,j]	y=array(M,i,j)		
3	y=VM[k]	y=array(VM,k)		
4	y=v[i]	y=array(v,i)	y=v[r]	y=array(v,r)
5	y=v[i,r]	y=array(v,i,r)	y=v[r,j]	y=array(v,r,j)
	y=v[i,j]	y=array(v,i,j)	y=v[r1,r2]	y=array(v,r1,r2)
6	y=s[i]	y=array(s,i)		

Arguments

Name	Type	Def. Range	Required
VM	$\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$(Syntax 1 and 3)
M	$\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$	$]-\infty,+\infty[$	$\surd$(Syntax 2)
v	$\mathbb{R}^{n}$,$\mathbb{C}^{n}$	$]-\infty,+\infty[$	$\surd$(Syntax 4 and 5)
r, r1, r2	Range$xs:xe$	$0\leq xs\leq n-1$, $xs\leq xe\leq n-1$	$\surd$(Syntax 4 and 5)
i	$\mathbb{N}$	$0\leq i\leq m-1$	$\surd$(Syntax 1, 2, 4, 5, 6)
j	$\mathbb{N}$	$0\leq j\leq n-1$	$\surd$(Syntax 1, 2, 5)
k	$\mathbb{N}$	$0\leq k\leq p-1$	$\surd$(Syntax 3)
s	String	Arbitrary characters	$\surd$(Syntax 6)

Description

This function reads out real or complex vectors of matrices, matrices and vectors or strings. Please refer to the following table for the return values:

Syntax	Argument 1	Argument 2	Argument 3	Result
y=VM[i,j]	$VM = (x_{ijk})$	$i \in \mathbb{N}$	$j \in \mathbb{N}$	Vector $(x_{ij1}, \cdots, x_{ijK})$
y=M[i,j]	$M = (x_{ij})$	$i \in \mathbb{N}$	$j \in \mathbb{N}$	Number x_{ij}
y=VM[k]	$VM = (x_{ijk})$	$k \in \mathbb{N}$		Matrix $\begin{pmatrix} x_{11k} & \cdots & x_{1nk} \\ \vdots & \ddots & \vdots \\ x_{m1k} & \cdots & x_{mnk} \end{pmatrix}$
y=v[i]	$v = (v_i)$	$i \in \mathbb{N}$		Number v_i
y=v[xs:xe]	$v = (v_i)$	$xs, \ldots, xe$		Vector $(v_{xs}, \cdots, v_{xe})$
y=v[i,xs:xe]	$v = (v_i)$	$i \in \mathbb{N}$	$xs, \ldots, xe$	Vector $(v_{xs}, \cdots, v_{xe})$
y=v[xs:xe,j]	$v = (v_i)$	$xs, \ldots, xe$	$xs, \ldots, xe$	Vector $(v_{xs}, \cdots, v_{xe})$
y=v[i,j]	$v = (v_i)$	$i \in \mathbb{N}$	$xs, \ldots, xe$	Vector $(v_{xs}, \cdots, v_{xe})$
y=v[xs1:xe1, xs2:xe2]	$v = (v_i)$	$xs1, \ldots, xe1$	$xs2, \ldots, xe2$	Vector $(v_{xs}, \cdots, v_{xe})$
y=s[i]	$s = (s_i)$	$i \in \mathbb{N}$		Character s_i

Again, v denotes a vector, M a matrix, VM a vector of matrices, s a vector of characters and *xs, xs1, xs2, xe, xe1, xe2* are range limiters.

Example

`v=linspace(1,2,4)` returns 1, 1.33, 1.67, 2. Then,

`y=v[3]` returns 2.

See also

det()

Determinant of a matrix.

Syntax

y=det(X)

Arguments

Name	Type	Def. Range	Required
X	$\mathbb{R}^{n\times n}$,$\mathbb{C}^{n\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the determinant of a quadratical n x n matrix X. The result is either a real or a complex number.

Example

`X=eye(2)*3` returns $\begin{array}{|c|c|}\hline 3 & 0 \\ \hline 0 & 3 \\ \hline \end{array}$. Then,

`y=det(X)` returns 9.

See also

eye()

inverse()

Matrix inverse.

Syntax

Y=inverse(X)

Arguments

Name	Type	Def. Range	Required
X	$\mathbb{R}^{n\times n}$,$\mathbb{C}^{n\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function inverts a quadratical n x n matrix X. The generated inverted matrix Y fulfills the equation

$X \cdot Y = X \cdot X^{-1} = 1$, where " $\cdot$ " denotes matrix multiplication and "1" the identity matrix.

The matrix X must be regular, that means that its determinant $\Delta \neq 0$.

Example

`X=eye(2)*3` returns $\begin{array}{|c|c|}\hline 3 & 0 \\ \hline 0 & 3 \\ \hline \end{array}$. Then,

`Y=inverse(X)` returns $\begin{array}{|c|c|}\hline 0.333 & 0 \\ \hline 0 & 0.333 \\ \hline \end{array}$.

See also

transpose(), eye(), det()

transpose()

Matrix transpose.

Syntax

Y=transpose(X)

Arguments

Name	Type	Def. Range	Required
X	$\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function transposes a m x n matrix X, which is equivalent to exchanging rows and columns according to

$Y = X^T = (x_{ij})^T = (x_{ji})$ with $1 \leq i \leq m,\ 1 \leq j \leq n$

The generated matrix Y is a n x m matrix.

Example

`X=eye(2)*3` returns $\begin{array}{|c|c|}\hline 3 & 0 \\ \hline 0 & 3 \\ \hline \end{array}$. Then,

`Y=transpose(X)` returns $\begin{array}{|c|c|}\hline 3 & 0 \\ \hline 0 & 3 \\ \hline \end{array}$.

See also

eye(), inverse()

length()

Length of a vector.

Syntax

y=length(v)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$ $]-\infty,+\infty[$	$\surd$	

Description

This function returns the length of vector v.

Example

`length(linspace(1,2,3))` returns 3.

See also

Elementary Mathematical Functions

Basic Real and Complex Functions

abs()

Absolute value.

Syntax

y=abs(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the absolute value of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = \begin{cases} x & for\, x \geq 0 \\ -x\, for\, x < 0 \end{cases}$

For $\mathbb{C} \ni x := a + i\,b \wedge a, b \in \mathbb{R}$: $y = \sqrt{a^2 + b^2}$

For x being a vector or a matrix the two equations above are applied to the components of x.

Examples

`y=abs(-3)` returns 3,

`y=abs(-3+4*i)` returns 5.

See also

mag(), norm(), real(), imag(), conj(), phase(), arg(), hypot()

angle()

Phase angle in radians of a complex number. Synonym for "arg".

Syntax

y=angle(x)

See also

abs(), mag(), norm(), real(), imag(), conj(), phase(), arg()

arg()

Phase angle in radians of a complex number.

Syntax

y=arg(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the phase angle in degrees of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = \begin{cases} 0 & for\, x \geq 0 \\ \pi & for\, x < 0 \end{cases}$

For $\mathbb{C} \ni x := a + i\,b \;\wedge\; a, b \in \mathbb{R}$:

Definition range	Result
$a > 0,\ b > 0$	$y = \arctan\left(\frac{b}{a}\right)$
$a < 0,\ b > 0$	$y = \arctan\left(\frac{b}{a}\right) + \pi$
$a < 0,\ b < 0$	$y = \arctan\left(\frac{b}{a}\right) - \pi$
$a > 0,\ b < 0$	$y = \arctan\left(\frac{b}{a}\right)$
$a = 0,\ b > 0$	$y = \frac{\pi}{2}$
$a > 0,\ b > 0$	$y = -\frac{\pi}{2}$
$a = 0,\ b = 0$	$y = 0$

In this case the arctan() function returns values in radians. The result y of the phase function is in the range $[-\pi,\ +\pi]$. For x being a vector or a matrix the two equations above are applied to the components of x.

Examples

`y=arg(-3)` returns 3.14,

`y=arg(-3+4*i)` returns 2.21.

See also

abs(), mag(), norm(), real(), imag(), conj(), phase()

conj()

Conjugate of a complex number.

Syntax

y=conj(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the conjugate of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = x$

For $\mathbb{C} \ni x := a + i\,b \wedge a, b \in \mathbb{R}$: $y = a - i\,b$

For x being a vector or a matrix the two equations above are applied to the components of x.

Example

`y=conj(-3+4*i)` returns -3-4*i.

See also

abs(), mag(), norm(), real(), imag(), phase(), arg()

deg2rad()

Converts phase from degrees into radians.

Syntax

y=deg2rad(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function converts a real phase, a complex phase or a phase vector given in degrees into radians.

For $x \in \mathbb{R}$: $y = \frac{\pi}{180}\, x$

For $x \in \mathbb{C}$: $y = \frac{\pi}{180}\, Re\,\{x\}$

For x being a vector the two equations above are applied to the components of x.

Example

`y=deg2rad(45)` returns 0.785.

See also

rad2deg(), phase(), arg()

hypot()

Euclidean distance function.

Syntax

z=hypot(x,y)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
y	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the Euclidean distance z between two real or complex numbers or vectors. For two numbers $x, y \in \mathbb{C}$, this is

$$z = \sqrt{|x|^2 + |y|^2}$$

For x, y being vectors (of same size) the equation above is applied componentwise.

Examples

`z=hypot(3,4)` returns 5,

`z=hypot(1+2*i,1-2*i)` returns 3.16.

See also

abs()

imag()

Imaginary value of a complex number.

Syntax

y=imag(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the imaginary value of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = 0$

For $\mathbb{C} \ni x := a + i\,b \ \wedge\ a, b \in \mathbb{R}$: $y = b$

For x being a vector or a matrix the two equations above are applied to the components of x.

Example

`y=imag(-3+4*i)` returns 4.

See also

abs(), mag(), norm(), real(), conj(), phase(), arg()

mag()

Magnitude of a complex number.

Syntax

y=mag(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the magnitude (absolute value) of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = \begin{cases} x & for\, x \geq 0 \\ -x & for\, x < 0 \end{cases}$

For $\mathbb{C} \ni x := a + i\,b \;\wedge\; a, b \in \mathbb{R}$: $y = \sqrt{a^2 + b^2}$

For x being a vector or a matrix the two equations above are applied to the components of x.

Examples

`y=mag(-3)` returns 3,

`y=mag(-3+4*i)` returns 5.

See also

abs(), norm(), real(), imag(), conj(), phase(), arg()

norm()

Square of the absolute value of a vector.

Syntax

y=norm(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the square of the absolute value of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = x^2$

For $\mathbb{C} \ni x := a + i\,b \ \wedge\ a, b \in \mathbb{R}$: $y = a^2 + b^2$

For x being a vector or a matrix the two equations above are applied to the components of x.

