1 Contents

1 Contents 2
2 Introduction 21
3 Limitations 22
4 Operators table 23
5 Operators 24

- and : Logical AND ................................................. 24
- colon : Colon operator ............................................ 24
- ctranspose : Matrix conjugate transpose ...................... 24
- eq : Equal ............................................................ 25
- ge : Greater than or equal ...................................... 25
- gt : Greater than ................................................... 25
- indexing : Access submatrices ................................ 26
- ldivide : Inverse division ....................................... 26
- le : Less than or equal .......................................... 27
- lt : Less than ....................................................... 27
- mand : Logical AND ............................................. 27
- minus : Subtraction ............................................... 28
- mldivide : Matrix inverse division ......................... 28
- mnot : Logical NOT ............................................. 28
- mor : Logical OR .................................................. 29
- mpower : Matrix power ......................................... 29
- mdivide : Matrix division ....................................... 29
- mtimes : Matrix multiplication ............................... 30
- ne : Not equal ...................................................... 30
- not : Logical NOT ................................................ 30
- or : Logical OR ..................................................... 31
- plus : Addition ..................................................... 31
- power : Power ....................................................... 31
- ramp : Integer ramp .............................................. 32
- rdivide : Division ................................................ 32
• times : Multiplication .............................................. 32
• transpose : Matrix transpose ................................... 33
• uminus : Unary minus ............................................. 33
• uplus : Unary plus .................................................. 33

