R-Tek Scratchpad Help Index

The Main Toolbar

The Toolbar is a row of buttons at the top of the main window which represent application commands. Clicking one of the buttons is a quick alternative to choosing a command from the menu. Buttons on the toolbar activate and deactivate according to the state of the application.

Button Action Menu Equivalent

	Create a new document	File|New
	Locate and open a file	File|Open
	Save the file in the active window	File|Save
	Close the document of the active view	
	Cut selected objects to Clipboard	Edit|Cut
	Copy selected objects to Clipboard	Edit|Copy
	Paste objects from Clipboard	Edit|Paste
	Undo previous editor action	Edit|Undo
	Search for text	Search|Find
	Repeat last Find or Replace operation	Search|Next
	Print the active window	File|Print
	Preview the printout of the active window	File|Print Preview
	Create a graph at the caret position	
	Display the calculated value	
	Run the Program	
	Show/Hide the Greek alphabet & ops toolbar	
	Display help file contents	Help|Contents

The Greek Alphabet & Ops Toolbar

The Greek Alphabet and Ops Toolbar requires little explanation. Clicking on one of the Greek letters will insert that letter at the current caret position in the document. There are four small buttons left of the large Ops button. They let you select the category of op and then the up and down arrows right of the Ops button lets you select the particular op within the category. Finally pressing the Ops button itself will insert the op into the scratchpad at the current caret position. The Ops button displays specific examples of the result of the operation.

Button	Selection
	Numeric ops
	Matrix specific ops
	Text ops
	Logical ops

Note that ops available without Ctrl or Alt are not present on this toolbar. Use the keyboard. (In fact, I recommend that you not use this toolbar at all. Instead memorize the hot keys. If you try to memorize the hot keys, you will notice that they are largely mnemonic to aid memorization.) Note also, that both left and right grouping characters appear in the Ops button list, so be sure to pick the right one.

General Operators

Operator	Keystrokes	
assignment	:	(type only the colon)
display value	Ctrl+=	
exchange	Alt+=	
grouping	( )	
	{ }	absolute value, complex magnitude, matrix determinant
	[ ]	matrix or vector element subscript grouping pair
	Ctrl+[Ctrl+]	column subvector grouping pair
	Alt+[Alt+]	row subvector grouping pair

Logical Operators

Operator	Keystrokes
not	Ctrl+n
and	Ctrl+a
or	Ctrl+o
equality	=
less than	<
less than or equal	<=
greater than	>
greater than or equal	>=

Note that the not operator may be combined with the above comparison operators but that in those cases you will need to strike the right arrow key between the not keystroke and the comparison operator keystrokes.

Numeric Operators

Operator	Keystrokes
addition	+
subtraction	-
multiplication	*
division	/
power	^
percent	% (simply means divide by 100)
square root	Ctrl+r
factorial	!
absolute value	{ }
complex conjugate	Ctrl+c
complex magnitude	{ }
derivative	Ctrl+d
integral	Ctrl+i
iterated sum	Ctrl+s
iterated product	Ctrl+p
permutations	Alt+p
combinations	Alt+c

Matrix Operators

Operator	Keystrokes	
addition	+	
subtraction	-	
multiplication	*	(multiplication by a scalar is allowed, division is not)
inverse	^-1	(division of matrices is not allowed: inverse powers of a matrix are defined as powers of the inverse)
determinant	{ }	
vector dot product	Alt+*	
vector cross product	Alt+x	
transpose	Ctrl+t	
complex conjugate	Ctrl+c	
augment	Alt+a	
matrix element subscript	[ ]	
vector element subscript	[ ]	(only one subscript required for vector)
row subvector	Alt+[Alt+]	
column subvector	Ctrl+[Ctrl+]	

Numeric Functions

Simple

- least common multiple
- greatest common divisor
- integer divide
- modulus
- ceiling
- floor

Breakdown

- integer part
- fractional part
- numerator
- denominator
- real
- imaginary
- arg

Trigonometric

- sine
- cosine
- tangent
- secant
- cosecant
- cotangent
- arcsine
- arccosine
- arctangent

Hyperbolic

- hyperbolic sine
- hyperbolic cosine
- hyperbolic tangent
- hyperbolic secant
- hyperbolic cosecant
- hyperbolic cotangent
- hyperbolic arcsine
- hyperbolic arccosine
- hyperbolic arctangent

Logarithm

- natural
- base ten

Small Prime

- is a small prime
- small prime factorization
- next small prime
- previous small prime

Random Number

- random
- randomize
- random excluding zero
- random excluding zero and one
- random excluding integers
- random normal deviate

Miscellaneous

- root polish
- Heaviside step function
- Kronecker delta
- completely antisymmetric tensor rank 3

Least Common Multiple

See Also

Syntax

number lcm(number n1, number n2)

Description

returns the smallest integer which can be evenly divided by both n1 and n2

numeric functions
greatest common divisor

Greatest Common Divisor

See Also

Syntax

number gcd(number n1, number n2)

Description

returns the largest integer which can evenly divide both n1 and n2

numeric functions
least common multiple

Integer Divide

See Also

Syntax

number div(number n1, number n2)

Description

returns the integer part of the quotient of n1 divided by n2

numeric functions
modulus

Modulus

See Also

Syntax

number mod(number n1, number n2)

Description

returns the remainder produced when n1 is divided by n2

numeric functions
integer divide

Ceiling

See Also

Syntax

number ceil(number n)

Description

returns the smallest integer greater than or equal to n

numeric functions
floor

Floor

See Also

Syntax

number floor(number n)

Description

returns the largest integer less than or equal to n

numeric functions
ceiling

Integer Part

See Also

Syntax

number ipart(number n)

Description

strips the decimal part of the number n and returns the resulting integer.

numeric functions
fractional part

Fractional Part

See Also

Syntax

number fpart(number n)

Description

strips the integer part of the number n and returns the resulting decimal part

numeric functions
integer part

Numerator

See Also

Syntax

number num(number n)

Description

returns the numerator of the rational number n

numeric functions
denominator

Denominator

See Also

Syntax

number denom(number n)

Description

returns the denominator of the rational number n

numeric functions
numerator

Real

See Also

Syntax

number real(number n)

Description

returns the real part of the complex number n

numeric functions
imaginary

Imaginary

See Also

Syntax

number imag(number n)

Description

returns the imaginary part of the complex number n

numeric functions
real

Arg

See Also

Syntax

number arg(number n)

Description

returns the angle(in radians) of the number n in the complex plane

numeric functions

Sine

See Also

Syntax

number sin(number n)

Description

returns the sine of the angle n (n is in radians)

numeric functions

	cosine	**tangent**
secant	**cosecant**	**cotangent**
arcsine	**arccosine**	**arctangent**

Cosine

See Also

Syntax

number cos(number n)

Description

returns the cosine of the angle n (n is in radians)

numeric functions

sine		**tangent**
secant	**cosecant**	**cotangent**
arcsine	**arccosine**	**arctangent**

Tangent

See Also

Syntax

number tan(number n)

Description

returns the tangent of the angle n (n is in radians)

numeric functions

sine	**cosine**	
secant	**cosecant**	**cotangent**
arcsine	**arccosine**	**arctangent**

Secant

See Also

Syntax

number sec(number n)

Description

returns the secant of the angle n (n is in radians)

numeric functions

sine	**cosine**	**tangent**
	cosecant	**cotangent**
arcsine	**arccosine**	**arctangent**

Cosecant

See Also

Syntax

number csc(number n)

Description

returns the cosecant of the angle n (n is in radians)

numeric functions

sine	**cosine**	**tangent**
secant		**cotangent**
arcsine	**arccosine**	**arctangent**

Cotangent

See Also

Syntax

number cot(number n)

Description

returns the cotangent of the angle n (n is in radians)

numeric functions
sine **cosine** **tangent**
secant **cosecant**
arcsine **arccosine** **arctangent**

ArcSine

See Also

Syntax

number asin(number n)

Description

returns the angle (in radians) whose sine is n

numeric functions
sine **cosine** **tangent**
secant **cosecant** **cotangent**
arccosine **arctangent**

ArcCosine

See Also

Syntax

number acos(number n)

Description

returns the angle (in radians) whose cosine is n

numeric functions

sine	**cosine**	**tangent**
secant	**cosecant**	**cotangent**
arcsine		**arctangent**

ArcTangent

See Also

Syntax

number atan(number n)

Description

returns the angle (in radians) whose tangent is n

numeric functions

sine	**cosine**	**tangent**
secant	**cosecant**	**cotangent**
arcsine	**arccosine**	

Hyperbolic Sine

See Also

Syntax

number sinh(number n)

Description

returns the hyperbolic sine of the angle n

numeric functions

hyperbolic cosine **hyperbolic tangent**
hyperbolic secant **hyperbolic cosecant** **hyperbolic cotangent**
hyperbolic arcsine **hyperbolic arccosine** **hyperbolic arctangent**

Hyperbolic Cosine

See Also

Syntax

number cosh(number n)

Description

returns the hyperbolic cosine of the angle n

numeric functions
hyperbolic sine **hyperbolic tangent**
hyperbolic secant **hyperbolic cosecant** **hyperbolic cotangent**
hyperbolic arcsine **hyperbolic arccosine** **hyperbolic arctangent**

Hyperbolic Tangent

See Also

Syntax

number tanh(number n)

Description

returns the hyperbolic tangent of the angle n

numeric functions
hyperbolic sine **hyperbolic cosine**
hyperbolic secant **hyperbolic cosecant** **hyperbolic cotangent**
hyperbolic arcsine **hyperbolic arccosine** **hyperbolic arctangent**

Hyperbolic Secant

See Also

Syntax

number sech(number n)

Description

returns the hyperbolic secant of the angle n

numeric functions

hyperbolic sine	**hyperbolic cosine**	**hyperbolic tangent**
	hyperbolic cosecant	**hyperbolic cotangent**
hyperbolic arcsine	**hyperbolic arccosine**	**hyperbolic arctangent**

Hyperbolic Cosecant

See Also

Syntax

number csch(number n)

Description

returns the hyperbolic cosecant of the angle n

numeric functions
hyperbolic sine **hyperbolic cosine** **hyperbolic tangent**
hyperbolic secant **hyperbolic cotangent**
hyperbolic arcsine **hyperbolic arccosine** **hyperbolic arctangent**

Hyperbolic Cotangent

See Also

Syntax

number coth(number n)

Description

returns the hyperbolic cotangent of the angle n

numeric functions
hyperbolic sine **hyperbolic cosine** **hyperbolic tangent**
hyperbolic secant **hyperbolic cosecant**
hyperbolic arcsine **hyperbolic arccosine** **hyperbolic arctangent**