Example

`y=norm(-3+4*i)` returns 25.

See also

abs(), mag(), real(), imag(), conj(), phase(), arg()

phase()

Phase angle in degrees of a complex number.

Syntax

y=phase(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the phase angle in degrees of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = \begin{cases} 0 & for\, x \geq 0 \\ 180 & for\, x < 0 \end{cases}$

For $\mathbb{C} \ni x := a + i\,b \;\wedge\; a, b \in \mathbb{R}$:

Definition range	Result
$a > 0,\ b > 0$	$y = \arctan\left(\frac{b}{a}\right)$
$a < 0,\ b > 0$	$y = \arctan\left(\frac{b}{a}\right) + 180$
$a < 0,\ b < 0$	$y = \arctan\left(\frac{b}{a}\right) - 180$
$a > 0,\ b < 0$	$y = \arctan\left(\frac{b}{a}\right)$
$a = 0,\ b > 0$	$y = 90$
$a > 0,\ b > 0$	$y = -90$
$a = 0,\ b = 0$	$y = 0$

In this case the arctan() function returns values in degrees. The result y of the phase function is in the range $[-180,\ +180]$. For x being a vector or a matrix the two equations above are applied to the components of x.

Examples

`y=phase(-3)` returns 180,

`y=phase(-3+4*i)` returns 127.

See also

abs(), mag(), norm(), real(), imag(), conj(), arg()

polar()

Transform from polar coordinates into complex number.

Syntax

c=polar(a,p)

Arguments

Name	Type	Def. Range	Required
a	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
p	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function transforms a point given in polar coordinates (amplitude a and phase p in degrees) in the complex plane into the corresponding complex number:

$x + i\,y = a\,e^{ip} = a\,\cos p + i\,a\,\sin p$

For a or p being vectors the equation above is applied to the components of a or p.

Example

`c=polar(3,45)` returns 2.12+j2.12.

See also

abs(), mag(), norm(), real(), imag(), conj(), phase(), arg(), exp(), cos(), sin()

rad2deg()

Converts phase from degrees into radians.

Syntax

y=rad2deg(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function converts a real phase, a complex phase or a phase vector given in radians into degrees.

For $x \in \mathbb{R}$: $y = \dfrac{180}{\pi}\, x$

For $x \in \mathbb{C}$: $y = \dfrac{180}{\pi}\, Re\,\{x\}$

For x being a vector the two equations above are applied to the components of x.

Example

`y=rad2deg(45)` returns 0.785.

See also

deg2rad(), phase(), arg()

real()

Real value of a complex number.

Syntax

y=real(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n}$,$\mathbb{C}^{m\times n}$,$\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the real value of a real or complex number, vector or matrix.

For $x \in \mathbb{R}$: $y = x$

For $\mathbb{C} \ni x := a + i\,b \wedge a, b \in \mathbb{R}$: $y = a$

For x being a vector or a matrix the two equations above are applied to the components of x.

Example

`y=real(-3+4*i)` returns -3.

See also

abs(), mag(), norm(), imag(), conj(), phase(), arg()

signum()

Signum function.

Syntax

y=signum(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the sign of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \begin{cases} 1 & for\, x > 0 \\ 0 & for\, x = 0 \\ -1 & for\, x < 0 \end{cases}$

For $x \in \mathbb{C}$: $y = \begin{cases} \dfrac{x}{|x|} & for\, x \neq 0 \\ 0 & for\, x = 0 \end{cases}$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=signum(-4)` returns -1,

`y=signum(3+4*i)` returns 0.6+j0.8.

See also

abs(), sign()

sign()

Sign function.

Syntax

y=sign(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the sign of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \begin{cases} 1 & for\ x >= 0 \\ -1 & for\ x < 0 \end{cases}$

For $x \in \mathbb{C}$: $y = \begin{cases} \frac{x}{\lvert x \rvert} & for\ x \neq 0 \\ 1 & for\ x = 0 \end{cases}$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=sign(-4)` returns -1,

`y=sign(3+4*i)` returns 0.6+j0.8.

See also

abs(), signum()

sqr()

Square of a number.

Syntax

y=sqr(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty, +\infty[$	$\surd$

Description

This function calculates the square root of a real or complex number or vector.

$y = x^2$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=sqr(-4)` returns 16,

`y=sqr(3+4*i)` returns -7+j24.

See also

sqrt()

sqrt()

Square root.

Syntax

y=sqrt(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the square root of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \begin{cases} \sqrt{x} \quad for\, x \geq 0 \\ i\sqrt{-x}\, for\, x < 0 \end{cases}$

For $x \in \mathbb{C}$: $y = \sqrt{|x|}\, e^{i\frac{\varphi}{2}}$ with $\varphi = \arg(x)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=sqrt(-4)` returns 0+j2,

`y=sqrt(3+4*i)` returns 2+j1.

See also

sqr()

unwrap()

Unwraps a phase vector in radians.

Syntax

y=unwrap(x)

y=unwrap(x, t)

Arguments

Name	Type	Def. Range	Required	Default
x	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$	
t	$\mathbb{R}$	$]-\infty,+\infty[$		π

Description

This function unwraps a phase vector x to avoid phase jumps. If two consecutive values of x differ by more than tolerance t, $\mp 2\pi$(depending on the sign of the difference) is added to the current element of x. The predefined value of the optional parameter t is π.

Examples

`y=unwrap(3.15*linspace(-2,2,5))` returns -6.3, -9.43, -12.6, -15.7, -18.8,

`y=unwrap(2*linspace(-2,2,5),1)` returns -4, -8.28, -12.6, -16.8, -21.1,

`y=unwrap(2*linspace(-2,2,5),3)` returns -4, -2, 0, 2, 4.

See also

abs(), mag(), norm(), real(), imag(), conj(), phase(), arg()

Exponential and Logarithmic Functions

exp()

Exponential function.

Syntax

y=exp(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the exponential function of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = e^x$

For $\mathbb{C} \ni x := a + i\,b \wedge a, b \in \mathbb{R}$: $y = e^x = e^{a+i\,b} = e^a\ (\cos b + i\ \sin b)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=exp(-4)` returns 0.0183,

`y=exp(3+4*i)` returns -13.1-j15.2.

See also

limexp(), ln(), log10(), log2(), cos(), sin()

limexp()

Limited exponential function.

Syntax

y=limexp(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function is equivalent to the exponential function $\exp(x)$, as long as $x <= 80$. For larger arguments x, it limits the result to $y = \exp(80) \cdot (1 + x - 80)$. The argument can be a real or complex number or vector.

For $x \in \mathbb{R}$: $y = e^x$ for $x \leq 80$, $y = e^{80} \cdot (1 + x - 80)$ else.

For $\mathbb{C} \ni x := a + i\,b \wedge a, b \in \mathbb{R}$: $y = \text{limexp}(x) = \text{limexp}(a + i\,b) = \text{limexp}(a)\ (\cos b + i\ \sin b)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=limexp(81)` returns 1.11e+35, whereas `y=exp(81)` returns 1.51e+35, which shows the limiting effect of the limexp() function.

`y=limexp(3+4*i)` returns -13.1-j15.2.

See also

exp(), ln(), log10(), log2(), cos(), sin()

log10()

Decimal logarithm.

Syntax

y=log10(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus \{0\}$	$\surd$

Description

This function calculates the principal value of the decimal logarithm (base 10) of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \begin{cases} \dfrac{\ln(x)}{\ln(10)} & for\ x > 0 \\ \dfrac{\ln(-x)}{\ln(10)} + i\dfrac{\pi}{\ln(10)} & for\ x < 0 \end{cases}$

For $x \in \mathbb{C}$: $y = \dfrac{\ln(|x|)}{\ln(10)} + i\dfrac{\arg(x)}{\ln(10)}$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=log10(-4)` returns 0.602+j1.36,

`y=log10(3+4*i)` returns 0.699+j0.403.

See also

ln(), log2(), exp(), arg()

log2()

Binary logarithm.

Syntax

y=log2(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus \{0\}$	$\surd$

Description

This function calculates the principal value of the binary logarithm (base 2) of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \begin{cases} \dfrac{\ln(x)}{\ln(2)} & for\ x > 0 \\ \dfrac{\ln(-x)}{\ln(2)} + i\dfrac{\pi}{\ln(2)} & for\ x < 0 \end{cases}$

For $x \in \mathbb{C}$: $y = \dfrac{\ln(|x|)}{\ln(2)} + i\dfrac{\arg(x)}{\ln(2)}$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=log2(-4)` returns 2+j4.53,

`y=log2(3+4*i)` returns 2.32+j1.34.

See also

ln(), log10(), exp(), arg()

ln()

Natural logarithm (base e).

Syntax

y=ln(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus \{0\}$	$\surd$

Description

This function calculates the principal value of the natural logarithm (base e) of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \begin{cases} \ln(x) & for\ x > 0 \\ \ln(-x) & for\ x < 0 \end{cases}$

For $x \in \mathbb{C}$: $y = \ln(|x|) + i\ \arg(x)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=ln(-4)` returns 1.39+j3.14,

`y=ln(3+4*i)` returns 1.61+j0.927.

See also

log2(), log10(), exp(), arg()

Trigonometry

cos()

Cosine function.

Syntax

y=cos(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty, +\infty[$	$\surd$

Description

This function calculates the cosine of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \cos(x)$ with $y \in [-1,\, 1]$

For $x \in \mathbb{C}$: $y = \frac{1}{2}\,(\exp(i\,x) + \exp(-i\,x))$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=cos(-0.5)` returns 0.878,

`y=cos(3+4*i)` returns -27.0-j3.85.

See also

sin(), tan(), arccos()

cosec()

Cosecant.

Syntax

y=cosec(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus \{k\pi\},\ k \in \mathbb{Z}$	$\surd$

Description

This function calculates the cosecant of a real or complex number or vector.

$$y = \operatorname{cosec} x = \frac{1}{\sin\ x}$$

For x being a vector the equation above is applied to the components of x.

Example

`y=cosec(1)` returns 1.19.

See also

sin(), sec()

cot()

Cotangent function.

Syntax

y=cot(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus \{k\pi\},\ k \in \mathbb{Z}$	$\surd$

Description

This function calculates the cotangent of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \dfrac{1}{\tan(x)}$ with $y \in [-\infty, +\infty]$

For $x \in \mathbb{C}$: $y = i\left(\dfrac{\exp(i\,x)^2 + 1}{\exp(i\,x)^2 - 1}\right)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=cot(-0.5)` returns -1.83,

`y=cot(3+4*i)` returns -0.000188-j1.

See also

tan(), sin(), cos(), arctan(), arccot()

sec()

Secant.