6 Functions 35
• abs : Absolute value ................................................. 35
• all : All non zero elements ....................................... 35
• angle : Angle .......................................................... 36
• any : Any non zero elements ..................................... 36
• automesh : Are inputs meshable ............................... 36
• cat : Concatenate matrices ...................................... 37
• ceil : Round upward ................................................ 37
• complex : Construct complex numbers ........................ 37
• computer : Platform .................................................. 38
• conj : Conjugate ...................................................... 38
• cplxpair : Pair complex numbers ............................... 38
• cross : Cross product ................................................. 39
• cumprod : Cumulative product .................................. 39
• cumsum : Cumulative sum ........................................ 40
• delaunay : Delaunay triangulation ............................. 40
• diff : Difference ....................................................... 41
• dot : Dot product ..................................................... 41
• dsearch : Find nearest point .................................... 42
• exp : Exponential ...................................................... 42
• eye : Identity matrix ................................................ 42
• find : Find non zero elements .................................... 43
• fix : Return integer part ............................................ 43
• floor : Round downward ............................................ 44
• flops : Flops counter ............................................... 44
• griddata : Fit to irregular data ................................... 44
• horzcat : Horizontal concatenation ............................ 45
• imag : Imaginary part ............................................... 45
• interp1 : 1-d interpolation ........................................ 45
• interp2 : 2-d interpolation ........................................ 46
• interp3 : 3-d interpolation ........................................ 46
• kron : Kronecker product ......................................... 47
• lasterr : Last error message ..................................... 47
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>linspace</td>
<td>Linearly-spaced vector</td>
<td>48</td>
</tr>
<tr>
<td>log</td>
<td>Natural logarithm</td>
<td>48</td>
</tr>
<tr>
<td>log10</td>
<td>base-10 logarithm</td>
<td>48</td>
</tr>
<tr>
<td>log2</td>
<td>base-2 logarithm</td>
<td>49</td>
</tr>
<tr>
<td>logical</td>
<td>Turn on logical flag</td>
<td>49</td>
</tr>
<tr>
<td>logspace</td>
<td>Log-spaced vector</td>
<td>49</td>
</tr>
<tr>
<td>mabs</td>
<td>Absolute value</td>
<td>50</td>
</tr>
<tr>
<td>magic</td>
<td>Magic square</td>
<td>50</td>
</tr>
<tr>
<td>many</td>
<td>Any non zero elements</td>
<td>50</td>
</tr>
<tr>
<td>max</td>
<td>Maximum</td>
<td>51</td>
</tr>
<tr>
<td>mcomplex</td>
<td>Construct complex numbers</td>
<td>51</td>
</tr>
<tr>
<td>mdouble</td>
<td>Convert to double</td>
<td>52</td>
</tr>
<tr>
<td>meshdom</td>
<td>Mesh grid for plotting</td>
<td>52</td>
</tr>
<tr>
<td>meshgrid</td>
<td>Mesh grid for plotting</td>
<td>52</td>
</tr>
<tr>
<td>mexp</td>
<td>Exponential</td>
<td>53</td>
</tr>
<tr>
<td>mfix</td>
<td>Return integer part</td>
<td>53</td>
</tr>
<tr>
<td>min</td>
<td>Minimum</td>
<td>53</td>
</tr>
<tr>
<td>mlog</td>
<td>Natural logarithm</td>
<td>54</td>
</tr>
<tr>
<td>mlogical</td>
<td>Turn on logical flag</td>
<td>54</td>
</tr>
<tr>
<td>mmax</td>
<td>Maximum</td>
<td>54</td>
</tr>
<tr>
<td>mmin</td>
<td>Minimum</td>
<td>55</td>
</tr>
<tr>
<td>mmod</td>
<td>Modulo</td>
<td>55</td>
</tr>
<tr>
<td>mod</td>
<td>Modulo</td>
<td>55</td>
</tr>
<tr>
<td>mrem</td>
<td>Reminder</td>
<td>56</td>
</tr>
<tr>
<td>msqrt</td>
<td>Square root</td>
<td>56</td>
</tr>
<tr>
<td>mxor</td>
<td>Logical XOR</td>
<td>56</td>
</tr>
<tr>
<td>ndgrid</td>
<td>N-dimensional mesh grid</td>
<td>57</td>
</tr>
<tr>
<td>nextpow2</td>
<td>Next power of two</td>
<td>57</td>
</tr>
<tr>
<td>pow2</td>
<td>Power of 2</td>
<td>57</td>
</tr>
<tr>
<td>prod</td>
<td>Product</td>
<td>58</td>
</tr>
<tr>
<td>quit</td>
<td>Abort program</td>
<td>58</td>
</tr>
<tr>
<td>real</td>
<td>Real part</td>
<td>58</td>
</tr>
<tr>
<td>rem</td>
<td>Reminder</td>
<td>59</td>
</tr>
<tr>
<td>round</td>
<td>Round</td>
<td>59</td>
</tr>
<tr>
<td>sign</td>
<td>Number sign</td>
<td>59</td>
</tr>
<tr>
<td>sort</td>
<td>Sort matrix</td>
<td>60</td>
</tr>
<tr>
<td>sortrows</td>
<td>Sort matrix rows</td>
<td>60</td>
</tr>
<tr>
<td>sound</td>
<td>Play sound</td>
<td>61</td>
</tr>
</tbody>
</table>
7 Linear algebra functions

- soundM: Play sound
- soundsc: Play scaled sound
- spline: Interpolation by cubic spline
- sqrt: Square root
- sum: Sum
- trapz: Approximate integral
- tsearch: Find enclosing triangle
- vertcat: Vertical concatenation
- wilkinson: Wilkinson matrix
- xor: Logical XOR