Hyperbolic ArcSine

See Also

Syntax

number asinh(number n)

Description

returns the angle whose hyperbolic sin is n

numeric functions
hyperbolic sine **hyperbolic cosine** **hyperbolic tangent**
hyperbolic secant **hyperbolic cosecant** **hyperbolic cotangent**
hyperbolic arccosine **hyperbolic arctangent**

Hyperbolic ArcCosine

See Also

Syntax

number acosh(number n)

Description

returns the angle whose hyperbolic cosine is n

numeric functions
hyperbolic sine **hyperbolic cosine** **hyperbolic tangent**
hyperbolic secant **hyperbolic cosecant** **hyperbolic cotangent**
hyperbolic arcsine **hyperbolic arctangent**

Hyperbolic ArcTangent

See Also

Syntax

number atanh(number n)

Description

returns the angle whose hyperbolic tangent n

numeric functions
hyperbolic sine **hyperbolic cosine** **hyperbolic tangent**
hyperbolic secant **hyperbolic cosecant** **hyperbolic cotangent**
hyperbolic arcsine **hyperbolic arccosine**

Natural Logarithm

See Also

Syntax

number ln(number n)

Description

returns the natural logarithm (base e) of the number n

numeric functions
logarithm base ten

Logarithm

See Also

Syntax

number log(number n)

Description

returns the logarithm (base 10) of the number n

numeric functions
natural logarithm

Is a Small Prime

See Also

Syntax

bool issp(number n)

Description

returns true if n is a prime, false otherwise
(the scratchpad only tests primes less than 100,000,000)

Small Prime Factorization

See Also

Syntax

matrix spfact(number n)

Description

returns the prime factorization of n in a matrix that has two rows and a number of columns equal to the prime factors of n. The elements of row two are the prime factors of n and each element of row one is the number of times that the prime factor in the same column appears in the prime factorization of n.
(the scratchpad only tests primes less than 100,000,000)

numeric functions
is a small prime
previous small prime
next small prime

Previous Small Prime

See Also

Syntax

number prevsp(number n)

Description

returns the largest prime smaller than n
(the scratchpad only tests primes less than 100,000,000)

numeric functions
is a small prime
is a small prime
small prime factorization
next small prime

Next Small Prime

See Also

Syntax

number nxtsp(number n)

Description

returns the smallest prime larger than n
(the scratchpad only tests primes less than 100,000,000)

numeric functions
is a small prime
small prime factorization
previous small prime

Randomize

See Also

Syntax

void randomize(number n)

Description

initializes (seeds) the random number generator. Call this function before using the random number generator if you want your random sequence to be repeatable from one running of your program to the next. The seed should be an integer less than 2 to the 31st power.

numeric functions
random

Random

See Also

Syntax

number rnd(number n)

Description

returns a random number dependent on the number n.

n	returns random
positive integer	integer between zero and n. (n is excluded)
negative integer	integer between -n and n. (n and -n are excluded)
positive real	real between zero and n.
negative real	real between -n and n
positive rational	rational with numerator and denominator treated as separate randoms
negative rational	rational with numerator and denominator treated as separate randoms
imaginary	imaginary dependent on coefficient of square root of minus one
complex	complex with real and imaginary parts treated separately

numeric functions
randomize
random (excluding zero)
random (excluding zero and one)
random (excluding integers)
random normal deviate

Random (excluding zero)

See Also

Syntax

number rndex0(number n)

Description

returns a random number dependent on the number n.
acts just like rnd(n), with the exception that rndex0 will not return zero

numeric functions
randomize
random
random (excluding zero and one)
random (excluding integers)
random normal deviate

Random (excluding zero and one)

See Also

Syntax

number rndex01(number n)

Description

returns a random number dependent on the number n.
acts just like rnd(n), with the exception that rndex01 will not return zero or one

numeric functions
randomize
random
random (excluding zero)
random (excluding integers)
random normal deviate

Random (excluding integers)

See Also

Syntax

number rndexint(number n)

Description

returns a random number dependent on the number n.
acts just like rnd(n), with the exception that rndexint will not return an integer

numeric functions
randomize
random
random (excluding zero)
random (excluding zero and one)
random normal deviate

Random normal deviate

See Also

Syntax

number rnddev(number n)

Description

returns n times a random normal deviate with zero mean and unit variance.

Root polish

See Also

Syntax

number rootpolish(function f, identifier x)

Description

returns a value of x that satisfies the equation f(x) = 0, provided the identifier x has been initialized with a good estimate of a roots value, perhaps determined graphically. The function may not converge or the returned value may be wildly inaccurate if the initial guess is poor.

numeric functions

Heaviside step function

See Also

Syntax

number Φ(number n)

Description

returns one if n is greater than or equal to zero, zero otherwise.

numeric functions

Kronecker delta

See Also

Syntax

number δ(number n1, number n2)

Description

returns one if n1 is equal to n2, zero otherwise.

numeric functions

Completely antisymmetric tensor rank 3

See Also

Syntax

number ε(number n1, number n2, number n3)

Description

n1, n2, and n3 are integers between one and three inclusive. returns one for even permutations, minus one for odd permutations, and zero if any two are the same.

numeric functions

Matrix Functions

Creation

zero matrix | identity matrix
display matrix | diagonal | ones vector
submatrix | subvector
random matrix
random matrix excluding zero
random matrix excluding zero and one
random matrix excluding integers
random matrix excluding singular
random matrix excluding singular and zero
random matrix excluding singular, zero and one
random matrix excluding singular and integers
random data

Matrix Property

rows | columns
maximum | minimum
sum | mean
trace
variance | standard deviation
slope intercept

Sorting

reverse
sort
column sort
row sort

Equation Solving Aids

rank | pivot
minor | adjoint
echelon | echelon transform
particular solution | homogeneous solution
eigenvalues & eigenvectors
singular value decomposition (natural inverse)
singular value decomposition backsubstitution
singular value decomposition variance covariance

Orthonormalization

Gram-Schmidt orthonormalization

Zero Matrix

See Also

Syntax

matrix zmat(number n1, number n2)

Description

returns n1 x n2 matrix automatically initialized so that all elements are zero.
Often used to create a matrix of proper size which is initialized in a MatLoop.

matrix functions
identity matrix
random matrix
display matrix
MatLoop

Identity Matrix

See Also

Syntax

matrix imat(number n)

Description

returns n x n square matrix automatically initialized so that its diagonal elements are one, all other elements zero.

matrix functions
zero matrix
random matrix
display matrix

Submatrix

See Also

Syntax

matrix submat(matrix M, number row1, number col1, number row2, number col2)

Description

returns a submatrix of M. row1, col1 specify the upperleft starting elements, row2, col2 specify the lower right ending element inclusive

matrix functions
subvector

Subvector

See Also

Syntax

matrix subvec(matrix V, number startIndex, number endIndex)

Description

returns a subvector of V, beginning with startIndex and ending with endIndex inclusive.

Random Matrix

See Also

Syntax

matrix rmat(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.

matrix functions
randomize
random
random matrix excluding zero
random matrix excluding zero and one
random matrix excluding integers
random matrix excluding singular
random matrix excluding singular and zero
random matrix excluding singular, zero and one
random matrix excluding singular and integers
random data

Random Matrix (excluding zero)

See Also

Syntax

matrix rmatex0(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.
excludes zero as an element of the matrix on creation

matrix functions
randomize
random
random matrix
random matrix excluding zero and one
random matrix excluding integers
random matrix excluding singular
random matrix excluding singular and zero
random matrix excluding singular, zero and one
random matrix excluding singular and integers
random data

Random Matrix (excluding zero and one)

See Also

Syntax

matrix rmatex01(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.
excludes zero and one as elements of the matrix on creation

matrix functions
randomize
random
random matrix
random matrix excluding zero
random matrix excluding integers
random matrix excluding singular
random matrix excluding singular and zero
random matrix excluding singular, zero and one
random matrix excluding singular and integers
random data

Random Matrix (excluding integers)

See Also

Syntax

matrix rmatexint(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.
excludes integers as elements of the matrix on creation

matrix functions
randomize
random
random matrix
random matrix excluding zero
random matrix excluding zero and one
random matrix excluding singular
random matrix excluding singular and zero
random matrix excluding singular, zero and one
random matrix excluding singular and integers
random data

Random Matrix (excluding singular)

See Also

Syntax

matrix rmatexs(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.
matrix will not be singular if square

matrix functions
randomize
random
random matrix
random matrix excluding zero
random matrix excluding zero and one
random matrix excluding integers
random matrix excluding singular and zero
random matrix excluding singular, zero and one
random matrix excluding singular and integers
random data

Random Matrix (excluding singular and zero)

See Also

Syntax

matrix rmatexs0(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.
matrix will not be singular if square and will not have a zero element

matrix functions
randomize
random
random matrix
random matrix excluding zero
random matrix excluding zero and one
random matrix excluding integers
random matrix excluding singular
random matrix excluding singular, zero and one
random matrix excluding singular and integers
random data

Random Matrix (excluding singular and zero and one)

See Also

Syntax

matrix rmatexs01(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.
matrix will not be singular if square and will not have zero or one as elements

Random Matrix (excluding singular and integers)

See Also

Syntax

matrix rmatexsint(number n1, number n2, number n3)

Description

returns n1 x n2 matrix initialized with random numbers based on n3.
matrix will not be singular if square and will not have integer elements

matrix functions
randomize
random
random matrix
random matrix excluding zero
random matrix excluding zero and one
random matrix excluding integers
random matrix excluding singular
random matrix excluding singular and zero
random matrix excluding singular, zero and one
random data

Random Data

See Also

Syntax

matrix rnddat(matrix M, number σ)

Description

returns a matrix the same size as M, but each of whose elements is equal to the corresponding element of M plus σ times a random normal deviate with 0 mean and unit variance.