Syntax

y=sec(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus \left\{\left(k+\frac{1}{2}\right)\pi\right\},\ k\in\mathbb{Z}$	$\surd$

Description

This function calculates the secant of a real or complex number or vector.

$$y = \sec\ x = \frac{1}{\cos\ x}$$

For x being a vector the equation above is applied to the components of x.

Example

`y=sec(0)` returns 1.

See also

cos(), cosec()

sin()

Sine function.

Syntax

y=sin(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty, +\infty[$	$\surd$

Description

This function calculates the sine of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \sin(x)$ with $y \in [-1,\, 1]$

For $x \in \mathbb{C}$: $y = \frac{1}{2}\, i\, (\exp(-i\, x) - \exp(i\, x))$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=sin(-0.5)` returns -0.479,

`y=sin(3+4*i)` returns 3.85-j27.

See also

cos(), tan(), arcsin()

tan()

Tangent function.

Syntax

y=tan(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus \left\{\left(k+\frac{1}{2}\right)\pi\right\},\ k\in\mathbb{Z}$	$\surd$

Description

This function calculates the tangent of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \tan(x)$ with $y \in [-\infty, +\infty]$

For $x \in \mathbb{C}$: $y = -i\left(\frac{\exp(i\,x)^2 - 1}{\exp(i\,x)^2 + 1}\right)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=tan(-0.5)` returns -0.546,

`y=tan(3+4*i)` returns -0.000187+j0.999.

See also

cot(), sin(), cos(), arctan(), arccot()

Inverse Trigonometric Functions

arccos()

Arc cosine (also known as "inverse cosine").

Syntax

y=arccos(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$[-1, +1]$	$\surd$

Description

This function calculates principal value of the the arc cosine of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \arccos(x)$ with $y \in [0,\, \pi]$

For $x \in \mathbb{C}$: $y = -i \ln\left(x + \sqrt{x^2 - 1}\,\right)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=arccos(-1)` returns 3.14,

`y=arccos(3+4*i)` returns 0.937-j2.31.

See also

cos(), arcsin(), arctan(), arccot()

arccosec()

Arc cosecant (also known as "inverse cosecant").

Syntax

y=arccosec(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\mathbb{C}\setminus\{0\}$	$\surd$

Description

This function calculates the principal value of the the arc cosecant of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \text{arccosec}\,(x)$ with $y \in \left[-\frac{\pi}{2},\, \frac{\pi}{2}\right]$

For $x \in \mathbb{C}$: $y = -i\,\ln\left[\sqrt{1-\frac{1}{x^2}} + \frac{i}{x}\right]$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=arccosec(-1)` returns -1.57,

`y=arccosec(3+4*i)` returns 0.119-j0.16.

See also

cosec(), arcsec()

arccot()

Arc cotangent.

Syntax

y=arccot(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty, +\infty[$	$\surd$

Description

This function calculates the principal value of the arc cotangent of a real or complex number or vector.

For $x \in \mathbb{R}$: y =arccot(x) with $y \in [0, \pi]$

For $x \in \mathbb{C}$: $y = \frac{i}{2} \ln\left(\frac{x-i}{x+i}\right)$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=arccot(-1)` returns 2.36,

`y=arccot(3+4*i)` returns 0.122-j0.159.

See also

cot(), tan(), arccos(), arcsin(), arctan()

arcsec()

Arc secant (also known as "inverse secant").

Syntax

y=arcsec(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\mathbb{C}\setminus\{0\}$	$\surd$

Description

This function calculates the principal value of the arc secant of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \operatorname{arcsec}(x)$ with $y \in [0,\, \pi]$

For $x \in \mathbb{C}$: $y = \frac{\pi}{2} + i\, \ln\left[\sqrt{1 - \frac{1}{x^2}} + \frac{i}{x}\right]$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=arcsec(-1)` returns 3.14,

`y=arcsec(3+4*i)` returns 1.45+j0.16.

See also

sec(), arccosec()

arcsin()

Arc sine (also known as "inverse sine").

Syntax

y=arcsin(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$[-1, +1]$	$\surd$

Description

This function calculates the principal value of the arc sine of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \arcsin(x)$ with $y \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$

For $x \in \mathbb{C}$: $y = -i \ln\left[i\,x + \sqrt{1 - x^2}\right]$

For x being a vector the two equations above are applied to the components of x.

Examples

`y=arcsin(-1)` returns -1.57,

`y=arcsin(3+4*i)` returns 0.634+j2.31.

See also

sin(), arccos(), arctan(), arccot()

arctan()

Arc tangent (also known as "inverse tangent").

Syntax

z=arctan(x)

z=arctan(y,x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
y	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	

Description

For the first syntax (z=arctan(x)), this function calculates the principal value of the arc tangent of a real or complex number or vector.

For $x \in \mathbb{R}$: $y = \arctan(x)$ with $y \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$

For $x \in \mathbb{C}$: $y = -\frac{1}{2}\, i\, \ln\left[\frac{2\,i}{x+i} - 1\right]$

For x being a vector the two equations above are applied to the components of x.

If the second syntax (z=arctan(y, x)) finds application, the expression

$z = \pm \arctan(y/x)$

(with the arctan() function defined above) is evaluated. The sign of z is determined by

$$\text{sign}(z)=\begin{cases} + & for\ Re\{x\} > 0 \\ - & for\ Re\{x\} > 0 \end{cases}.$$

Note that for the second syntax the case $x = y = 0$ is not defined.

Examples

`z=arctan(-1)` returns -0.785,

`z=arctan(3+4*i)` returns 1.45+j0.159,

`z=arctan(1,1)` returns 0.785.

See also

tan(), arccos(), arcsin(), arccot()

Hyperbolic Functions

cosh()

Hyperbolic cosine.

Syntax

y=cosh(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty, +\infty[$	$\surd$

Description

This function calculates the hyperbolic cosine of a real or complex number or vector.

$y = \frac{1}{2}\left(e^x + e^{-x}\right)$

For x being a vector the equation above is applied to the components of x.

Examples

`y=cosh(-1)` returns 1.54,

`y=cosh(3+4*i)` returns -6.58-j7.58.

See also

exp(), sinh(), tanh(), cos()

cosech()

Hyperbolic cosecant.

Syntax

y=cosech(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus\{0\}$	$\surd$

Description

This function calculates the hyperbolic cosecant of a real or complex number or vector.

$$y = \frac{1}{\sinh\, x}$$

For x being a vector the equation above is applied to the components of x.

Examples

`y=cosech(-1)` returns -0.851,

`y=cosech(3+4*i)` returns -0.0649+j0.0755.

See also

exp(), sinh(), sech(), cosec()

coth()

Hyperbolic cotangent.

Syntax

y=coth(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[\setminus\{0\}$	$\surd$

Description

This function calculates the hyperbolic cotangent of a real or complex number or vector.

$$y = \frac{1}{\tanh\, x} = \frac{e^x + e^{-x}}{e^x - e^{-x}}$$

For x being a vector the equation above is applied to the components of x.

Examples

`y=coth(-1)` returns -1.31,

`y=coth(3+4*i)` returns 0.999-j0.0049.

See also

exp(), cosh(), sinh(), tanh(), tan()

sech()

Hyperbolic secant.

Syntax

y=sech(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the hyperbolic secant of a real or complex number or vector.

$$y = \frac{1}{\cosh\, x}$$

For x being a vector the equation above is applied to the components of x.

Examples

`y=sech(-1)` returns 0.648,

`y=sech(3+4*i)` returns -0.0653+j0.0752.

See also

exp(), cosh(), cosech(), sec()

sinh()

Hyperbolic sine.

Syntax

y=sinh(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the hyperbolic sine of a real or complex number or vector.

$y = \frac{1}{2}\left(e^{x} - e^{-x}\right)$

For x being a vector the equation above is applied to the components of x.

Examples

`y=sinh(-1)` returns -1.18,

`y=sinh(3+4*i)` returns -6.55-j7.62.

See also

exp(), cosh(), tanh(), sin()

tanh()

Hyperbolic tangent.

Syntax

y=tanh(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the hyperbolic tangent of a real or complex number or vector.

$$y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

For x being a vector the equation above is applied to the components of x.

Examples

`y=tanh(-1)` returns -0.762,

`y=tanh(3+4*i)` returns 1+j0.00491.

See also

exp(), cosh(), sinh(), coth(), tan()

Inverse Hyperbolic Functions

arcosh()

Hyperbolic area cosine.

Syntax

y=arcosh(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$[1, +\infty[$	$\surd$

Description

This function calculates the hyperbolic area cosine of a real or complex number or vector, which is the inverse function to the "cosh" function.

$y = \operatorname{arcosh} x = \ln\left(x + \sqrt{x^2 - 1}\right)$

For x being a vector the equation above is applied to the components of x.

Examples

`y=arcosh(1)` returns 0,

`y=arcosh(3+4*i)` returns 2.31+j0.937.

See also

arsinh(), artanh(), cosh(), arccos(), ln(), sqrt()

arcosech()

Hyperbolic area cosecant.

Syntax

y=arcosech(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\mathbb{C}\setminus\{0\}$	$\surd$

Description

This function calculates the hyperbolic area cosecant of a real or complex number or vector, which is the inverse function to the "cosech" function.

For $x \in \mathbb{C}\setminus\{0\}$: $y = \ln\left(\sqrt{1+\frac{1}{x^2}}+\frac{1}{x}\right)$

For x being a vector the equation above is applied to the components of x.

Examples

`y=arcosech(1)` returns 0.881,

`y=arcosech(i)` returns -i1.57.

See also

cosech(), arsech(), ln(), sqrt()

arcoth()

Hyperbolic area cotangent.

Syntax

y=arcoth(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,-1[\cup]+1,+\infty[$	$\surd$

Description

This function calculates the hyperbolic area cotangent of a real or complex number or vector, which is the inverse function to the "cotanh" function.

$$y = \operatorname{arcoth} x = \frac{1}{2} \ln\left(\frac{x+1}{x-1}\right)$$

For x being a vector the equation above is applied to the components of x.

Examples

`y=arcoth(2)` returns 0.549,

`y=arcoth(3+4*i)` returns 0.118-j0.161.

See also

arsinh(), arcosh(), tanh(), arctan(), ln(), sqrt()

arsech()

Hyperbolic area secant.

Syntax

y=arsech(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\mathbb{C}\setminus\{0\}$	$\surd$

Description

This function calculates the hyperbolic area secant of a real or complex number or vector, which is the inverse function to the "sech" function.

For $x \in \mathbb{C}\setminus\{0\}$: $y = \ln\left(\sqrt{\frac{1}{x}-1}\sqrt{\frac{1}{x}+1}+\frac{1}{x}\right)$

For x being a vector the equation above is applied to the components of x.

Examples

`y=arsech(1)` returns 0,

`y=arsech(3+4*i)` returns 0.16-j1.45.