- axxbc: Solve Sylvester equation
- balance: Pre-eigenvalues computation
- cdf2rdf: Complex diagonal form to real diagonal form
- chol: Cholesky Factorization
- compan: Companion Matrix
- cond: Matrix condition number
-condeig: Matrix eigenvalue condition number
-det: Determinant
-diag: Matrix diagonal
-eig: Eigenvalues and eigenvectors
-eig_D: Eigenvalues
-eig_W: Eigenvectors
-expm: Matrix exponential
-givens: Givens plane rotation
-hess: Hessenberg form
-inv: Matrix Inverse
-logm: Matrix logarithm
- lu: LU factorization
- lu_L: L of LU decomposition
- lu_U: U of LU decomposition
-norm: Norm
-null: Null space
-orth: Orthonormal basis
-pinvv: Pseudo Inverse
-planerot: Plane rotation
-qr: QR factorization
• qrdelete : Update QR factorization ................................................. 75
• qrinsert : Update QR factorization ................................................. 75
• qz : QZ factorization .................................................................. 76
• qzval : Generalized eigenvalues ...................................................... 76
• rank : Matrix rank ...................................................................... 76
• rcond : Estimate for the reciprocal condition number ................... 77
• rref : Reduced row echelon form .................................................. 77
• rsf2csf : Real to complex Schur factorization ................................. 77
• schur : Schur factorization ............................................................ 78
• schur_T : T of Schur decomposition .............................................. 78
• schur_U : U of Schur decomposition .............................................. 78
• sqrtm : Matrix square root ............................................................. 79
• subspace : Angle between subspaces ............................................. 79
• svd : Singular values decomposition ............................................. 79
• svd_S : S of SVD decomposition .................................................... 80
• svd_U : U of SVD decomposition .................................................... 80
• svd_W : V of SVD decomposition .................................................... 81
• trace : Matrix trace ..................................................................... 81
• tril : Lower triangular matrix ....................................................... 81
• triu : Upper triangular matrix ....................................................... 82

8 Bit* functions .............................................................................. 83
• bitand : Bitwise AND ................................................................. 83
• bitcmp : Bitwise complement ....................................................... 83
• bitget : Bitwise get bit .................................................................. 83
• bitmax : Maximum representable integer ...................................... 84
• bitor : Bitwise OR ...................................................................... 84
• bitreset : Bitwise reset bit ............................................................ 84
• bitset : Bitwise set bit .................................................................. 85
• bitshift : Bitwise shift ................................................................. 85
• bitxor : Bitwise XOR ................................................................. 85
• mbitand : Bitwise AND ............................................................... 86
• mbitor : Bitwise OR ................................................................. 86

9 Control functions ........................................................................ 87
• ltifr : LTI frequency response ..................................................... 87
• ltitr : LTI time response .............................................................. 87
• ss2tf : State space to transfer function ....................................... 87
10 Set operations
- intersect : Sets intersection ........................................ 91
- ismember : Is member in set ........................................ 91
- union : Set union ...................................................... 91
- setdiff : Set difference ................................................ 92
- setxor : Set XOR ......................................................... 92
- union : Set union ...................................................... 92
- unique : Unique elements ............................................. 93

11 File and text I/O functions
- cd : Change directory .................................................. 94
- chdir : Change directory .............................................. 94
- copyfile : Copy file ..................................................... 94
- delete : Delete file ..................................................... 95
- deleteM : Delete file ................................................... 95
- dir : Show directory listing ......................................... 95
- dos : Run DOS command ............................................. 96
- error : Abort with error .............................................. 96
- exist : File existence .................................................. 96
- fclose : Close file handle .......................................... 97
- feof : Is end of file ................................................... 97
- ferror : File I/O error message .................................... 97
- fflush : Flush file ...................................................... 98
- fgetl : Get line from file ............................................ 98
- fgets : Get line from file ............................................ 98
- filesep : File separator .............................................. 99
- fopen : Open file handle ............................................ 99
- fprintf : Send formatted output to a file ....................... 100
- fread : Read from file ............................................... 101
- frewind : Move file pointer ......................................... 102
- fscanf : Get formatted input from file ........................... 102
- fseek : Seek in file .................................................. 102
• ftell : Tell file position ......................................................... 103
• fullfile : Filename from parts .................................................. 103
• fwrite : Write to file .............................................................. 104
• getenv : Get environment variable ............................................ 104
• home : Home cursor ............................................................... 105
• load : Load variables from file ................................................ 105
• menu : User menu ................................................................. 106
• mkdir : Create directory .......................................................... 106
• printf : Send formatted output to screen ................................. 106
• pwd : Get working directory .................................................... 107
• rmdir : Remove directory ......................................................... 107
• save : Save variables to file ..................................................... 107
• tempdir : Temporary directory ............................................... 108
• tempname : Temporary file name .............................................. 108
• vbload : Load variables from file ............................................. 108
• vbsave : Save variables to file ............................................... 109
• warning : Display a warning ................................................... 109
• wavread : Read wave file ....................................................... 109
• wavwrite : Write wave file ..................................................... 110