Display Matrix

See Also

Description

Ctrl+m brings up a dialog box to create a matrix of arbitrary size which is initialized manually.

matrix functions
zero matrix
identity matrix
random matrix

Diagonal

See Also

Syntax

matrix diag(matrix m)

Description

if m is a vector, diag returns a square matrix with the elements of m as diagonal element. If m is a matrix, diag returns a vector whose elements are the diagonal elements of m.

matrix functions

Ones Vector

See Also

Syntax

matrix onesvec(number n)

Description

creates an n x 1 vector all of whose elements are one

matrix functions

Maximum

See Also

Syntax

number max(matrix m)

Description

returns the maximum element of m.

matrix functions
minimum
sum **mean**
variance **standard deviation**
trace **slope intercept**

Minimum

See Also

Syntax

number min(matrix m)

Description

returns the minimum element of m

matrix functions
maximum
sum **mean**
variance **standard deviation**
trace **slope intercept**

Sum

See Also

Syntax

number sum(matrix m)

Description

returns the sum of the elements of m

matrix functions

maximum **minimum**

mean

variance **standard deviation**

trace **slope intercept**

Mean

See Also

Syntax

number mean(matrix m)

Description

returns the mean of the elements of m

matrix functions

maximum **minimum**

mean

variance **standard deviation**

trace **slope intercept**

Variance

See Also

Syntax

number var(matrix m)

Description

returns the variance of the elements of m

matrix functions

maximum	**minimum**
sum	**mean**
standard deviation	
trace	**slope intercept**

Standard Deviation

See Also

Syntax

number stddev(matrix m)

Description

returns the standard deviation of the elements of m

matrix functions

maximum	**minimum**
sum	**mean**
variance	
trace	**slope intercept**

Trace

See Also

Syntax

number tr(matrix m)

Description

returns the trace of m

matrix functions

maximum	**minimum**
sum	**mean**
variance	**standard deviation**

slope intercept

Slope intercept

See Also

Syntax

matrix slopeintercept(matrix x, matrix y)

Description

x and y are vectors of the same size. Corresponding elements of each are considered as coordinates (x, y) of points to which slopeintercept fits a straight line function (y=a x+b), returning a vector containing a and b.

matrix functions

maximum	**minimum**
sum	**mean**
variance	**standard deviation**
trace	

Reverse

See Also

Syntax

matrix reverse(matrix m)

Description

returns a matrix which is m with the order of its rows reversed.
if m is a vector, returns a vector which is m with the order of its elements reversed..

matrix functions
sort
column sort
row sort

Sort

See Also

Syntax

matrix sort(matrix v)

Description

returns a vector which is v with its elements sorted.

matrix functions
reverse
column sort
row sort

Column Sort

See Also

Syntax

matrix csort(matrix m, number n)

Description

returns a matrix which is m with its columns sorted based on the values in the nth row.

matrix functions
reverse
sort
column sort
row sort

Row Sort

See Also

Syntax

matrix rsort(matrix m, number n)

Description

returns a matrix which is m with its rows sorted based on the values in the nth column.

matrix functions
reverse
sort
column sort
row sort

Rows

See Also

Syntax

number rows(matrix m)

Description

returns the number of rows in the matrix m

matrix functions
cols

Columns

See Also

Syntax

number cols(matrix m)

Description

returns the number of columns in the matrix m

matrix functions
rows

Rank

See Also

Syntax

number rank(matrix m)

Description

returns the rank of m (ie the number of non-zero rows of the echelon form of m). If the rank of a square matrix is less than the number of rows, the matrix is singular (has no inverse).

matrix functions
echelon
echelon transform

Minor

See Also

Syntax

number minor(matrix m, number n1, number n2)

Description

returns the determinant of the submatrix formed by eliminating the n1th row and the n2th column of m.

matrix functions
adjoint

Adjoint

See Also

Syntax

matrix adj(matrix m)

Description

returns the adjoint matrix of m, ie the matrix whose elements are the transposed cofactors (signed minors) of its original elements. The adjoint matrix is related to the matrix inverse by Cramer's rule.

matrix functions
minor

Pivot

See Also

Syntax

matrix pivot(matrix m, number row, number col)

Description

performs a Gauss-Jordan pivot on the row, col element of m. Returns the result. The row, col element of m must not be zero. Gauss-Jordan elimination proceeds by dividing each element in the row of m containing the pivot element by the pivot element itself. This produces a one in the pivot column of the pivot row. Every other row of m is reduced by a multiple of the pivot row. The multiple for each row is the the element of the row that is in the same column as the pivot element. This has the effect of zeroing out the column above and below the pivot element.

matrix functions

Echelon

See Also

Syntax

matrix echelon(matrix m)

Description

returns the echelon form of m. If m has an inverse, the echelon form of m is the identity matrix.

matrix functions
echelon transform

Echelon Transformation Matrix

See Also

Syntax

matrix echtrans(matrix m)

Description

returns the matrix product of all the elementary row operations that convert m to echelon form. If m has an inverse, echtrans produces the inverse.

matrix functions
echelon

Particular Solution

See Also

Syntax

matrix psolve(matrix A, matrix b)

Description

The general solution to the matrix equation Ax=b when the equations are consistant and the solution is not unique is a linear combination of the particular solution plus any scalar multiple of the homogeneous solution, ie x = p + k h, where p is the particular solution, h is the homogeneous solution, and k is a scalar.

this function returns the particular solution

matrix functions
homogeneous solution

Homogeneous Solution

See Also

Syntax

matrix hsolve(matrix A)

Description

The general solution to the matrix equation Ax=b when the equations are consistant and the solution is not unique is a linear combination of the particular solution plus any scalar multiple of the homogeneous solution, ie x = p + k h, where p is the particular solution, h is the homogeneous solution, and k is a scalar.

this function returns the homogeneous solution

matrix functions
particular solution

Singular Value Decomposition

See Also

Syntax

matrix svd(matrix A, matrix u, matrix v)

Description

(The scratchpad does not currently do svd on complex matrices.)

Singular value decomposition is intended to transform a matrix A into the alternate representation : u ×d ×v , where d is diagonal and u and v are unitary (orthonormal). The return matrix w is a vector whose elements are the diagonal elements of d. The parameters u and v are modified as required.

Singular value decomposition is useful in determining the conditioning of a matrix and in the ill-conditioned case provides a method of correcting the ill-conditioning that produces a solution that is far superior to that obtained by more direct means. It should be the method of choice for least squares fitting.

Singular value decomposition is closely related to the topic of the natural inverse.

The natural inverse equals v ×di ×u , where di is a diagonal matrix with its elements equal to the inverse of the elements of d. (But if a diagonal element of d is zero, the corresponding element of di is also zero, not infinity!)

matrix functions
singular value decomposition backsubstitution
singular value decomposition variance covariance

Singular Value Decomposition Back Substitution

See Also

Syntax

matrix svdbksb(matrix u, matrix w, matrix v, matrix b)

Description

Singular value decomposition back substitution solves the matrix equation A x = b in a way that avoids some of the problems associated with ill-conditioning that troubles the standard method, ie $x := A^{-1} \times b$ for square A or $x := (A^T \times A)^{-1} \times A^T \times b$ for non-square A. Allow w to be the matrix returned from svd(A, u, v). w indicates the conditioning of A. The conditioning of A is the ratio of the largest element of w to the smallest element of w. Since the scratchpad calculates in double precision, a condition number greater than 10^{10} indicates an ill-conditioned matrix. (A singular matrix will have at least one element of zero and so will have an infinite condition number.)

The proper procedure for an ill conditioned matrix is to set the smallest elements of w to zero and then use the svdbksb routine to solve for x.

matrix functions
singular value decomposition **(natural inverse)**
singular value decomposition variance covariance

Singular Value Decomposition Variance Covariance

See Also

Syntax

matrix svdcovar(matrix w, matrix v)

Description

Singular value decomposition solves the matrix equation A x = b in a way that avoids some of the problems associated with ill-conditioning that troubles the standard method, ie $x := A^{-1} \cdot b$ for square A or $x := (A^T \cdot A)^{-1} \cdot A^T \cdot b$ for non-square A. Allow w to be the matrix returned from svd(A, u, v). w indicates the conditioning of A. The conditioning of A is the ratio of the largest element of w to the smallest element of w. Since the scratchpad calculates in double precision, a condition number greater than 10^{10} indicates an ill-conditioned matrix. (A singular matrix will have at least one element of zero and so will have an infinite condition number.)

The proper procedure for an ill conditioned matrix is to set the smallest elements of w to zero and then use the svdbksb routine to solve for x.

svdcovar returns the variance covariance matrix of A. This is $(A^T \cdot A)^{-1}$ in the standard matrix approach. The principal use of this procedure is in regression analysis.

matrix functions
singular value decomposition **(natural inverse)**
singular value decomposition backsubstitution

Eigenvalues & Eigenvectors

See Also

Syntax

matrix eigen(matrix m, matrix v)

Description

returns a column vector containing the eigenvalues of the square matrix m. On return v (which must be a matrix identifier) contains the eigenvectors of m. The i-th column of v contains the eigenvector corresponding to the i-th element of the return vector of eigenvalues.

Often referred to as characteristic values or latent values as well as eigenvalues.

matrix functions

Orthonormalization

See Also

Syntax

matrix orthonorm(matrix m)

Description

Gram-Schmidt orthonormalization.
returns a matrix whose columns provide an orthonormal basis for the vectors that are the columns of the original non-singular matrix m.

matrix functions

Statistical Functions

General

- histogram
- combinations
- permutations
- gamma
- incomplete gamma
- natural logarithm of gamma
- error

ANOVA

- one way ANOVA
- one way ANOVA vector form
- two way ANOVA
- two way ANOVA with interaction

Rank Statistics

- Wilcoxon rank sum
- Mann-Whitney
- signtest
- Wilcoxon signed rank
- Kruskal-Wallis
- Kruskal-Wallis vector form
- number of runs
- Friedman
- Spearmans Rank Correlation

Distributions

binomial	cumulative binomial	percentile binomial
geometric	cumulative geometric	percentile geometric
hypergeometric	cumulative hypergeometric	percentile hypergeometric
exponential	cumulative exponential	percentile exponential
normal	cumulative normal	percentile normal
chi-square	cumulative chi-square	percentile chi-square
	chi-square table	chi-square table expected
F	cumulative F	percentile F
Poisson	cumulative Poisson	percentile Poisson
student t	cumulative student t	percentile student t
	cumulative K-S	percentile K-S

Distributions come in two forms, discrete and continuous. Discrete means the result of the experiment is countable and is usually associated with problems involving "How many ..." or "the number of ...". Continuous means that the measurements are made with arbitrary precision. To help you distinguish, consider a measurement of 5 seconds. Five in this case does not have the same definiteness associated with 5 in a discrete measurement, eg 5 heads in a coin toss experiment. In the case of 5 seconds, the measurement is closer to 5 than to 4 or 6 seconds, but certainly is not **exactly** 5 seconds. There are no measuring devices that give exact results(implying an infinite number of decimal places) for continuous measurements! In the case of the coin toss experiment, however, we **do** mean **exactly** 5 heads!