See also

sech(), arcosech(), ln(), sqrt()

arsinh()

Hyperbolic area sine.

Syntax

y=arsinh(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the hyperbolic area sine of a real or complex number or vector, which is the inverse function to the "sinh" function.

$y = \operatorname{arsinh} x = \ln\left(x + \sqrt{x^2+1}\right)$

For x being a vector the equation above is applied to the components of x.

Examples

`y=arsinh(1)` returns 0.881,

`y=arsinh(3+4*i)` returns 2.3+j0.918.

See also

arcosh(), artanh(), sinh(), arcsin(), ln(), sqrt()

artanh()

Hyperbolic area tangent.

Syntax

y=artanh(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-1,+1[$	$\surd$

Description

This function calculates the hyperbolic area tangent of a real or complex number or vector, which is the inverse function to the "tanh" function.

$$y = \operatorname{artanh} x = \frac{1}{2} \ln\left(\frac{1+x}{1-x}\right)$$

For x being a vector the equation above is applied to the components of x.

Examples

`y=artanh(0)` returns 0,

`y=artanh(3+4*i)` returns 0.118+j1.41.

See also

arsinh(), arcosh(), tanh(), arctan(), ln(), sqrt()

Rounding

ceil()

Round to the next higher integer.

Syntax

y=ceil(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function rounds a real number x to the next higher integer value.

If x is a complex number both real part and imaginary part are rounded. For x being a vector the operation above is applied to the components of x.

Examples

`y=ceil(-3.5)` returns -3,

`y=ceil(3.2+4.7*i)` returns 4+j5.

See also

floor(), fix(), round()

fix()

Truncate decimal places from real number.

Syntax

y=fix(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function truncates the decimal places from a real number x and returns an integer.

If x is a complex number both real part and imaginary part are rounded. For x being a vector the operation above is applied to the components of x.

Examples

`y=fix(-3.5)` returns -3,

`y=fix(3.2+4.7*i)` returns 3+j4.

See also

ceil(), floor(), round()

floor()

Round to the next lower integer.

Syntax

y=floor(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function rounds a real number x to the next lower integer value.

If x is a complex number both real part and imaginary part are rounded. For x being a vector the operation above is applied to the components of x.

Examples

`y=floor(-3.5)` returns -4,

`y=floor(3.2+4.7*i)` returns 3+j4.

See also

ceil(), fix(), round()

round()

Round to nearest integer.

Syntax

y=round(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function rounds a real number x to its nearest integer value.

If x is a complex number both real part and imaginary part are rounded. For x being a vector the operation above is applied to the components of x.

Examples

`y=round(-3.5)` returns -4,

`y=round(3.2+4.7*i)` returns 3+j5.

See also

ceil(), floor(), fix()

besseli0()

Modified Bessel function of order zero.

Syntax

i0=besseli0(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function evaluates the modified Bessel function of order zero of a real or complex number or vector.

$$i0\left(x\right)=J_0\left(i\,x\right)=\sum_{k=0}^{\infty}\frac{\left(\frac{x}{2}\right)^{2k}}{k!\,\Gamma\left(k+1\right)},$$

where $J_0\left(x\right)$is the Bessel function of order zero and $\Gamma\left(x\right)$denotes the gamma function.

For x being a vector the equation above is applied to the components of x.

Example

`y=besseli0(1)` returns 1.266.

See also

besselj(), bessely()

besselj()

Bessel function of n-th order.

Syntax

jn=besselj(n,x)

Arguments

Name	Type	Def. Range	Required
n	$\mathbb{N}$	$[0,+\infty[$	$\surd$
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function evaluates the Bessel function of n-th order of a real or complex number or vector.

$$J_n\left(x\right)=\sum_{k=0}^{\infty}\frac{\left(-1\right)^k\left(\frac{x}{2}\right)^{n+2k}}{k!\,\Gamma\left(n+k+1\right)},$$

where $\Gamma\left(x\right)$denotes the gamma function.

For x being a vector the equation above is applied to the components of x.

Example

`y=besselj(1,1)` returns 0,44.

See also

besseli0(), bessely()

bessely()

Bessel function of second kind and n-th order.

Syntax

yn=bessely(n,x)

Arguments

Name	Type	Def. Range	Required
n	$\mathbb{N}$	$[0,+\infty[$	$\surd$
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function evaluates the Bessel function of second kind and n-th order of a real or complex number or vector.

$$Y_n(x) = \lim_{m \to n} \frac{J_m(x)\,\cos m\pi - J_{-m}(x)}{\sin m\pi},$$

where $J_m(x)$denotes the Bessel function of first kind and n-th order.

For x being a vector the equation above is applied to the components of x.

Example

`y=bessely(1,1)` returns -0.781.

See also

besseli0(), besselj()

erf()

Error function.

Syntax

y=erf(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function evaluates the error function of a real or complex number or vector. For $x \in \mathbb{R}$,

$$y = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

If x is a complex number both real part and imaginary part are subjected to the equation above. For x being a vector the equation is applied to the components of x.

Example

`y=erf(0.5)` returns 0.520.

See also

erfc(), erfinv(), erfcinv(), exp()

erfc()

Complementary error function.

Syntax

y=erfc(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function evaluates the complementary error function of a real or complex number or vector. For $x \in \mathbb{R}$,

$$y = 1 - \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

If x is a complex number both real part and imaginary part are subjected to the equation above. For x being a vector the equation is applied to the components of x.

Example

`y=erfc(0.5)` returns 0.480.

See also

erf(), erfinv(), erfcinv(), exp()

erfinv()

Inverse error function.

Syntax

y=erfinv(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-1,+1[$	$\surd$

Description

This function evaluates the inverse of the error function of a real or complex number or vector. For $-1 < x < 1$,

$y = \mathrm{erf}^{-1}(x)$

If x is a complex number both real part and imaginary part are subjected to the equation above. For x being a vector the equation is applied to the components of x.

Example

`y=erfinv(0.8)` returns 0.906.

See also

erf(), erfc(), erfcinv(), exp()

erfcinv()

Inverse complementary error function.

Syntax

y=erfcinv(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]0,+2[$	$\surd$

Description

This function evaluates the inverse of the complementary error function of a real or complex number or vector. For $0 < x < 2$,

$y = \mathrm{erfc}^{-1}(x)$

If x is a complex number both real part and imaginary part are subjected to the equation above. For x being a vector the equation is applied to the components of x.

Example

`y=erfcinv(0.5)` returns 0.477.

See also

erf(), erfc(), erfinv(), exp()

sinc()

Sinc function.

Syntax

y=sinc(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function evaluates the sinc function of a real or complex number or vector.

$$y = \begin{cases} \dfrac{\sin x}{x} & for\ x \neq 0 \\ 1 & for\ x = 0 \end{cases}$$

For x being a vector the equation above is applied to the components of x.

Examples

`y=sinc(-3)` returns 0.047,

`y=sinc(3+4*i)` returns -3.86-j3.86.

See also

sin()

step()

Step function.

Syntax

y=step(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function calculates the step function of a real or complex number or vector. For $x \in \mathbb{R}$,

$$y = \begin{cases} 0 & for\, x < 0 \\ 0.5 & for\, x = 0 \\ 1 & for\, x > 0 \end{cases}$$

If x is a complex number both real part and imaginary part are subjected to the equation above. For x being a vector the equation is applied to the components of x.

Example

`y=step(0.5)` returns 1.

See also

Data Analysis

Basic Statistics

avg()

Average of vector elements.

Syntax

y=avg(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, Range $xs:xe$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the sum of the elements of a real or complex vector or range.

For $x \in \mathbb{C}^n$: $y = \frac{1}{n}\sum_{i=1}^{n} x_i,\ 1 \le i \le n$ (for vectors) or $xs \le i \le xe$ (for ranges)

For x being a real or complex number, x itself is returned.

Example

`y=avg(linspace(1,3,10))` returns 2.

See also

sum(), max(), min()

cumavg()

Cumulative average of vector elements.

Syntax

y=cumavg(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the cumulative average of the elements of a real or complex vector.

For $x \in \mathbb{C}^n$: $y_k = \frac{1}{k}\sum_{i=1}^{k} x_i$, $1 \leq k \leq n$

For x being a real or complex number, x itself is returned.

Example

`y=cumavg(linspace(1,3,3))` returns 1, 1.5, 2.

See also

cumsum(), cumprod(), avg(), sum(), prod(), max(), min()

max()

Maximum value.

Syntax

y=max(x)

y=max(a,b)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, Range $xs:xe$	$]-\infty,+\infty[$	$\surd$
a	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$	$\surd$
b	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$	$\surd$

Description

For the first syntax (y=max(x)), this function returns the maximum value of a real or complex vector or range.

For $x \in \mathbb{R}^n$: $y = \max(x_i),\ 1 \le i \le n$ (for vectors) or $xs \le i \le xe$ (for ranges)

For $x \in \mathbb{C}^n$: $y = \max(\pm|x_i|),\ 1 \le i \le n$ (for vectors) or $xs \le i \le xe$ (for ranges),

with sign $\begin{cases} + & for\ |\arg(x_i)| \le \frac{\pi}{2} \\ - & else \end{cases}$

For x being a real or complex number: that is the case $n = 1$.

The second syntax (y=max(a,b)) finds application, if two (generally complex) numbers a and b need to be compared. In principle, the maximum of the absolute values is selected, but it must be considered whether a and b are located in the right or left complex half plane. If the latter is the case, the negative absolute

value of a and b needs to be regarded (for example, which is the case for negative real numbers), otherwise the positive absolute value is taken:

$y = \max(\pm|a|, \pm|b|)$,

with $|a|$ sign $\begin{cases} + & for\ |\arg(a)| \leq \frac{\pi}{2} \\ - & else \end{cases}$ and $|b|$ sign $\begin{cases} + & for\ |\arg(b)| \leq \frac{\pi}{2} \\ - & else \end{cases}$

Example

`y=max(linspace(1,3,10))` returns 3.

`y=max(1,3)` returns 3.

`y=max(1,1+i)` returns 1+j1.

`y=max(1,-1+i)` returns 1.

See also

min(), abs()

min()

Minimum value.

Syntax

y=min(x)

y=min(a,b)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$, Range $xs:xe$	$]-\infty,+\infty[$	$\surd$
a	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$	$\surd$
b	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$	$\surd$

Description

For the first syntax (y=min(x)), this function returns the minimum value of a real or complex vector or range.

For $x \in \mathbb{R}^n$: $y = \min(x_i),\ 1 \le i \le n$ (for vectors) or $xs \le i \le xe$ (for ranges)

For $x \in \mathbb{C}^n$: $y = \min(\pm|x_i|),\ 1 \le i \le n$ (for vectors) or $xs \le i \le xe$ (for ranges),

with sign $\begin{cases} + & for\ |\arg(x_i)| \le \frac{\pi}{2} \\ - & else \end{cases}$

For x being a real or complex number: that is the case $n = 1$.