12 Function functions ................................................................. 111
• ode15s : Solve stiff ODE ........................................................ 111
• ode23 : Solve ODE with Runge-Kutta (2,3) ................................ 111
• ode23s : Solve stiff ODE ........................................................ 111
• ode45 : Solve ODE with Runge-Kutta (4,5) ................................ 112
• odeget : Get ode* options ........................................................ 112
• odeNN : Solve ordinary differential equation ............................ 112
• odeset : Set ode* options ....................................................... 113
• quad : Integrate f(x) ............................................................... 113
• quad8 : Integrate f(x) .............................................................. 114

13 Is* functions ................................................................. 115
• finite : Is finite ................................................................. 115
• is_scalar : Is scalar .............................................................. 115
• is_square : Is square matrix .................................................... 115
• is_struct : Is structure .......................................................... 116
• is_vector : Is vector ............................................................ 116
• iscell : Is cell matrix ............................................................ 116
• ischar : Is char matrix ....................................................... 117
• iscomplex : Is matrix complex ........................................ 117
• isempty : Is empty matrix ............................................... 117
• isequal : Are arguments equal ......................................... 118
• isfinite : Is finite ............................................................. 118
• isinf : Is infinite ............................................................. 118
• isletter : Is letter ............................................................. 119
• islogical : Is logical ........................................................ 119
• islower : Is lowercase ....................................................... 119
• isnan : Is Not-A-Number .................................................. 120
• isnumeric : Is matrix numeric .......................................... 120
• isprime : Is matrix prime ............................................... 120
• isreal : Is matrix real ...................................................... 121
• isspace : Is space ........................................................... 121
• isupper : Is uppercase ...................................................... 121

14 Optimization functions .................................................. 122
• conls : Constrained least squares .................................... 122
• constr : Constrained optimization ..................................... 122
• curvefit : Curve fitting .................................................... 123
• fmin : Find minimum of f(x) .............................................. 123
• fmins : Find minimum of f(x1,x2,...) ................................. 124
• fminu : Unconstrained minimization .................................. 124
• foptions : Set options to function functions ......................... 125
• fsolve : Solve non-linear equations .................................... 125
• fzero : Find zero of f(x) ................................................... 126
• leastsq : Non-linear least squares ..................................... 126
• lp : Linear programming .................................................. 127
• minimax : Minimax optimization ...................................... 127
• mnlx : Non-linear least squares ....................................... 128
• qp : Quadratic programming ............................................ 128

15 Polynomials ................................................................. 129
• mkpp : Make piecewise polynomial .................................... 129
• poly : Create polynomial ................................................ 129
• polyder : Derivative of polynomial ................................... 130
• polyfit : Fit polynomial for data series .............................. 130
• polyval : Polynomial value ............................................... 130
- polyvalm: Polynomial matrix value .................................................. 131
- ppval: Evaluate piecewise polynomial ............................................. 131
- roots: Polynomial roots ................................................................... 131
- unmkpp: Evaluate piecewise polynomial ........................................... 132