Example probability formulas for discrete distributions(illustrated with the binomial distribution):

exactly x successes	prob_E:=binom(x, N, p)
less than or equal to x successes	prob_LTE:=cbinom(x, N, p)
less than x successes	prob_LT:=prob_LTE-prob_E
greater than x successes	prob_GT:=1-prob_LTE
greater than or equal to x successes	prob_GTE:=prob_GT+prob_E
between x and y successes (inclusive, y>x)	prob_B_IN:=prob_LTE(y)-prob_LT(x)
between x and y successes (exclusive, y>x)	prob_B_EX:=prob_LT(y)-prob_LTE(x)
x for which cumulative prob <= 0.99	pbinom(0.99, N, p)

For continuous distributions, the probability of any **exact** value is zero since the area under a point of the probability density function is zero. This means that prob_E must be zero. Also, prob_LT must equal prob_LTE, prob_GT must equal prob_GTE, and prob_B_IN must equal prob_B_EX.

Example probability formulas for continuous distributions(illustrated with the normal distribution):

exactly z	prob_E:=0
less than or equal to z	prob_LTE:=cnorm(z)
less than z	prob_LT:=prob_LTE
greater than z	prob_GT:=1-prob_LTE
greater than or equal to z	prob_GTE:=prob_GT
between x and y (y>x)	prob_B:=prob_LTE(y)-prob_LTE(x)
z for which cumulative prob <= 0.99	pnorm(0.99)

Histogram

See Also

Syntax

matrix hist(matrix intervals, matrix m)

Description

returns a histogram of the matrix m divided into intervals defined by the ordered elements of the vector intervals. The return vector has one less element than does intervals.

statistical functions

Permutations

See Also

Description

$$_{N}P_{x} = \frac{N!}{(N-x)!}$$

The hot key Alt+p produces the permutation function, which returns the number of different groupings of items taken x at a time from a set of N distinct items. This differs from combinations in that the order in which the items are selected is important, ie groupings containing the same elements but in differing order are considered distinct groupings.

statistical functions
combinations

Combinations

See Also

Description

$$C_{N\,x} = \frac{N!}{x!\,(N-x)!}$$

The hot key Alt+c produces the combination function, which returns the number of different groupings of items taken x at a time from a set of N distinct items. This differs from permutations in that the order in which the items are selected is not important.

Note that this formula is identical to that for calculating the number of permutations of N objects, x of one type of nondistinct object with the remainder being a separate type of nondistinct object, e.g. the number of permutations of N balls, x black and N-x white. We use this function in this respect when discussing the binomial and hypergeometric distributions.

statistical functions
permutations

Gamma

See Also

Syntax

number Γ(number n)

Description

returns the gamma function which is related to the factorial : Γ(n) = (n-1)!

statistical functions
incomplete gamma
natural logarithm of gamma

Incomplete Gamma

See Also

Syntax

number incΓ(number a, number x)

Description

returns the incomplete gamma function. The function has a limiting value of zero at x=0 and a limiting value of one as x approaches infinity. It is centered about (a-1) with width approximately sqrt(a)

Note that the chi-square function is a special case of the incomplete gamma.

Natural logarithm of the Gamma Function

See Also

Syntax

number lnΓ(number n)

Description

return the natural logarithm of the gamma function. This is useful for reals because large factorials can overflow whereas their logarithms may not, especially considering many times the equation of interest involves a ratio of factorials.

statistical functions
gamma
incomplete gamma

Error function

See Also

Syntax

number erf(number n)

Description

returns the value of the error function at n

statistical functions

one way ANOVA

See Also

Syntax

matrix anova1(matrix M)

Description

ANOVA is used to test whether the means of two or more populations are the same.
There are two forms of the one way ANOVA in the scratchpad. This form applies when all the samples are the same size.

M is constructed of columns of treatment data. Each column is a treatments data and the number of rows of M is the number of data items per treatment, which must be the same for all treatments. The matrix returned is a 3 x 4 ANOVA matrix with the following elements:

	sum of squares	degrees of freedom	mean square	statistic
treatments	TSS	c - 1	TMS	$F := \frac{TMS}{EMS}$
errors	ESS	(r - 1) * c	EMS	
total	SS	r * c - 1		

c is columns of M = number of treatments
r is rows of M = data items per treatment

statistical functions
one way ANOVA vector
two way ANOVA
two way ANOVA with interaction

one way ANOVA vector

See Also

Syntax

matrix anova1v(matrix nItems, matrix V)

Description

ANOVA is used to test whether the means of two or more populations are the same.
There are two forms of the one way ANOVA in the scratchpad. This form applies when all the samples are not the same size.

To use this form: First combine the samples into the single vector V. Form another vector nItems that has the same number of elements as there are samples. Then set each element of the nItems vector to the number of elements of each sample, in the same order that the samples were combined into V.

The matrix returned is a 3 x 4 ANOVA matrix with the following elements:

	sum of squares	degrees of freedom	mean square	test statistic
treatments	TSS	c - 1	TMS	$F := \frac{TMS}{EMS}$
errors	ESS	(r - 1) ×c	EMS	
total	SS	r ×c - 1		

c is columns of M = number of treatments
r is rows of M = data items per treatment

statistical functions
one way ANOVA
two way ANOVA
two way ANOVA with interaction

two way ANOVA

See Also

Syntax

matrix anova2(matrix M, number blockSize)

Description

ANOVA is used to test whether the means of two or more populations are the same. In two way ANOVA, the data items of each treatment are divided into blocks. The scratchpad requires that all the blocks be the same size.
M is constructed of columns of treatment data. Each column is a treatments data and the number of rows of M is the number of data items per treatment, which must be the same for all treatments. In addition, each treatments data is composed of an integral number of blocks of size blockSize elements. The matrix returned is a 4 x 4 ANOVA matrix with the following elements:

	sum of squares	degrees of freedom	mean square	test statistic
treatments	TSS	c - 1	TMS	$F_T := \frac{TMS}{EMS}$
blocks	BSS	b - 1	BMS	$F_B := \frac{BMS}{EMS}$
errors	ESS	n - c - b + 1	EMS	
total	SS	n - 1		

c is columns of M = number of treatments
b is the number of blocks
n is the total number of data items = r * c

statistical functions
one way ANOVA
one way ANOVA vector
two way ANOVA with interaction

two way ANOVA with interactions

See Also

Syntax

matrix anova2i(matrix M, number blockSize)

Description

ANOVA is used to test whether the means of two or more populations are the same. In two way ANOVA, the data items of each treatment are divided into blocks. The scratchpad requires that all the blocks be the same size. In two way ANOVA with interaction, there is assumed to be interaction between the blocks and treatments.
M is constructed of columns of treatment data. Each column is a treatments data and the number of rows of M is the number of data items per treatment, which must be the same for all treatments. In addition, each treatments data is composed of an integral number of blocks of size blockSize elements. The matrix returned is a 5 x 4 ANOVA matrix with the following elements:

	squares	freedom	square	statistic
treatments	TSS	c - 1	TMS	$F_T := \frac{TMS}{EMS}$
blocks	BSS	b - 1	BMS	$F_B := \frac{BMS}{EMS}$
interactions	ISS	(b - 1) ×(c - 1)	IMS	$F_I := \frac{IMS}{EMS}$
errors	ESS	n - c ×b	EMS	
total	SS	n - 1		

c is columns of M = number of treatments
b is the number of blocks
n is the total number of data items = r * c

statistical functions
one way ANOVA
one way ANOVA vector
two way ANOVA

statistically rank

See Also

Syntax

matrix statrank(matrix M)

Description

returns a matrix the same size as M, but whose elements correspond to the ranked elements of M ranging from 1 to rows(M) * cols(M)

statistical functions
Mann-Whitney

Wilcoxon rank sum

See Also

Syntax

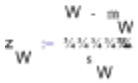
number ranksum(matrix A, matrix B)

Description

Used to determine if two continuous populations differ.

The elements of A and Bare pooled and the pool is ranked. ranksum returns the sum of the ranked elements that correspond to the elements of A. Both matrices must contain 10 elements or more.

Also,



statistical functions
Mann-Whitney

Mann-Whitney

See Also

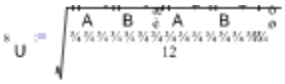
Syntax

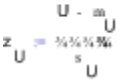
number mannwhitney(matrix A, matrix B)

Description

equivalent test to the Wilcoxon rank sum test

The W used here must have been calculated based on A, ie call ranksum(A, B), not ranksum(B, A)
Also,





statistical functions
Wilcoxon rank sum

sign test

See Also

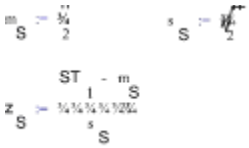
Syntax

matrix signtest(matrix A, matrix B)

Description

returns a 2 element vector. Element 1 contains the number of positive differences between the elements of the vectors A and B. Element 2 contains the number of nonzero differences between the elements of A and B

Then
ST= the vector returned from signtest
S is the number of positive differences
n is the number of nonzero differences, must be 10 or more



statistical functions

Wilcoxon signed rank test

See Also

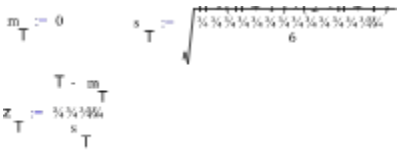
Syntax

matrix signedrank(matrix A, matrix B)

Description

returns a 2 element vector. Element 1 contains the sum of the signed ranks of the vectors A and B for pairs with nonzero differences. Element 2 contains the number of nonzero differences between the elements of A and B

Then for:
T=sum of signed ranks for the pairs with nonzero differences
n=number of nonzero differences, must be 10 or more



statistical functions

Kruskal-Wallis

See Also

Syntax

number kruskalwallis(matrix M)

Description

similar to the Wilcoxon rank sum test, but applies to more than two samples. There are two forms of the Kruskal Wallis test in the scratchpad. This form applies when all the samples are the same size and each sample is represented by a column of M.
Then,

is equivalent to

where n is the number of elements of M

statistical functions
kruskalwallisv

Kruskal-Wallis vector form

See Also

Syntax

number kruskalwallisv(matrix nItems, matrix V)

Description

similar to the Wilcoxon rank sum test, but applies to more than two samples. There are two forms of the Kruskal Wallis test in the scratchpad. This form applies when all the samples are not the same size.