The second syntax (y=min(a,b)) finds application, if two (generally complex) numbers a and b need to be compared. In principle, the maximum of the absolute values is selected, but it must be considered whether a and b are located in the right or left complex half plane. If the latter is the case, the negative absolute

value of a and b needs to be regarded (for example, which is the case for negative real numbers), otherwise the positive absolute value is taken:

$y = \min(\pm|a|, \pm|b|)$,

with $|a|$ sign $\begin{cases} + & for\ |\arg(a)| \leq \frac{\pi}{2} \\ - & else \end{cases}$ and $|b|$ sign $\begin{cases} + & for\ |\arg(b)| \leq \frac{\pi}{2} \\ - & else \end{cases}$

Example

`y=min(linspace(1,3,10))` returns 1.

`y=min(1,3)` returns 1.

`y=min(1,1+i)` returns 1.

`y=min(1,-1+i)` returns -1+j1.

See also

max(), abs()

rms()

Root Mean Square of vector elements.

Syntax

y=rms(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the rms (root mean square) value of the elements of a real or complex vector. By application of the trapezoidal integration rule,

$$\text{for } x \in \mathbb{C}^n\text{: } y = \sqrt{\frac{1}{n}\sum_{i=1}^{n} a_i\, x_i\, x_i^*},\ 1 \le i \le n,\ a_i = \begin{cases} 1 & \text{for } 2 \le i \le n-1 \\ \frac{1}{2} & \text{for } i = 1 \text{ or } i = n \end{cases}$$

For x being a real or complex number, $|x|$ itself is returned.

Example

`y=rms(linspace(1,2,8))` returns 1.43.

See also

variance(), stddev(), avg()

runavg()

Running average of vector elements.

Syntax

y=runavg(x,m)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
m	$\mathbb{N}$	$[1,+\infty[$	$\surd$

Description

This function returns the running average over m elements of a real or complex vector.

For $x \in \mathbb{C}^n$: $y_k = \frac{1}{m} \sum_{i=k}^{k+m-1} x_i$, $1 \leq k \leq n$

For x being a real or complex number, x itself is returned.

Example

`y=runavg(linspace(1,3,6),2)` returns 1.2, 1.6, 2, 2.4, 2.8.

See also

cumavg(), cumsum(), avg(), sum()

stddev()

Standard deviation of vector elements.

Syntax

y=stddev(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the stddev of the elements of a real or complex vector x.

For $x \in \mathbb{C}^n$: $y = \sqrt{\text{variance}(x)}$

For x being a real or complex number, 0 is returned.

Example

`y=stddev(linspace(1,3,10))` returns 0.673.

See also

stddev(), avg(), max(), min()

variance()

Variance of vector elements.

Syntax

y=variance(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the variance of the elements of a real or complex vector.

For $x \in \mathbb{C}^n$: $y = \dfrac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$, where $\overline{x}$ denotes mean (average) value of x.

For x being a real or complex number, 0 is returned.

Example

`y=variance(linspace(1,3,10))` returns 0.453.

See also

stddev(), avg(), max(), min()

random()

Random number between 0.0 and 1.0.

Syntax

y=random()

Arguments

None.

Description

This function returns a pseudo-random real number between 0.0 (including) and 1.0 (excluding). The starting point of the random number generator can be set by srandom().

Example

```
y=random()
```

See also

srandom()

srandom()

Set seed for a new series of pseudo-random numbers.

Syntax

y=srandom(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$	$]-\infty,+\infty[$	$\surd$

Description

This function establishes x as the seed for a new series of pseudo-random numbers. Please note that only integer values for x are considered, so for example $x = 1.1$ will give the same seed as $x = 1$.

Example

```
y=srandom(100)
```

See also

random()

Basic Operation

cumprod()

Cumulative product of vector elements.

Syntax

y=cumprod(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the cumulative product of the elements of a real or complex vector.

For $x \in \mathbb{C}^n$: $y_k = \prod_{i=1}^{k} x_i$, $1 \leq k \leq n$

For x being a real or complex number, x itself is returned.

Example

`y=cumprod(linspace(1,3,3))` returns 1, 2, 6.

See also

cumsum(), cumavg(), prod(), sum(), avg(), max(), min()

cumsum()

Cumulative sum of vector elements.

Syntax

y=cumsum(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the cumulative sum of the elements of a real or complex vector.

For $x \in \mathbb{C}^n$: $y_k = \sum_{i=1}^{k} x_i$, $1 \leq k \leq n$

For x being a real or complex number, x itself is returned.

Example

`y=cumsum(linspace(1,3,3))` returns 1, 3, 6.

See also

cumprod(), cumavg(), sum(), prod(), avg(), max(), min()

interpolate()

Equidistant spline interpolation of data vector.

Syntax

z=interpolate(y,t,m)

z=interpolate(y,t)

Arguments

Name	Type	Def. Range	Required	Default
y	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$	
t	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$	
m	$\mathbb{N}$	$[3,+\infty[$		64

Description

This function uses spline interpolation to interpolate between the points of a vector *y(t)*. If the number of samples n is not specified, a default value of $n = 64$ is assumed.

Example

```
z=interpolate(linspace(0,2,3)*linspace(0,2,3),linspace(0,2,3))
```

returns a smooth parabolic curve:

Use the Cartesian diagram to display it.

See also

sum(), prod()

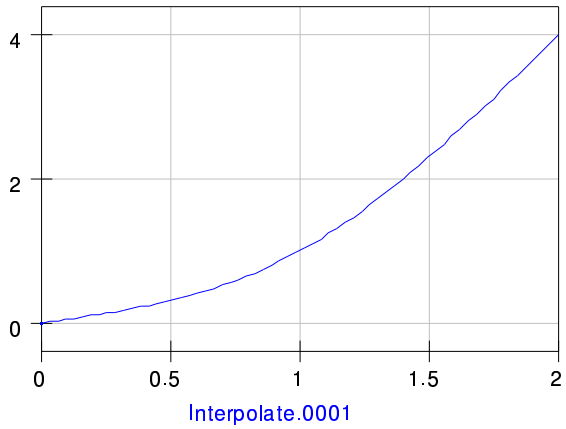

Figure 4: Interpolated curve

prod()

Product of vector elements.

Syntax

y=prod(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the product of the elements of a real or complex vector.

For $x \in \mathbb{C}^n$: $y = \prod_{i=1}^{n} x_i$

For x being a real or complex number, x itself is returned.

Example

`y=prod(linspace(1,3,10))` returns 583.

See also

sum(), avg(), max(), min()

sum()

Sum of vector elements.

Syntax

y=sum(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the sum of the elements of a real or complex vector.

For $x \in \mathbb{C}^n$: $y = \sum_{i=1}^{n} x_i$

For x being a real or complex number, x itself is returned.

Example

`y=sum(linspace(1,3,10))` returns 20.

See also

prod(), avg(), max(), min()

xvalue()

Returns x-value which is associated with the y-value nearest to a specified y-value in a given vector.

Syntax

x=xvalue(f,yval)

Arguments

Name	Type	Def. Range	Required
f	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
yval	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the x-value which is associated with the y-value nearest to *yval* in the given vector f; therefore the vector f must have a single data dependency.

Example

`x=xvalue(f,1)`.

See also

yvalue(), interpolate()

yvalue()

Returns y-value of a given vector which is located nearest to the specified x-value.

Syntax

y=yvalue(f,xval)

Arguments

Name	Type	Def. Range	Required
f	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
xval	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the y-value of the given vector f which is located nearest to the x-value *xval*; therefore the vector f must have a single data dependency.

Example

`y=yvalue(f,1)`.

See also

xvalue(), interpolate()

Differentiation and Integration

ddx()

Differentiate mathematical expression with respect to a given variable.

Syntax

y=ddx(f(x),x)

Arguments

Name	Type	Def. Range	Required	Default
f(x)			√	
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^m$, $\mathbb{C}^m$	$]-\infty,+\infty[$	√	

Description

This function executes a symbolic differentiation on a function *f(x)* with respect to a variable x. The result is evaluated at the contents x_0 of x.

$$y = \left.\frac{df}{dx}\right|_{x_0}$$

If x is a vector, the differential quotient is evaluated for all components of x, giving a result vector y.

Example

Create a vector x by setting `x=linspace(0,2,3)`, thus $x = [0, 1, 2]^T$. Entering

`y=ddx(sin(x),x` returns 1, 0.54, -0.416.

Why? $\frac{df}{dx} = \frac{d\sin(x)}{dx} = \cos(x)$, and $\cos(x)$ evaluated at $x = [0, 1, 2]^T$ gives the result above.

See also

diff()

diff()

Differentiate vector with respect to another vector.

Syntax

z=diff(y,x,n)

Arguments

Name	Type	Def. Range	Required	Default
y	$\mathbb{R}^k$, $\mathbb{C}^k$	$]-\infty,+\infty[$	$\surd$	
x	$\mathbb{R}^m$, $\mathbb{C}^m$	$]-\infty,+\infty[$	$\surd$	
n	$\mathbb{N}$			1

Description

This function numerically differentiates a vector y with respect to a vector x. If the optional integer parameter n is given, the n-th derivative is calculated. Differentiation is executed for N=min*(k,m)* elements. For n=1,

$$\frac{\Delta y_i}{\Delta x_i} = \begin{cases} \frac{1}{2}\left(\frac{y_i - y_{i-1}}{x_i - x_{i-1}} + \frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right) & \text{for } N-1 > i > 0 \\ \frac{y_{i+1} - y_i}{x_{i+1} - x_i} & \text{for } i = 0 \\ \frac{y_i - y_{i-1}}{x_i - x_{i-1}} & \text{for } i = N-1 \end{cases}$$

If *n>1*, the result of the differentiation above is assigned to y and the aforementioned differentiation step is repeated until the number of those steps is equal to n.

Example

`z=diff(linspace(1,3,3),linspace(2,3,3))` returns 2, 2, 2.

See also

integrate(), sum(), max(), min()

integrate()

Integrate vector.

Syntax

z=integrate(y,h)

Arguments

Name	Type	Def. Range	Required
y	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
h	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$	$\surd$

Description

This function numerically integrates a vector x with respect to a differential h. The integration method is according to the trapezoidal rule:

$$\int f\left(t\right)dt \approx h\left(\frac{y_0}{2}+y_1+y_2+\ldots+y_{n-1}+\frac{y_n}{2}\right)$$

Example

Calculate an approximation of the integral $\int_1^3 t\,dt$ using 101 points:

`z=integrate(linspace(1,3,101))` returns 4.

See also

diff(), sum(), max(), min()

Signal Processing

dft()

Discrete Fourier Transform.

Syntax

y=dft(v)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function computes the Discrete Fourier Transform (DFT) of a vector v. The advantage of this function compared to fft() is that the number n of components of v is arbitrary, while for the latter n must be a power of 2. The drawbacks are that dft() is slower and less accurate than fft().

Example

This calculates the spectrum y of a DC signal:

`y=dft(linspace(1,1,7))` returns

y
1
-1.59e-17+j1.59e-17
⋮
2.22e-16-j1.11e-16

Please note that in this example 7 points are used for the time vector v. Since 7 is not a power of 2, the same expression used together with the fft() function would lead to wrong results. Note also the rounding errors where "0" would be the correct value.

See also

idft(), fft(), ifft(), Freq2Time(), Time2Freq()

fft()

Fast Fourier Transform.

Syntax

y=fft(v)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function computes the Fast Fourier Transform (FFT) of a vector v. The number n of components of v must be a power of 2.

Example

This calculates the spectrum y of a DC signal:

`y=fft(linspace(1,1,8))` returns

y
1
0
⋮
0

See also

ifft(), dft(), idft(), Freq2Time(), Time2Freq(), fftshift()

idft()

Inverse Discrete Fourier Transform.

Syntax

y=idft(v)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function computes the Inverse Discrete Fourier Transform (IDFT) of a vector v. The advantage of this function compared to ifft() is that the number n of components of v is arbitrary, while for the latter n must be a power of 2. The drawbacks are that idft() is slower and less accurate than ifft().

Example

This calculates the time function y belonging to a white spectrum:

`y=idft(linspace(1,1,7))` returns

y
7
-1.11e-16-j1.11e-16
⋮
1.55e-15+j7.77e-16

Please note that in this example 7 points are used for the spectrum vector v. Since 7 is not a power of 2, the same expression used together with the ifft() function would lead to wrong results. Note also the rounding errors where "0" would be the correct value.

See also

dft(), ifft(), fft(), Freq2Time(), Time2Freq(), fftshift()

ifft()

Inverse Fast Fourier Transform.

Syntax

y=ifft(v)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function computes the Inverse Fast Fourier Transform (IFFT) of a vector v. The number n of components of v must be a power of 2.

Example

This calculates the time function y belonging to a white spectrum:

`y=ifft(linspace(1,1,8))` returns

y
8
0
⋮
0

See also

fft(), dft(), idft(), Freq2Time(), Time2Freq(), fftshift()

fftshift()

Move the frequency 0 to the center of the FFT vector.

Syntax

y=fftshift(v)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty, +\infty[$	$\surd$

Description

This function shuffles the FFT values of vector v in order to move the frequency 0 to the center of the vector. Below of it the components with negative frequencies are located, above those with positive frequencies. Herewith the "classical" look of a spectrum as gained by a spectrum analyzer is obtained.

Example

Suppose x to be the result of a FFT of 8 elements, e.g.

x
1
2
⋮
8

The result of the FFT is sorted in such a way that the component with frequency zero is the first element (1) of the vector. The components with positive frequency follow (2,3,4). After that, the components with negative frequency (5,6,7,8) are arranged, starting from the most negative value. This pattern can be exemplarily generated in Qucs by writing `x=linspace(1,8,8)`. Then

`y=fftshift(x)` returns

y
5
6
7
8
1
2
3
4

As you can see, the component with frequency 0 (element 1) is moved to the middle of the spectrum vector. Beneath of it the components with negative frequencies appear (5,6,7,8), above those with positive frequencies (2,3,4).

See also

fft(), ifft(), dft(), idft()

Time2Freq()

Interpreted Discrete Fourier Transform.

Syntax

y=Time2Freq(v,t)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
t	$\mathbb{R}^k$, $\mathbb{C}^k$	$]-\infty,+\infty[$	$\surd$

Description

This function computes the Discrete Fourier Transform (DFT) of a vector v with respect to a time vector t.

Example

This calculates the spectrum y(f) of a DC signal:

`y=Time2Freq(linspace(1,1,7),linspace(0,1,2))` returns

Frequency	y
0	1
0.167	-1.59e-17+j1.59e-17
⋮	⋮
1	2.22e-16-j1.11e-16

Please note that in this example 7 points are used for the time vector v. Note also the rounding errors at t>0, where "0" would be the correct value.

See also

idft(), fft(), ifft(), Freq2Time()

Freq2Time()

Interpreted Inverse Discrete Fourier Transform.

Syntax

y=Freq2Time(v,f)

Arguments

Name	Type	Def. Range	Required
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$
f	$\mathbb{R}^k$, $\mathbb{C}^k$	$]-\infty,+\infty[$	$\surd$

Description

This function computes the Inverse Discrete Fourier Transform (IDFT) of a vector v with respect to a frequency vector *f*.

Example

This calculates the time function y(t) belonging to a white spectrum:

`y=Freq2Time(linspace(1,1,7),linspace(0,1,2))` returns

Frequency	y
0	7
0.167	-1.11e-16-j1.11e-16
⋮	⋮
1	1.55e-15+j7.77e-16

Please note that in this example 7 points are used for the spectrum vector v. Note also the rounding errors at t>0, where "0" would be the correct value.

See also

dft(), ifft(), fft(), Time2Freq()

kbd()

Kaiser-Bessel derived window.

Syntax

y=kbd(a,n)

y=kbd(a)

Arguments

Name	Type	Def. Range	Required	Default
a	$\mathbb{R}$	$]-\infty,+\infty[$	$\surd$	
n	$\mathbb{N}$	$[1,+\infty[$		64

Description

This function generates a Kaiser-Bessel window according to

$$y_k = \sqrt{\frac{\sum_{i=0}^{k} I_0\left(\pi\, a \sqrt{1-\left(\frac{4\,i}{n}-1\right)}\right)}{\sum_{i=0}^{\frac{n}{2}} I_0\left(\pi\, a \sqrt{1-\left(\frac{4\,i}{n}-1\right)}\right)}},$$

$y_{n-k-1} = y_k$

for $0 \leq k < \frac{n}{2}$

If the parameter n is not specified, n=64 is assumed.

Example

`y=kbd(0.1,4)` returns .

See also

dft(), ifft(), fft()

Electronics Functions

Unit Conversion

dB()

dB value.

Syntax

y=dB(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the dB value of a real or complex number or vector.

$y = 20 \log |x|$

For x being a vector the equation above is applied to the components of x.

Example

`y=db(10)` returns 20.

See also

log10()

dbm()

Convert voltage to power in dBm.

Syntax

y=dBm(u,Z0)

y=dBm(u)

Arguments

Name	Type	Def. Range	Required	Default
u	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$		50

Description

This function returns the corresponding dBm power of a real or complex voltage or vector *u*. The impedance *Z0* referred to is either specified or 50Ω.

$$y = 10\ \log \frac{|u|^2}{Z_0\, 0.001W}$$

For u being a vector the equation above is applied to the components of u.

Please note that u is considered as a rms value, not as an amplitude.

Example

`y=dbm(1)` returns 13.

See also

dbm2w(), w2dbm(), log10()

dbm2w()

Convert power in dBm to power in Watts.

Syntax

y=dBm2w(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function converts the real or complex power or power vector, given in dBm, to the corresponding power in Watts.

$y = 0.001\,10^{\frac{x}{10}}$

For x being a vector the equation above is applied to the components of x.

Example

`y=dbm2w(10)` returns 0.01.

See also

dbm(), w2dbm()

w2dbm()

Convert power in Watts to power in dBm.

Syntax

y=w2dBm(x)

Arguments

Name	Type	Def. Range	Required
x	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function converts the real or complex power or power vector, given in Watts, to the corresponding power in dBm.

$$y = 10 \log \frac{x}{0.001W}$$

For x being a vector the equation above is applied to the components of x.

Example

`y=w2dbm(1)` returns 30.

See also

dbm(), dbm2w(), log10()

Reflection Coefficients and VSWR

rtoswr()

Converts reflection coefficient to voltage standing wave ratio (VSWR).

Syntax

s=rtoswr(r)

Arguments

Name	Type	Def. Range	Required
r	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\|r\| \leq 1$	$\surd$

Description

For a real or complex reflection coefficient r, this function calculates the corresponding voltage standing wave ratio (VSWR) s according to

$$s = \frac{1 + |r|}{1 - |r|}$$

VSWR is a real number and if usually given in the notation "s : 1".

For r being a vector the equation above is applied to the components of r.

Examples

`s=rtoswr(0)` returns 1.

`s=rtoswr(0.1+0.2*i)` returns 1.58.

See also

ytor(), ztor(), rtoy(), rtoz()

rtoy()

Converts reflection coefficient to admittance.

Syntax

y=rtoy(r)

y=rtoy(r, Z0)

Arguments

Name	Type	Def. Range	Required	Default
r	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\|r\| \leq 1$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$	$]-\infty, +\infty[$		50

Description

For a real or complex reflection coefficient r, this function calculates the corresponding admittance y according to

$$y = \frac{1}{Z_0}\,\frac{1-r}{1+r}$$

If the reference impedance *Z0* is not provided, the function assumes $Z0 = 50\Omega$.

For r being a vector the equation above is applied to the components of r.

Example

`y=rtoy(0.333)` returns 0.01.

See also

ytor(), ztor(), rtoswr()

rtoz()

Converts reflection coefficient to impedance.

Syntax

z=rtoz(r)

z=rtoz(r, Z0)

Arguments

Name	Type	Def. Range	Required	Default
r	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\|r\| \leq 1$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$	$]-\infty, +\infty[$		50

Description

For a real or complex reflection coefficient r, this function calculates the corresponding impedance Z according to

$$Z = Z_0 \frac{1-r}{1+r}$$

If the reference impedance *Z0* is not provided, the function assumes $Z0 = 50\Omega$.

For r being a vector the equation above is applied to the components of r.

Example

`z=rtoz(0.333)` returns 99.9.

See also

ztor(), ytor(), rtoswr()

ytor()

Converts admittance to reflection coefficient.

Syntax

r=ytor(Y)

r=ytor(Y, Z0)

Arguments

Name	Type	Def. Range	Required	Default
Y	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$		50

Description

For a real or complex admittance y, this function calculates the corresponding reflection coefficient according to

$$r = \frac{1 - Y\,Z_0}{1 + Y\,Z_0}$$

For Y being a vector the equation above is applied to the components of Y.

If the reference impedance *Z0* is not provided, the function assumes $Z0 = 50\Omega$.

Often a dB measure is given for the reflection coefficient, the so called "return loss":

$RL = -20\,\log|r|$ [dB]

Example

`r=ytor(0.01)` returns 0.333.

See also

rtoy(), rtoz(), rtoswr(), log10(), dB()

ztor()

Converts impedance to reflection coefficient.