To use this form: First combine the samples into the single vector V. Form another vector nItems that has the same number of elements as there are samples. Then set each element of the nItems vector to the number of elements of each sample, in the same order that the samples were combined into V.
Finally, compute:

statistical functions
kruskalwallis

number of runs test

See Also

Syntax

matrix numruns(matrix V, number separator)

Description

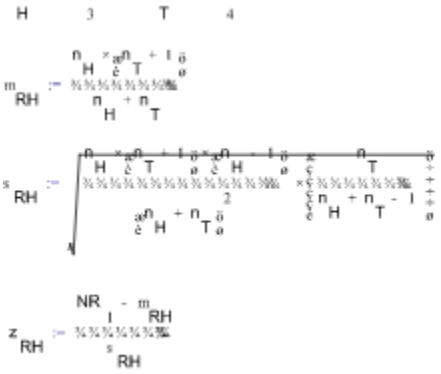
used to discover if the elements of a sequence are in random order.

passed a sequence vector V and a number that is used to divide the elements of V into two groups (head and tails), numruns returns a 4 element vector. Its elements are:

number of runs >= separator
number of runs < separator
number of elements of V >= separator (referred to below as H for heads)
number of elements of V < separator (referred to below as T for tails)

Let

then,



statistical functions

Friedman

See Also

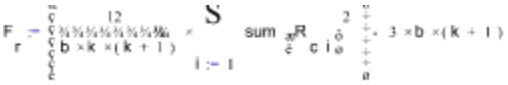
Syntax

number friedman(matrix M)

Description

another test used to test whether populations differ. The form in the scratchpad is used to analyze a randomized block design with b blocks and k treatments.

It calculates



where R is a matrix of the same size as M and

statistical functions

Spearmans rank correlation

See Also

Syntax

number rankcorr(matrix A, matrix B)

Description

returns a measure of how well the two ranking vectors A and B are correlated. This measure ranges from -1 to 1 with 0 indicating no correlation and 1 indicating perfect correlation

statistical functions

Binomial distribution

See Also

Syntax

number binom(number x, number N, number p)

Description

The binomial distribution is discrete:

$$prob(x) = {}_{N}C_{x} \times p^{x} \times q^{N-x}$$

where x is the number of successes
N is the number of independent trials
p is the probability of success per trial
q is the probability of failure per trial (1-p).
C is the number of combinations of N distinct objects taken x at a time.

Also,

$$m = N \times p \qquad s = N \times p \times q$$

binom(x, N, p) returns the probability of exactly x successes in N trials given p, the probability of success per trial. Note that although the binomial coefficient is replaced here with the Combinations term, you should realize that a replacement has been made (there is no binomial coefficient function in the scratchpad) and that the intention here is to count the number of **permutations** of **nondistinct** objects and that it is merely accidental that the formula for permutations of two types of nondistinct objects (all successes are alike and all failures are alike) is identical to that for **combinations** of **distinct** objects taken x at a time.

statistical functions
combinations
cumulative binomial
percentile binomial

Cumulative binomial

See Also

Syntax

number cbinom(number x, number N, number p)

Description

returns the cumulative probability of x successes or less given N and p.

statistical functions
binomial
percentile binomial

Percentile binomial

See Also

Syntax

number pbinom(number cumProb, number N, number p)

Description

returns the x value that corresponds to the desired cumulative probability, ie the x value such that cumProb = cbinom(x, N, p)

statistical functions
binomial
cumulative binomial

Geometric distribution

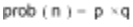
See Also

Syntax

number geom(number n, number p)

Description

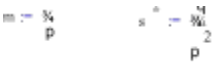
The geometric distribution is discrete:



where n is the number of trials required to get a successful outcome
p is the probability of success on a single trial
q is the probability of failure per trial (1-p).

geom(n, p) returns the probability that success will first occur on trial number n.

Also,



statistical functions
cumulative geometric
percentile geometric

Cumulative geometric

See Also

Syntax

number cgeom(number n, number p)

Description

returns the cumulative probability that success will occur in n trials or less given the probability per trial p.

statistical functions
geometric
percentile geometric

Percentile geometric

See Also

Syntax

number pgeom(number cumProb, number p)

Description

returns the n value that corresponds to the desired cumulative probability given the probability per trial p., ie the number of trials n expected to be required to yield success such that cumProb = cgeom(n, p)

Note that this is strictly a discrete function and although the value returned will likely be between two integers, the fractional part of this value results from a simple linear interpolation between the bounding integers.

statistical functions
geometric
cumulative geometric

Hypergeometric distribution

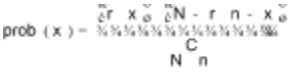
See Also

Syntax

number hgeom(number x, number n, number N, number r)

Description

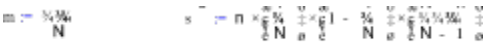
The hypergeometric distribution is discrete:



where x is the number of successes contained in the n drawn elements.
n is the number of elements drawn from the total N.
N is the total number of elements.
r is the total number of successes contained in N.
C is the number of combinations of N distinct objects taken n at a time.

hgeom(x, n, N, r) returns the probability of exactly x successes contained in a draw of n elements without replacement from a pool of N, given that only r of the N are successes. Note that the replacement of the binomial coefficient with the Combination function as discussed under the binomial distribution applies here as well.

Also,



You should see the strong similarity to the binomial mean and variance. The final multiplicative term in the variance formula is called the finite population correction factor.

statistical functions
combinations
cumulative hypergeometric
percentile hypergeometric

Cumulative hypergeometric

See Also

Syntax

number chgeom(number x, number n, number N, number r)

Description

returns the cumulative probability of x successes or less with a draw of n elements without replacement from a pool of N, given that only r of the N are successes.

statistical functions
hypergeometric
percentile hypergeometric

Percentile hypergeometric

See Also

Syntax

number phgeom(number cumProb, number n, number N, number r)

Description

returns the x value that corresponds to the desired cumulative probability cumProb, ie the number of successes with a draw of n elements without replacement from a pool of N, given that only r of the N are successes such that cumProb=chgeom(x, n, N, r)

Note that this is strictly a discrete function and although the value returned will likely be between two integers, the fractional part of this value results from a simple linear interpolation between the bounding integers.

statistical functions
hypereometric
cumulative hypergeometric

Exponential distribution

See Also

Syntax

number expon(number x, number m)

Description

The exponential distribution is continuous. It is sometimes called the "waiting time distribution". It is closely related to the Poisson distribution. The Poisson distribution is used when concerned with the number of events occurring within a specific space or time, the exponential distribution when concerned with the interval between events.

expon(x, m) returns value of the exponential probability distribution at x, given m, the distribution mean and standard deviation.

Also,

statistical functions
cumulative exponential
percentile exponential

Cumulative exponential

See Also

Syntax

number cexpon(number x, number m)

Description

returns the area under the standard exponential distribution curve from zero to x, given m, the distribution mean and standard deviation.

statistical functions
exponential
percentile exponential

Percentile exponential

See Also

Syntax

number pexpon(number cumProb, number m)

Description

returns the x value that corresponds to the desired cumulative probability cumProb, ie the x value such that cumProb=cexpon(x, m).

statistical functions
exponential
cumulative exponential

Normal distribution

See Also

Syntax

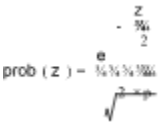
number norm(number z)

Description

The normal distribution is continuous

where x is the normal random variable. Under the transformation:

the normal distribution is transformed to the "standard" normal distribution:



norm(z) returns the value of the normal probability distribution at z, given the distribution mean and standard deviation. The standard normal function is symmetric about zero and normalized.

Also useful for the transformed "standard" normal distribution:

statistical functions
cumulative normal
percentile normal

Cumulative normal

See Also

Syntax

number cnorm(number z)

Description

returns the area under the standard normal distribution curve from -infinity to z.

statistical functions
normal
percentile normal

Percentile normal

See Also

Syntax

number pnorm(number cumProb)

Description

returns the z value that corresponds to the desired cumulative probability cumProb, ie the z value such that cumProb = cnorm(z).

statistical functions
normal
cumulative normal

Chi-square distribution

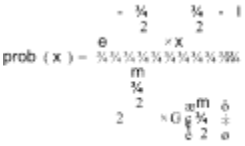
See Also

Syntax

number chisq(number x, number m)

Description

The chi-square distribution is continuous.



where x is the particular value of chi-square
m is the number of degrees of freedom

chisq(x, m) returns the value of the chi-square probability distribution at x(chi-square) given m (degrees of freedom). This function is normalized (area under curve from zero to infinity equals one) and skew. In practice, a chi-square value is calculated by a separate formula for a sample and the result is compared to this theoretical distribution.

Also,

Cumulative chi-square

See Also

Syntax

number cchisq(number x, number m)

Description

returns the area under the chi-square distribution function curve from zero to x for any particular value of m.

statistical functions
chi-square
percentile chi-square
chi-square table
chi-square table expected

Percentile chi-square

See Also

Syntax

number pchisq(number cumProb, number m)

Description

returns the chi-square value that corresponds to the desired cumulative probability cumProb for the particular number of degrees of freedom m, ie the x value such that cumProb = cchisq(x, m)

statistical functions
chi-square
cumulative chi-square
chi-square table
chi-square table expected

Chi-square table

See Also

Syntax

matrix chisq_t(matrix table)

Description

returns a three column matrix. table is a contingency table.

The first column of the returned matrix is the observed data, ie the matrix elements of table put into a single column. The second column of the returned matrix is the expected value of the matrix element contained in the first column. All expected values should be 5 or more for a valid test. The third column element of each row of the returned matrix is the square of the difference between the observed and expected values divided by the expected value. Tables in your textbook are likely to provide additional columns for the difference and the difference squared, but this 3 column vector should suffice to provide a check of your calculations. You should sum the elements of the third column of this matrix to obtain the chi-square value for the contingency table.

statistical functions
chi-square
cumulative chi-square
percentile chi-square
chi-square table expected

Chi-square table expected

See Also

Syntax

matrix chisq_te(matrix table)

Description

returns a three column matrix. table is a contingency table. The final row of table must contain the expected probabilities of the elements in each column. This row is only known when you have some prior knowledge of the population. The probabilities in this row must total one.

The first column of the returned matrix is the observed data, ie the matrix elements of table put into a single column. The second column of the returned matrix is the expected value of the matrix element contained in the first column. All expected values should be 5 or more for a valid test. The third column element of each row of the returned matrix is the square of the difference between the observed and expected values divided by the expected value. Tables in your textbook are likely to provide additional columns for the difference and the difference squared, but this 3 column vector should suffice to provide a check of your calculations. You should sum the elements of the third column of this matrix to obtain the chi-square value for the contingency table.

statistical functions
chi-square
cumulative chi-square
percentile chi-square
chi-square table

F distribution

See Also

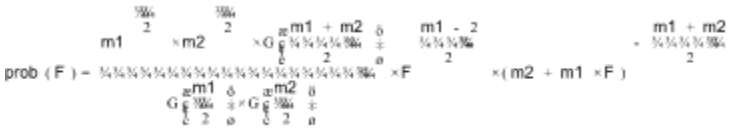
Syntax

number fdist(number F, number m1, number m2)

Description

The F distribution is continuous.