Syntax

r=ztor(Z)

r=ztor(Z, Z0)

Arguments

Name	Type	Def. Range	Required	Default
Z	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$	$]-\infty,+\infty[$		50

Description

For a real or complex impedance Z, this function calculates the corresponding reflection coefficient according to

$$r = \frac{Z - Z_0}{Z + Z_0}$$

For Z being a vector the equation above is applied to the components of Z.

If the reference impedance *Z0* is not provided, the function assumes $Z0 = 50\Omega$.

Often a dB measure is given for the reflection coefficient, the so called "return loss":

$RL = -20 \log |r|$ [dB]

Example

`r=ztor(100)` returns 0.333.

See also

rtoz(), rtoy(), rtoswr(), log10(), dB()

N-Port Matrix Conversions

stos()

Converts S-parameter matrix to S-parameter matrix with different reference impedance(s).

Syntax

y=stos(S, Zref)

y=stos(S, Zref, Z0)

Arguments

Name	Type	Def. Range	Required	Default
S	$\mathbb{R}^{n\times n}$, $\mathbb{C}^{n\times n}$	$\lvert S_{ij}\rvert \in]-\infty,+\infty[\,,\ 1\le i,j\le n$ $\lvert S_{ii}\rvert \le 1,\ 1\le i\le n$	$\surd$	
Zref	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$		50

Description

This function converts a real or complex scattering parameter matrix S into a scattering matrix Y. S has a reference impedance *Zref*, whereas the created scattering matrix Y has a reference impedance *Z0*.

If the reference impedance *Z0* is not provided, the function assumes $Z0 = 50\Omega$.

Both *Zref* and *Z0* can be real or complex numbers or vectors; in the latter case the function operates on the elements of *Zref* and *Z0*.

Example

Conversion of 50Ω terminated S-parameters to 100Ω terminated S-parameters:

`S2=stos(eye(2)*0.1,50,100)` returns

-0.241	0
0	-0.241

.

See also

twoport(), stoy(), stoz()

stoy()

Converts S-parameter matrix to Y-parameter matrix.

Syntax

Y=stoy(S)

Y=stoy(S, Zref)

Arguments

Name	Type	Def. Range	Required	Default
S	$\mathbb{R}^{n\times n}$, $\mathbb{C}^{n\times n}$	$\lvert S_{ij}\rvert \in \left]-\infty,+\infty\right[,\ 1\le i,j\le n$ $\lvert S_{ii}\rvert \le 1,\ 1\le i\le n$	$\surd$	
Zref	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\left]-\infty,+\infty\right[$		50

Description

This function converts a real or complex scattering parameter matrix S into an admittance matrix Y. S has a reference impedance *Zref*, which is assumed to be $Zref = 50\Omega$ if not provided by the user.

Zref can be real or complex number or vector; in the latter case the function operates on the elements of *Zref*.

Example

`Y=stoy(eye(2)*0.1,100)` returns $\begin{bmatrix} 0.00818 & 0 \\ 0 & 0.00818 \end{bmatrix}$.

See also

twoport(), stos(), stoz(), ytos()

stoz()

Converts S-parameter matrix to Z-parameter matrix.

Syntax

Z=stoz(S)

Z=stoz(S, Zref)

Arguments

Name	Type	Def. Range	Required	Default
S	$\mathbb{R}^{n\times n}$, $\mathbb{C}^{n\times n}$	$\lvert S_{ij}\rvert \in \left]-\infty,+\infty\right[,\ 1\le i,j\le n$ $\lvert S_{ii}\rvert \le 1,\ 1\le i\le n$	$\surd$	
Zref	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$\left]-\infty,+\infty\right[$		50

Description

This function converts a real or complex scattering parameter matrix S into an impedance matrix Z. S has a reference impedance *Zref*, which is assumed to be $Zref = 50\Omega$ if not provided by the user.

Zref can be real or complex number or vector; in the latter case the function operates on the elements of *Zref*.

Example

`Z=stoz(eye(2)*0.1,100)` returns $\begin{bmatrix} 122 & 0 \\ 0 & 122 \end{bmatrix}$.

See also

twoport(), stos(), stoy(), ztos()

twoport()

Converts a two-port matrix from one representation into another.

Syntax

U=twoport(X, from, to)

Arguments

Name	Type	Def. Range	Required
X	$\mathbb{R}^{2\times2}$, $\mathbb{C}^{2\times2}$	$]-\infty,+\infty[$	$\surd$
from	Character	$\{'Y', 'Z', 'H', 'G', 'A', 'S', 'T'\}$	$\surd$
to	Character	$\{'Y', 'Z', 'H', 'G', 'A', 'S', 'T'\}$	$\surd$

Description

This function converts a real or complex two-port matrix X from one representation into another.

Example

Transfer a two-port Y matrix Y1 into a Z matrix:

```
Y1=eye(2)*0.1
```

`Z1=twoport(Y1,'Y','Z')` returns $\begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$.

See also

stos(), ytos(), ztos(), stoz(), stoy(), ytoz(), ztoy()

ytos()

Converts Y-parameter matrix to S-parameter matrix.

Syntax

S=ytos(Y)

S=ytos(Y, Z0)

Arguments

Name	Type	Def. Range	Required	Default
Y	$\mathbb{R}^{n\times n}$, $\mathbb{C}^{n\times n}$	$]-\infty,+\infty[$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$		50

Description

This function converts a real or complex admittance matrix Y into a scattering parameter matrix S. Y has a reference impedance *Z0*, which is assumed to be *Z0* = 50Ω if not provided by the user.

Z0 can be real or complex number or vector; in the latter case the function operates on the elements of *Z0*.

Example

`S=ytos(eye(2)*0.1,100)` returns

-0.818	0
0	-0.818

.

See also

twoport(), stos(), ztos(), stoy()

ytoz()

Converts Y-parameter matrix to Z-parameter matrix.

Syntax

Z=ytoz(Y)

Arguments

Name	Type	Def. Range	Required
Y	$\mathbb{R}^{n\times n}$, $\mathbb{C}^{n\times n}$	$]-\infty,+\infty[$	$\surd$

Description

This function converts a real or complex admittance matrix Y into an impedance matrix Z.

Example

`Z=ytoz(eye(2)*0.1)` returns $\begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$.

See also

twoport(), ztoy()

ztos()

Converts Z-parameter matrix to S-parameter matrix.

Syntax

S=ztos(Z)

S=ztos(Z, Z0)

Arguments

Name	Type	Def. Range	Required	Default
Z	$\mathbb{R}^{n\times n}$, $\mathbb{C}^{n\times n}$	$]-\infty,+\infty[$	$\surd$	
Z0	$\mathbb{R}$, $\mathbb{C}$, $\mathbb{R}^{n}$, $\mathbb{C}^{n}$	$]-\infty,+\infty[$		50

Description

This function converts a real or complex impedance matrix *Z* into a scattering parameter matrix *S*. *Z* has a reference impedance *Z0*, which is assumed to be *Z0* = 50Ω if not provided by the user.

Z0 can be real or complex number or vector; in the latter case the function operates on the elements of *Z0*.

Example

`S=ztos(eye(2)*0.1,100)` returns

-0.998	0
0	-0.998

.

See also

twoport(), twoport(), stos(), ytos(), stoz()

ztoy()

Converts Z-parameter matrix to Y-parameter matrix.

Syntax

Y=ztoy(Z)

Arguments

Name	Type	Def. Range	Required
Z	$\mathbb{R}^{n\times n}$, $\mathbb{C}^{n\times n}$	$]-\infty,+\infty[$	$\surd$

Description

This function converts a real or complex impedance matrix Z into an admittance matrix Y.

Example

`Y=ztoy(eye(2)*0.1)` returns $\begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$.

See also

twoport(), ytoz()

Amplifiers

GaCircle()

Circle(s) with constant available power gain Ga in the source plane.

Syntax

y=GaCircle(X,Ga,v)

y=GaCircle(X,Ga,n)

y=GaCircle(X,Ga)

Arguments

Name	Type	Def. Range	Required	Default
X	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$	$]-\infty,+\infty[$	$\surd$	
v	$\mathbb{R}^n$	$[0,360]^o$		
Ga	$\mathbb{R}$, $\mathbb{R}^m$	$[0,+\infty[$	$\surd$	
n	$\mathbb{N}$	$[2,+\infty[$		64

Description

This function generates the points of the circle of constant available power gain G_A in the complex source plane (r_S) of an amplifier. The amplifier is described by a two-port S-parameter matrix S. Radius r and center c of this circle are calculated as follows:

$$r = \frac{\sqrt{1 - 2\cdot K\cdot g_A\cdot |S_{12}S_{21}| + g_A^2\cdot |S_{12}S_{21}|^2}}{\left|1 + g_A\cdot\left(|S_{11}|^2 - |\Delta|^2\right)\right|} \text{ and } c = \frac{g_A\left(S_{11}^* - S_{22}\,\Delta^*\right)}{1 + g_A\left(|S_{11}|^2 - |\Delta|^2\right)},$$

where $g_A = \dfrac{G_A}{|S_{21}|^2}$ and K Rollet stability factor. Δ denotes determinant of S.

The points of the circle can be specified by the angle vector v, where the angle must be given in degrees. Another possibility is to specify the number n of angular equally distributed points around the circle. If no additional argument to X is given, 64 points are taken. The available power gain can also be specified in a vector *Ga*, leading to the generation of m circles, where m is the size of *Ga*.

Please also refer to "Qucs - Technical Papers", chapter 1.5.

Example

```
v=GaCircle(S)
```

See also

GpCircle(), Rollet()

GpCircle()

Circle(s) with constant operating power gain Gp in the load plane.

Syntax

y=GpCircle(X,Gp,v)

y=GpCircle(X,Gp,n)

y=GpCircle(X,Gp)

Arguments

Name	Type	Def. Range	Required	Default
X	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$	$]-\infty,+\infty[$	$\surd$	
v	$\mathbb{R}^n$	$[0,360]^o$		
Gp	$\mathbb{R}$, $\mathbb{R}^m$	$[0,+\infty[$	$\surd$	
n	$\mathbb{N}$	$[2,+\infty[$		64

Description

This function generates the points of the circle of constant operating power gain G_P in the complex load plane (r_L) of an amplifier. The amplifier is described by a two-port S-parameter matrix S. Radius r and center c of this circle are calculated as follows:

$$r = \frac{\sqrt{1 - 2\cdot K\cdot g_P\cdot |S_{12}S_{21}| + g_P^2\cdot |S_{12}S_{21}|^2}}{\left|1 + g_P\cdot\left(|S_{22}|^2 - |\Delta|^2\right)\right|} \text{ and } c = \frac{g_A\left(S_{22}^* - S_{11}\,\Delta^*\right)}{1 + g_P\left(|S_{22}|^2 - |\Delta|^2\right)},$$

where $g_A = \dfrac{G_P}{|S_{21}|^2}$ and K Rollet stability factor. Δ denotes determinant of S.