Given u, a statistic with m1 degrees of freedom, and v, a statistic with m2 degrees of freedom, with both u and v possessing independent chi-square distributions, then with:



fdist(F, m1, m2) returns the value of the F distribution at a particular F, given m1 and m2, the degrees of freedom of the numerator and denominator of F respectively. The function is normalized, ie the total area under the curve from zero to infinity equals one. The F distribution is often used to test whether two sample variances were taken from populations with the same variance.

statistical functions
cumulative F
percentile F
gamma

Cumulative F

See Also

Syntax

number cfdist(number F, number m1, number m2)

Description

returns the area under the F distribution function curve from zero to F for particular values of m1 and m2. Since the function is normalized, this area can be considered the probability that observed statistics u and v would result in this F or less. (The numerator of F is u divided by m1. The denominator of F is v divided by m2.)

Percentile F

See Also

Syntax

number pfdist(number cumProb, number m1, number m2)

Description

returns the F value that corresponds to the desired cumulative probability at particular m1 and m2, ie the F value such that cumProb = cfdist(F, m1, m2).

statistical functions
F
cumulative F

Poisson distribution

See Also

Syntax

number poiss(number x, number μ)

Description

The Poisson distribution is a discrete distribution. It is closely related to the exponential distribution. The Poisson distribution is used when concerned with the number of events occurring within a specific space or time, the exponential distribution when concerned with the interval between events.

$$\text{prob}(x) = \frac{m^{x}}{x!} \times e^{-m}$$

where x is the number of events
μ is the mean number of events per unit time (distance, area, volume, etc.)

poiss(x, μ) returns the probability of exactly x successes in a selected unit of space or time knowing only the mean number of occurrences per unit space or time.

Also,

$$s := m$$

statistical functions
cumulative Poisson
percentile Poisson

Cumulative Poisson

See Also

Syntax

number cpoiss(number x, number μ)

Description

returns the cumulative probability of x successes or less given μ.

statistical functions
Poisson
percentile Poisson

Percentile Poisson

See Also

Syntax

number ppoiss(number cumProb, number μ)

Description

returns the x value that corresponds to the desired probability, ie the x value such that cumProb=cpoiss(x,μ)

statistical functions
Poisson
cumulative Poisson

Student t distribution

See Also

Syntax

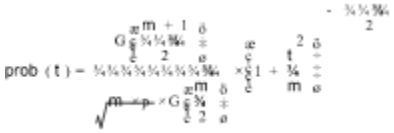
number studt(number t, number m)

Description

The student t distribution is continuous. It is used for small sample sizes where the sample standard deviation provides a poor estimate of the population standard deviation leading to inaccurate z values.

Given u and v as independently distributed statistics, u normally distributed with zero mean and unit variance, and v squared having a chi-square distribution with m degrees of freedom, consider:

then:



studt(t, m) returns the value of the student t probability distribution at a particular value of t, given m. The function is symmetric about zero and normalized. An important t is:

where s is the sample standard deviation. You can immediately see the similarity of t to the z of the normal distribution. In this case the number of degrees of freedom m=n-1.

Also useful for this "standard" student t distribution:

statistical functions
cumulative student t
percentile student t
gamma

Cumulative Student t

See Also

Syntax

number cstudt(number t, number m)

Description

returns the area under the student t distribution function curve from -infinity up to t for a particular m.

statistical functions
student t
percentile student t

Percentile Student t

See Also

Syntax

number pstudt(number cumProb, number m)

Description

returns the t value that corresponds to the desired cumulative probability at a particular m, ie the t value such that cumProb=cstudt(t, m).

statistical functions
student t
cumulative student t

Cumulative Kolmogorov-Smirnov

See Also

Syntax

number ck_s(number d, number n)

Description

The Kolmogorov-Smirnov statistic d is the maximum absolute value of the difference between the sample distribution function and the proposed hypothetical distribution function. This function returns the probability that the observed d would result in this difference or less. (d is between zero and one)

Two different approximations for this function are used. For n less than or equal to one hundred, the approximation is more accurate but slower to calculate. Above one hundred the limiting distribution at infinity is applied to the particular value of n.

The return values from this function should only be considered valid for cumulative probabilities greater than 0.80 although you will receive no warning for smaller values.

statistical functions
percentile Kolmogorov-Smirnov

Percentile Kolmogorov-Smirnov

See Also

Syntax

number pk_s(number cumProb, number n)

Description

returns the d value that corresponds to the desired cumulative probability at a particular n, ie the d value such that cumProb=ck_s(d, n). (cumProb is restricted to values between 0.80 and one)

Two different approximations for this function are used. For n less than or equal to one hundred, the approximation is more accurate but much slower to calculate. Above one hundred the limiting distribution and infinity is applied to the particular value of n.

If you don't require the extra accuracy and don't want to wait, you can force the limiting distribution evaluation to be applied for n less than or equal to one hundred by using the following formula:

statistical functions
cumulative Kolmogorov-Smirnov

Regression (Curve Fitting)

See Also

Description

Mathematical models describe how variables are related. Models can be created based on an underlying knowledge of the data or merely to simplify data analysis. Regression analysis is concerned with matrix equations of the form $Y=X\beta$, where Y represents some class of data and X is a matrix of values related to a different class of data. It is important to remember that X is not restricted to data values themselves, but can be any calculable function of the data. The important thing is that the equation is linear in terms of the coefficients β.

Regression analysis is a multistage process in the scratchpad, with function names regressa, regressb, and regressc.
1: construct the X matrix from the data matrix, which will usually at least involve adding a column of ones which corresponds to having an intercept constant in the model.
2: use either singular value decomposition or standard matrix techniques to determine the coefficient matrix b and the variance covariance matrix (svd is recommended to overcome problems with ill-conditioning (multicollinearity)).
3: use regressa to produce an analysis of variance table which enables you to determine how well the model coefficients explain the variation in the data.
4: use regressb to produce an analysis of the errors of β.
5. use regressc to produce an analysis of the errors of predicting y values corresponding to particular x values..

regression nalysis of variance
analysis of regression coefficients
standard deviations for mean and individual y

Regression Analysis of Variance

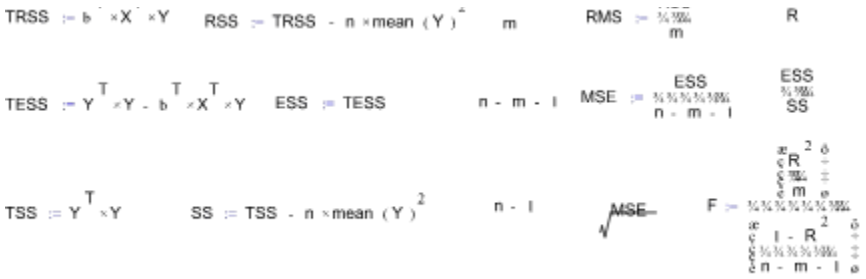
See Also

Syntax

matrix regressa(matrix X, matrix β, matrix Y)

Description

one step of the regression analysis process. X, b, and Y should have been previously calculated. regressa returns a 3x5 ANOVA matrix containing the following terms.



where m is the number of coefficients in β less one (for the intercept).

R is the sample coefficient of multiple determination It approaches one as the data agree with the model. MSE is the estimator of variance for the model. Its square root is called the standard error of the estimate of Y. F provides a global test of the significance of the regression (null hypotheses is that all elements of β are zero).

Regression(Curve Fitting)
analysis of regression coefficients
standard deviations for mean and individual y

Analysis of Regression Coefficients

See Also

Syntax

matrix regressb(matrix β, matrix covar, number variance)

Description

returns a matrix with three columns.
Column 1 repeats the coefficient matrix b for convenience.
Column 2 contains the standard errors of the elements of β.
Column 3 contains the t values associated with the elements of β,
ie. for each row, column 3 contains column 1 divided by column 2.

This data enables you to obtain confidence intervals for the elements of β. Note that using MSE to estimate the population variance amounts to an assumtion that the model is correct!

Regression(Curve Fitting)
analysis of variance
standard deviations for mean and individual y

Standard Deviations for Mean and Individual Y

See Also

Syntax

matrix regressc(matrix x, matrix covar, number variance)

Description

returns a matrix with two columns.
Column 1 contains the standard errors of the predicted mean value of y for each x row.
Column 2 contains the standard errors of the predicted individual value of y for each x row.

This lets you create confidence intervals for predicted mean and individual values of y given x.
Beware of errors due to extrapolation!

Regression(Curve Fitting)
analysis of variance
analysis of regression coefficients

Financial Functions

Five-Value Problems

See Also

Syntax

matrix fiveval(matrix v, number i)
matrix fivevalb(matrix v, number i)

Description

Many financial problems involve five interdependent variables, any one of which is uniquely determined by the other four. The five are :
1: present value
2 future value
3 number of payment periods
4 interest per payment period
5 payment amount

To solve problems of this sort, first create a vector of five elements and set the four known values (the elements of the vector must be in the order presented above). Then call the function fiveval(v, n) where v is the five vector and n is the vector element index of the unknown. The function returns a new five vector which includes the correct value for the unknown element.

fiveval(v, n) should be used for ordinary annuity calculations (payment at end of period)
fivevalb(v, n) should be used for annuity due calculations (payment at beginning of period)

financial functions
interest associated with file value problems
principal associated with five value problems

Interest Calculation for Five-Value Problems

See Also

Syntax

number ifiveval(matrix v, number p1, number p2)
number ifivevalb(matrix v, number p1, number p2)

Description

returns the interest paid (or earned) from payment p1 to payment p2 inclusive with the financial parameters as given in the five value vector v whose elements are:
1: present value
2 future value
3 number of payment periods
4 interest per payment period
5 payment amount

ifiveval(v, p1, p2) should be used for ordinary annuity calculations (payment at end of period)
ifivevalb(v, p1, p2) should be used for annuity due calculations (payment at beginning of period)

financial functions
five value problems
principal associated with five value problems

Principal Calculation for Five-Value Problems

See Also

Syntax

number pfiveval(matrix v, number p1, number p2)
number pfivevalb(matrix v, number p1, number p2)

Description

returns the principal paid (or earned) from payment p1 to payment p2 inclusive with the financial parameters as given in the five value vector v whose elements are:
1: present value
2 future value
3 number of payment periods
4 interest per payment period
5 payment amount

pfiveval(v, p1, p2) should be used for ordinary annuity calculations (payment at end of period)
pfivevalb(v, p1, p2) should be used for annuity due calculations (payment at beginning of period)

financial functions
five value problems
interest associated with file value problems

EFF to APR

See Also

Syntax

number EFFtoAPR(number EFF, number n)

Description

The APR is the annual percentage rate. It is the interest rate per compounding period multiplied by the number of compounding periods per year.

The EFF is the annual effective rate. It is the interest rate that produces the same interest as the APR, but with a single annual compounding period.

n is the number of compounding periods per year for the APR. APR and EFF are in percent.
return APR

financial functions

APR to EFF

See Also

Syntax

number APRtoEFF(number APR, number n)

Description

The APR is the annual percentage rate. It is the interest rate per compounding period multiplied by the number of compounding periods per year.

The EFF is the annual effective rate. It is the interest rate that produces the same interest as the APR, but with a single annual compounding period.

n is the number of compounding periods per year for the APR. APR and EFF are in percent.
return EFF

financial functions

Interpolation and Extrapolation Functions

linear interpolation
matrix interpolation using spline matrix
single point interpolation using spline matrix

Note: extrapolation is treated identically to interpolation but is a far more error prone process and should not be used much past the sample data range

linear spline
parabolic spline
cubic spline

Linear Interpolation

See Also

Syntax

number linterp(matrix x, matrix y, number xinterp)

Description

returns the linearly interpolated value of y at xinterp given the known x and y vectors.

interpolation and extrapolation
matrix interpolation using spline matrix
single point interpolation
linear spline
parabolic spline
cubic spline

Matrix Interpolation

See Also

Syntax

matrix minterp(matrix sv, matrix x, matrix y, matrix xvinterp)

Description

returns a vector of interpolated y values associated with the elements of the xvinterp vector, given the known x and y vectors and the previously calculated spline vector sv.

interpolation and extrapolation
single point interpolation
linear interpolation
linear spline
parabolic spline
cubic spline

Single point interpolation

See Also

Syntax

number interp(matrix sv, matrix x, matrix y, number xinterp)

Description

returns a single interpolated y value associated with the number xinterp, given the known x and y vectors and the previously calculated spline vector sv.

interpolation and extrapolation
linear interpolation
matrix interpolation using spline matrix
linear spline
parabolic spline
cubic spline

Linear Spline

See Also

Syntax

matrix lspline(matrix x, matrix y)

Description

returns a vector of coefficients of the spline with linear ends associated with the vectors x and y. This spline vector is meant to be used with the matrix interpolation function minterp.

Parabolic Spline

See Also

Syntax

matrix pspline(matrix x, matrix y)

Description

returns a vector of coefficients of the spline with parabolic ends associated with the vectors x and y. This spline vector is meant to be used with the matrix interpolation function minterp.

interpolation and extrapolation
linear interpolation
matrix interpolation using spline matrix
single point interpolation
linear spline
cubic spline

Cubic Spline

See Also

Syntax

matrix cspline(matrix x, matrix y)

Description

returns a vector of coefficients of the spline with cubic ends associated with the vectors x and y. This spline vector is meant to be used with the matrix interpolation function minterp.

interpolation and extrapolation
linear interpolation
matrix interpolation using spline matrix
single point interpolation
linear spline
parabolic spline

Logical Functions

- if

if

See Also

Syntax

number if(bool b, number n1, number n2)

Description

returns n1 if b is true, n2 if b is false

logical functions

Fast Fourier Transform Functions

Fourier transform functions are most often used when converting between time and frequency domains.

The formula used by the scratchpad when converting data to transformed data is:

and the inverse transform is:

The alternate forms of these functions are calculated as follows:

with an inverse transform of:

Fast Fourier Transform

See Also

Syntax

matrix fft(matrix m)

Description

returns the Fourier transformed vector of the original vector m. fft requires that the number of elements of the original m vector be a power of two. Uses the symmetric form.

fast fourier transform functions
inverse fast fourier transform

Inverse Fast Fourier Transform

See Also

Syntax

matrix ifft(matrix m)

Description

returns the inverse Fourier transformed vector of vector m. ifft(fft(data))=data. ifft requires that the number of elements of the m vector be a power of two. Uses the symmetric form.

fast fourier transform functions
fast fourier transform

Fast Fourier Transform (alternate form)

See Also

Syntax

matrix FFT(matrix m)

Description

returns the Fourier transformed vector of the original vector m. FFT requires that the number of elements of the original m vector be a power of two. Uses the alternate form.

fast fourier transform functions
inverse fast fourier transform

Inverse Fast Fourier Transform (alternate form)

See Also

Syntax

matrix IFFT(matrix m)

Description

returns the inverse Fourier transformed vector of vector m. IFFT(FFT(data))=data. IFFT requires that the number of elements of the m vector be a power of two. Uses the alternate form.

fast fourier transform functions
fast fourier transform

Linear Programming Functions

Linear Programming Maximize

See Also

Syntax

number lpmax(matrix c, matrix A, matrix r, matrix b,
matrix x, matrix tableau)

Description

returns the value of the maximized objective function and modifies the inputs x and tableau appropriately.
Objective Function
c is a vector of objective function coefficients.
Constraints
The constraints are of the form A r b
A is a matrix of the constraint coefficients.
r is a vector representing the constraint relationships:
(The r values are the same as the slack variable coefficients for positive b.)
less than or equal relationship is indicated by 1
equal relationship is indicated by 0
greater than or equal relationship is indicated by -1
b is a vector of the constraint constants
(non-negativity constraints are implied,ie not explicitly stated)
Modified Parameters
x is the solution vector that maximizes the objective function.
tableau is the final simplex tableau.

linear programming functions
linear programming minimize

Linear Programming Minimize

See Also

Syntax

number lpmin(matrix c, matrix A, matrix r, matrix b,
matrix x, matrix tableau)

Description

returns the value of the minimized objective function and modifies the inputs x and tableau appropriately.
Objective Function
c is a vector of objective function coefficients.
Constraints
The constraints are of the form A r b
A is a matrix of the constraint coefficients.
r is a vector representing the constraint relationships:
(The r values are the same as the slack variable coefficients for positive b.)
less than or equal relationship is indicated by 1
equal relationship is indicated by 0
greater than or equal relationship is indicated by -1
b is a vector of the constraint constants
(non-negativity constraints are implied,ie not explicitly stated)
Modified Parameters
x is the solution vector that minimizes the objective function.
tableau is the final simplex tableau.

linear programming functions
linear programming maximize

Accelerators (Hot Keys)

Keystroke	Interpretation
F1	help contents
Ctrl+F1	keyword help, depends on caret position
F3	repeat last find or replace
Alt+F4	exit
F5	display value, same as Ctrl+=
F7	compile and align
F8	compile
F9	run
F10	run and align
Ctrl+1,2,3,4 or 5	set jump marker
Ctrl+a	and
Ctrl+c	complex conjugate
Ctrl+d	derivative
Ctrl+e	exchange
Ctrl+g	convert next character to Greek alphabet equivalent
Ctrl+h	backspace
Ctrl+i	integral
Ctrl+j	square root of minus one, can be displayed as i or j
Ctrl+m	display matrix
Ctrl+n	not
Ctrl+o	or
Ctrl+p	iterated product
Ctrl+r	square root
Ctrl+s	iterated sum
Ctrl+t	matrix transpose
Ctrl+w	mixed number (whole)
Alt+1,2,3,4 or 5	jump to jump marker
Alt+a	augment
Alt+c	combinations
Alt+d	literal subscript (down)
Alt+i	infinity character, but no value is pre-assigned!
Alt+m	brings up a dialog to change the size of a display matrix, if the caret is positioned within a display matrix
Alt+o	degree character
Alt+p	permutations
Alt+u	literal superscript (up)
Alt+x	vector cross product
Alt+*	vector dot product
Alt+<	angle character
Ctrl+=	display the calculated value of the expression, slows calculations if used in a loop
Alt+=	exchange

Grouping Pairs: The scratchpad will force the number of right chars of each grouping pair to equal the number of left chars by adding or deleting right chars. This has the potential to be the most troubling aspect of using the scratchpad. You must practice until you fully understand the use of parentheses. You cannot avoid this! Tip: Always begin by placing the left grouping char first! When editing, if you need to reposition a left grouping char, delete the old one first and then add new left and right chars. If you need only to move a right char farther left, you can usually simply insert it in the proper position. If you need to move it farther right, delete the existing right char first and then insert a new one as needed. Ignore any

intermediate rearrangement. The other grouping pairs are balanced just like parentheses. Dont forget you can toggle the display of invisible chars by right clicking on the active expression.

( )	most common grouping pair
{ }	absolute value, complex magnitude, matrix determinant
[ ]	matrix or vector element subscript grouping pair
Ctrl+[Ctrl+]	column subvector grouping pair
Alt+[Alt+]	row subvector grouping pair
"	text entry
Ctrl+Enter	hard page break
Alt+Enter	exit a text object from the keyboard

right click — if within an active object, allows you to modify its properties
left click — activates an inactive object or positions caret within an already active object

if an object has been previously selected

Shift+left click — selects a range of objects
Ctrl+left click — adds or removes an object from the list of selected objects

selected objects may be deleted, cut to the clipboard ,dragged(hold left button down) to a new position, or have their properties modified as a group

Caret Control

Key	Active Expression	None Active
Home	beginning of expression	beginning of document
End	end of expression	end of document
←	character left	character left
→	character right	character right
Ctrl+←	jump left to previous subexpression (graphs, integrals, permutations, etc.)	document left
Ctrl+→	jump right to next subexpression (graphs, integrals, permutations, etc.)	document right
	move up in matrix or previous in graph	character up
↓	move down in matrix or next in graph	character down
Ctrl+	none	top of screen
Ctrl+↓	none	bottom of screen
Tab	next expression	none
Shift+Tab	previous expression	none
PageUp	none	up one page
PageDown	none	down one page

You can also scroll by dragging the mouse, whether objects are selected or not.