The points of the circle can be specified by the angle vector v, where the angle must be given in degrees. Another possibility is to specify the number n of angular equally distributed points around the circle. If no additional argument to X is

given, 64 points are taken. The available power gain can also be specified in a vector Gp, leading to the generation of m circles, where m is the size of Gp.

Please also refer to “Qucs - Technical Papers”, chapter 1.5.

Example

```
v=GpCircle(S)
```

See also

GaCircle(), Rollet()

Mu()

Mu stability factor of a two-port S-parameter matrix.

Syntax

y=Mu(S)

Arguments

Name	Type	Def. Range	Required
S	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$,$\mathbb{R}^{2\times2}$, $\mathbb{C}^{2\times2}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the Mu stability factor μ of an amplifier being described by a two-port S-parameter matrix S:

$$\mu = \frac{1-|S_{11}|^2}{|S_{22}-S_{11}^*\,\Delta|+|S_{21}\,S_{12}|}$$

Δ denotes determinant of S.

The amplifier is unconditionally stable if $\mu > 1$.

For S being a vector of matrices the equation above is applied to the sub-matrices of S.

Example

```
m=Mu(S)
```

See also

Mu2(), Rollet(), StabCircleS(), StabCircleL()

Mu2()

Mu' stability factor of a two-port S-parameter matrix.

Syntax

y=Mu2(S)

Arguments

Name	Type	Def. Range	Required
S	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$,$\mathbb{R}^{2\times2}$, $\mathbb{C}^{2\times2}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the Mu' stability factor μ' of an amplifier being described by a two-port S-parameter matrix S:

$$\mu' = \frac{1-|S_{22}|^2}{|S_{11}-S_{22}^*\,\Delta|+|S_{21}\,S_{12}|}$$

Δ denotes determinant of S.

The amplifier is unconditionally stable if $\mu' > 1$.

For S being a vector of matrices the equation above is applied to the sub-matrices of S.

Example

```
m=Mu2(S)
```

See also

Mu2(), Rollet(), StabCircleS(), StabCircleL()

NoiseCircle()

Generates circle(s) with constant Noise Figure(s).

Syntax

y=NoiseCircle(Sopt,Fmin,Rn,F,v)

y=NoiseCircle(Sopt,Fmin,Rn,F,n)

y=NoiseCircle(Sopt,Fmin,Rn,F)

Arguments

Name	Type	Def. Range	Required	Default
Sopt	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	√	
Fmin	$\mathbb{R}^n$	$[1,+\infty[$	√	
Rn	$\mathbb{R}^n$, $\mathbb{C}^n$	$[0,+\infty[$	√	
F	$\mathbb{R}$, $\mathbb{R}^n$	$[1,+\infty[$	√	
v	$\mathbb{R}^n$	$[0,360]^o$		
n	$\mathbb{N}$	$[2,+\infty[$		64

Description

This function generates the points of the circle of constant Noise Figure (NF) F in the complex source plane (r_S) of an amplifier. Generally, the amplifier has its minimum NF F_{min}, if the source reflection coefficient $r_S = S_{opt}$(noise matching). Note that this state with optimum source reflection coefficient S_{opt} is different from power matching ! Thus power gain under noise matching is lower than the maximum obtainable gain. The values of S_{opt}, $F_{min}$and the normalised equivalent noise resistance R_n/Z_0can be usually taken from the data sheet of the amplifier.

Radius r and center c of the circle of constant NF are calculated as follows:

$$r = \frac{\sqrt{N^2 + N \cdot \left(1 - |S_{opt}|^2\right)}}{1+N} \text{ and } c = \frac{S_{opt}}{1+N}, \text{ with } N = \frac{F - F_{min}}{4\,R_n} \cdot Z_0 \cdot |1 + S_{opt}|^2$$

.

The points of the circle can be specified by the angle vector v, where the angle must be given in degrees. Another possibility is to specify the number n of angular equally distributed points around the circle. If no additional argument to X is given, 64 points are taken.

Please also refer to "Qucs - Technical Papers", chapter 2.2.

Example

```
v=NoiseCircle(Sopt,Fmin,Rn,F)
```

See also

GaCircle(), GpCircle()

PlotVs()

Returns a data item based upon vector or matrix vector with dependency on a given vector.

Syntax

y=PlotVs(X, v)

Arguments

Name	Type	Def. Range	Required
X	$\mathbb{R}^n$, $\mathbb{C}^n$, $\mathbb{R}^{m\times n\times p}$, $\mathbb{C}^{m\times n\times p}$	$]-\infty,+\infty[$	$\surd$
v	$\mathbb{R}^n$, $\mathbb{C}^n$	$]-\infty,+\infty[$	$\surd$

Description

This function returns a data item based upon a vector or matrix vector X with dependency on a given vector v.

Example

```
PlotVs(Gain,frequency/1E9).
```

See also

Rollet()

Rollet stability factor of a two-port S-parameter matrix.

Syntax

y=Rollet(S)

Arguments

Name	Type	Def. Range	Required
S	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$,$\mathbb{R}^{2\times2}$, $\mathbb{C}^{2\times2}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the Rollet stability factor K of an amplifier being described by a two-port S-parameter matrix S:

$$K = \frac{1 - |S_{11}|^2 - |S_{22}|^2 + |\Delta|^2}{2\,|S_{21}|\,|S_{12}|}$$

Δ denotes determinant of S.

The amplifier is unconditionally stable if $K > 1$ and $|\Delta| < 1$.

Note that a large K may be misleading in case of a multi-stage amplifier, pretending extraordinary stability. This is in conflict with reality where a large gain amplifier usually suffers from instability due to parasitics.

For S being a vector of matrices the equation above is applied to the sub-matrices of S.

Example

```
K=Rollet(S)
```

See also

Mu(), Mu2(), StabCircleS(), StabCircleL()

StabCircleL()

Stability circle in the load plane.

Syntax

y=StabCircleL(X)

y=StabCircleL(X,v)

y=StabCircleL(X,n)

Arguments

Name	Type	Def. Range	Required	Default
X	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$	$]-\infty,+\infty[$	$\surd$	
v	$\mathbb{R}^n$	$[0,360]^o$		
n	$\mathbb{N}$	$[2,+\infty[$		64

Description

This function generates the stability circle points in the complex load reflection coefficient (r_L) plane of an amplifier. The amplifier is described by a two-port S-parameter matrix S. Radius r and center c of this circle are calculated as follows:

$$r = \left|\frac{S_{21}\,S_{12}}{|S_{22}|^2 - |\Delta|^2}\right| \text{ and } c = \frac{S_{22}^* - S_{11}\cdot\Delta^*}{|S_{22}|^2 - |\Delta|^2}$$

Δ denotes determinant of S.

The points of the circle can be specified by the angle vector v, where the angle must be given in degrees. Another possibility is to specify the number n of angular equally distributed points around the circle. If no additional argument to X is given, 64 points are taken.

If the center of the $r_L$plane lies within this circle and $|S_{11}| \leq 1$ then the circuit is stable for all reflection coefficients inside the circle. If the center of the $r_L$plane lies outside the circle and $|S_{11}| \leq 1$ then the circuit is stable for all reflection coefficients outside the circle (please also refer to “Qucs - Technical Papers”, chapter 1.5).

Example

```
v=StabCircleL(S)
```

See also

StabCircleS(), Rollet(), Mu(), Mu2()

StabCircleS()

Stability circle in the source plane.

Syntax

y=StabCircleS(X)

y=StabCircleS(X,v)

y=StabCircleS(X,n)

Arguments

Name	Type	Def. Range	Required	Default
X	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$	$]-\infty,+\infty[$	$\surd$	
v	$\mathbb{R}^n$	$[0,360]^o$		
n	$\mathbb{N}$	$[2,+\infty[$		64

Description

This function generates the stability circle points in the complex source reflection coefficient (r_S) plane of an amplifier. The amplifier is described by a two-port S-parameter matrix S. Radius r and center c of this circle are calculated as follows:

$$r = \left|\frac{S_{21}\,S_{12}}{|S_{11}|^2 - |\Delta|^2}\right| \text{ and } c = \frac{S_{11}^* - S_{22}\cdot\Delta^*}{|S_{11}|^2 - |\Delta|^2}$$

Δ denotes determinant of S.

The points of the circle can be specified by the angle vector v, where the angle must be given in degrees. Another possibility is to specify the number n of angular equally distributed points around the circle. If no additional argument to X is given, 64 points are taken.

If the center of the $r_S$plane lies within this circle and $|S_{22}| \leq 1$ then the circuit is stable for all reflection coefficients inside the circle. If the center of the $r_S$plane lies outside the circle and $|S_{22}| \leq 1$ then the circuit is stable for all reflection coefficients outside the circle (please also refer to “Qucs - Technical Papers”, chapter 1.5).

Example

```
v=StabCircleS(S)
```

See also

StabCircleL(), Rollet(), Mu(), Mu2()

StabFactor()

Stability factor of a two-port S-parameter matrix. Synonym for Rollet()

Syntax

y=StabFactor(S)

See also

Rollet()

StabMeasure()

Stability measure B1 of a two-port S-parameter matrix.

Syntax

y=StabMeasure(S)

Arguments

Name	Type	Def. Range	Required
S	$\mathbb{R}^{2\times2\times p}$, $\mathbb{C}^{2\times2\times p}$,$\mathbb{R}^{2\times2}$, $\mathbb{C}^{2\times2}$	$]-\infty,+\infty[$	$\surd$

Description

This function returns the stability measure *B1* of a two-port S-parameter matrix S:

$$B1 = 1 + |S_{11}|^2 - |S_{22}|^2 - |\Delta|^2$$

Δ denotes determinant of S.

The amplifier is unconditionally stable if $B1 > 0$ and the Rollet factor $K > 1$.

For S being a vector of matrices the equation above is applied to the sub-matrices of S.

Example

```
B1=StabMeasure(S)
```

See also

Rollet(), Mu(), Mu2(), StabCircleS(), StabCircleL()

vt()

Thermal voltage for a given temperature in Kelvin.

Syntax

y=vt(t)

Arguments

Name	Type	Def. Range	Required	Default
t	$\mathbb{R}$	$[0, +\infty[$	$\surd$	

Description

This function returns the corresponding thermal voltage V_t in Volt of a real absolute temperature (vector) T in Kelvin according to

$$V_t = \frac{kT}{e}$$

where k is the Boltzmann constant and e denotes the electrical charge on the electron. For t being a vector the equation above is applied to the components of k.

Please note that t is always larger than or equal to zero.

Example

`y=vt(300)` returns 0.0259.

Index

U

V

W

X

Y

Z