Predefined Constants

e	natural logarithm base
π	pi (entered with Ctrl+g followed by p)
i or j	square root of minus one (entered as Ctrl+j whether choose to display as i or j)
TOL	TOL effects internal calculations initially set at 0.001
true	logical constant
false	

Greek Letters

Greek characters are inserted into your expressions by typing two characters.
The first character must be Ctrl+g, the second the English equivalent as shown below.

alpha	a	α	A	A
beta	b	β	B	B
chi	c	χ	C	X
delta	d	δ	D	Δ
epsilon	e	ε	E	E
phi	f	ϕ	F	Φ
gamma	g	γ	G	Γ
eta	h	η	H	H
iota	i	ι	I	I
	j	φ	J	ϑ
kappa	k	κ	K	K
lambda	l	λ	L	Λ
mu	m	μ	M	M

nu	n	ν	N	N
omicron	o	ο	O	O
pi	p	π	P	Π
theta	q	θ	Q	Θ
rho	r	ρ	R	P
sigma	s	σ	S	Σ
tau	t	τ	T	T
upsilon	u	υ	U	Y
	v	ϖ	V	ς
omega	w	ω	W	Ω
xi	x	ξ	X	Ξ
psi	y	ψ	Y	Ψ
zeta	z	ζ	Z	Z

Programming Commands

Programming Overview

proc	return	endproc	
if	else	endif	
matloop	endmatloop		
while	endwhile	break	continue
comments	endcomments		
autosubblock	endautosubblock		
autosub	endautosub		
show	hide		

Programming Overview

See Also

I use the terms proc, procedure and function interchangeably.

Any time you use the scratchpad as a calculator, you are writing and executing a program. This is true even if you are in scratchpad mode. This might surprise you if you have been using the scratchpad for a while and had not thought about the fact that this was really a simple form of computer programming.

The scratchpad has a special Program Mode that you can use to extend the capabilities of the scratchpad or to just experiment with writing procedures. You enter/exit this mode by selecting Program/Program Mode from the menu.

When you enter program mode, you must at a minimum create a void Main() procedure and a matching endproc. You have to do this before you are able to run a program. Most scratchpad commands are available without entering Program Mode. All program modes are entered into the scratchpad in the same way: type the lowercase keyword and press enter. The keyword will change into a comand automatically (indicated by changing to blue). Procedures have a special input method discussed under the topic proc.

Procedures must have at least one parameter! This differs from most programming languages, but since the primary use of empty parameter lists is to call system functions, it shouldn't be much of an inconvenience - especially neither global or static variables are not allowed. (Most functions need some data to operate on.)

Execution begins with Main() and continues to the endproc associated with Main(). Parameters are passed by reference, ie the subroutine can modify the original values directly. If you wish to protect your original parameters, make copies in either the called or calling routine.

Note that the only way to remove program commands from the scratchpad is by selecting them with the mouse and then deleting them.

Programming Commands

proc

See Also

Procedure declarations have their own unique input method. First type proc followed by enter. A procedure declaration will appear in blue if you have done this properly. You will need to complete the declaration yourself. It must be of the following form:

returnType name(param1Type param1Name, param2Type param2Name, ...)

where the return and parameter types are one of the following: bool, number, matrix
The return type may also be void. You will only need to enter the first letter of one of these types.

There must be at least one parameter! This differs from most programming languages, but in this simple language it should not be much of an inconvenience. (Most functions need some data to operate on, but if you really need such a function, simply give it a dummy numeric parameter and call it with a zero for the dummy parameter)

Initially, a new proc will not contain any parameters. This allows you to type the function void Main() (the m in Main must be capitalized) which has no parameters. You add an initial parameter by pressing insert when the caret is left of the functions left parentheses and additional parameters by pressing insert when the caret is just right of the preceding parameter name. You delete parameters by pressing backspace when the caret is just left of the parameter type.

Each proc must be paired with an endproc. If a return type is specified other than void, the procedure must contain a return statement.

Programming Commands
Programming Overview
return
endproc

return

See Also

The return statement is used to exit a procedure before the endproc statement is encountered. The value of the next constant or identifier following the return is returned to the calling procedure.

Programming Commands

proc

endproc

endproc

See Also

The endproc statement together with the proc declaration define the boundaries of the procedure. All statements must be in some procedure and each procedure declarations must have its own endproc.

if

See Also

The expression immediately following the if statement is evaluated. It must be a boolean(logical expression).

If it evaluates to true, program execution continues with the expression following the test boolean, continues on until either the else or endif associated with this if is encountered whereupon program execution continues with the expression following the endif associated with the if.

If it evaluates to false, program flow continues with the expression following the endif associated with the if.

else

See Also

optional statement within and if - endif programming structure. See if

Programming Commands
if
endif

endif

See Also

terminating statement of an if - endif programming structure. Each if must be paired with an endif.

Programming Commands

- if
- else

matloop

See Also

looping command similar to while. The expression following the matloop is evaluated and must be a matrix element identifier. The identifiers used in the matrix element subscript are then available for use within the loop. They will take on values that enable the loop to cover all the matrix elements. If the matrix element identifier is a vector, then only a single looping variable is required.

The effect of the matloop is that of two nested whiles with the outer loop over row indices and the inner loop over the column indices.

Programming Commands
endmatloop

endmatloop

See Also

terminating statement of a matloop - endmatloop programming structure. Each matloop must be paired with a endmatloop.

Programming Commands
matloop

while

See Also

The expression immediately following the while statement is evaluated. It must be a boolean(logical expression).

If it evaluates to true, program execution continues with the expression following the test boolean and continues on until it encounters the matching endwhile at which time the logical expression is reevaluated and program flow is decided as before.

If it evaluates to false, program execution continues with the expression following the associated endwhile.

Each while must be matched with an endwhile.

Programming Commands
endwhile
break
continue

endwhile

See Also

terminating statement of a while - endwhile programming structure. Each while must be paired with an endwhile.

Programming Commands
while
break
continue

break

See Also

causes execution to continue with the expression following the innermost enclosing endwhile or endmatloop

Programming Commands
while
endwhile
continue

continue

See Also

causes execution to continue with the innermost enclosing while or matloop where the loop condition is reevaluated.

Programming Commands
while
endwhile
break

comments

See Also

causes execution to continue with the expression immediately following the matching endcomments.
Each comments must be paired with a matching endcomments.

Programming Commands
endcomments

endcomments

See Also

terminating statement of a comments - endcomments programming structure. Each comments must be paired with an endcomments.

Programming Commands
comments

autosubblock

See Also

beginning statement of autosubblock - endautosubblock programming structure. Each autosubblock must be paired with an endautsubblock. Variable names placed within the autosubblock - endautosubblock pair will be replaced later in the program with their calculated values. The autosubstition begins immediately following the endautosublock command.

endautosubblock

See Also

terminating statement of autosubblock - endautosubblock programming structure. Each autosubblock must be paired with an endautsubblock. Variable names placed within the autosubblock - endautosubblock pair will be replaced later in the program with their calculated values. The autosubstition begins immediately following the endautosublock command.

autosub

See Also

Variable names placed within the autosubblock - endautosubblock pair can be replaced later in the program with their calculated values. Autosubstition automatically begins immediately following the endautosublock command, but can be stopped by an endautsub command. An autosub command reenables the autosubstitution.

endautosub

See Also

Variable names placed within the autosubblock - endautosubblock pair can be replaced later in the program with their calculated values. Autosubstition automatically begins immediately following the endautosublock command, but can be stopped by an endautsub command.

show

See Also

Expressions marked as display value (those with a red equal sign) will automatically display the expressions value when the program is run. The hide command can disable this behaviour. The show command reenables the effect of the red equal sign.

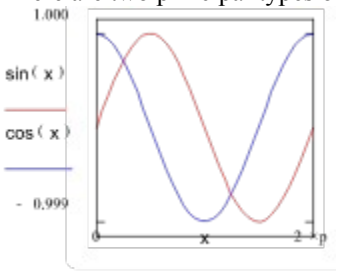
hide

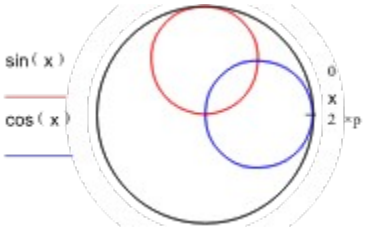
See Also

Expressions marked as display value (those with a red equal sign) will automatically display the expressions value when the program is run. The hide command can disable this behaviour.

Graphing

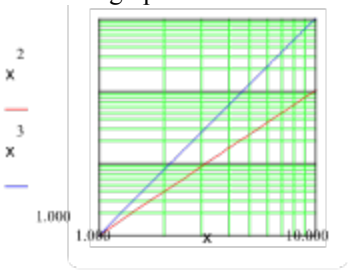
There are two principal types of graphs in the scratchpad, cartesian and polar.





cartesian polar

Cartesian graphs can be either log or normal.



log

Normal graphs can be easily converted to polar and vice versa. Log graphs cannot be converted to polar. Graphs can be either automatic or matrix.

The above graphs are all automatic, that is, the x axis variable automatically takes on values between its lower and upper limits and the y variables are calculated at each of these points and the results are then graphed (The number of points per inch can be set in the graph properties dialog) The scratchpad can also plot matrices that have been calculated previously. The y matrix and the x matrix should have the same number of points

The following is a simple matrix graph and its matloop construction.

yv := zmat(101 , 1)

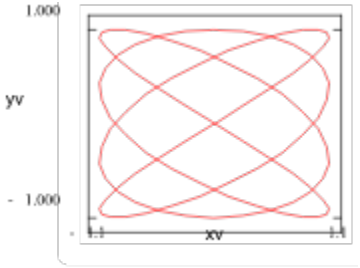
f1 := 4 f2 := 3 q := 0 $q := \frac{p}{2}$

MatLoop xv_i

$xv_i := \sin\left(2 \cdot p \cdot f1 \cdot \frac{i - 1}{100}\right)$

$yv_i := \sin\left(2 \cdot p \cdot f2 \cdot \frac{i - 1}{100} + q\right)$

EndMatLoop



The graph following is a combined automatic and matrix graph. The first trace **x squared vs x** is an automatic graph and the second trace **yv vs xv** is a matrix graph. yv and xv were calculated in the matloop above. Such combined graphs require that the autocalc variable (in this case x) be defined in the first trace (it can be used in other traces thereafter)

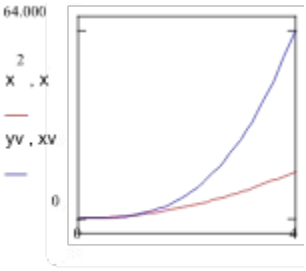
yv := zmat(41 , 1)

MatLoop xv_i

$xv_i := \frac{i - 1}{10.0}$

$yv_i := xv_i^3$

EndMatLoop



You can change a graphs properties by right clicking on an active graph expression, and choose the Graph Properties from the popup menu. A dialog box will appear that you can modify as desired. There is also a global graph properties dialog where you can set defaults. Its entries are self explanatory.

Graphs are sized with the Ctrl+arrow keys. You move between entry positions by using the up and down arrow keys. You can move the graph like any other object by click shift clicking to select the graph and then dragging. You can also align graphs like other objects by selecting two or more objects, then right clicking in one of them. Choose the desired alignment and the objects will align based on the object that was right clicked.