16 Signal processing functions ............................................................... 133
- conv: Convolution ............................................................................. 133
- conv2: 2-D Convolution ..................................................................... 133
- corrcov: Correlation Coefficients ....................................................... 134
- cov: Covariance matrix ...................................................................... 134
- deconv: Deconvolution ...................................................................... 134
- del2: Discrete Laplacian ..................................................................... 135
- dft: Discrete Fourier transform ........................................................... 135
- fft: Fourier transform ....................................................................... 136
- fft2: 2-Dimensional fourier transform .............................................. 136
- fftshift: Shift fft results ................................................................... 137
- filter: Digital FIR .............................................................................. 137
- filter2: Digital 2-dimensional FIR ..................................................... 137
- gradient: Gradient ............................................................................. 138
- ifft: Inverse Fourier transform ............................................................ 138
- ifft2: 2-Dimensional inverse Fourier transform ................................. 139
- ifftshift: Inverse fftshift ................................................................. 139
- sinc: Sinc function ............................................................................ 140
- unwrap: Smooth jumps ....................................................................... 140

17 Sparse functions ................................................................................. 141
- colmmd: Column permutation for sparsity ......................................... 141
- colperm: Column permutation for sparsity ......................................... 141
- full: Make matrix full ........................................................................ 141
- issparse: Is sparse ............................................................................. 142
- nnz: Number of non zero elements ................................................... 142
- nonzeros: Non-zero elements .............................................................. 142
- nzmax: Allocated number of non zero elements ................................. 143
- spalloc: Allocate sparse matrix ......................................................... 143
- sparse: Create sparse matrix from nonzero vectors .......................... 143
- spconvert: Create sparse matrix from columns ............................... 144
- speye: Sparse identity matrix ............................................................. 144
- spones: Sparse ones .......................................................................... 145
• sprand : Sparse random matrix .............................................. 145
• sprandn : Sparse random matrix ............................................. 145
• spzeros : Sparse zeros matrix ............................................. 146

18 Size and shape functions ................................................. 147
• columns : Number of columns .............................................. 147
• flipdim : Flip matrix at a dimension ...................................... 147
• fliplr : Flip matrix left-right .............................................. 147
• flipud : Flip matrix up-down .............................................. 148
• ind2sub : Linear index to subscript ....................................... 148
• ipermute : Inverse permute of matrix dimensions .................... 148
• length : Length of matrix .................................................. 149
• ndims : Number of dimensions ............................................ 149
• permute : Permute matrix dimensions ................................... 149
• repmat : Replicate matrix ................................................. 150
• reshape : Reshape matrix .................................................. 150
• rot90 : Rotate matrix ....................................................... 150
• rows : Number of rows ..................................................... 151
• size : Matrix dimensions .................................................. 151
• squeeze : Squeeze dimensions ............................................ 151
• sub2ind : Subscript to linear index ...................................... 152

19 Statistical and random functions .................................. 153
• erf : Error function .......................................................... 153
• erfc : Complementary error function ................................... 153
• erfcx : Scaled complementary error function ......................... 153
• erfinv : Inverse error function ........................................... 154
• mean : Average value ...................................................... 154
• median : Median value ..................................................... 155
• nchoosek : Combinations .................................................. 155
• perms : Permutations ........................................................ 155
• rand : Random numbers : uniform distribution ....................... 156
• randm : Random numbers : uniform distribution ................... 156
• randn : Random numbers : normal distribution ................. 157
• randperm : Random permutation ........................................ 157
• std : Standard deviation .................................................. 157
• stdM : Standard deviation ................................................. 158
20 Special functions

- airy : Airy function ........................................ 159
- bessel : Bessel function of the first kind .................. 159
- bessela : Bessel function of the first kind ................ 160
- besselh : Bessel functions .................................. 160
- besseli : Modified Bessel function of the first kind .......... 160
- besselj : Bessel function of the first kind .................. 161
- besselk : Modified Bessel function of the second kind ....... 161
- bessely : Bessel function of the second kind ............... 162
- beta : Beta function ........................................ 162
- betainc : Incomplete beta function ......................... 163
- betaln : Ln beta function ................................... 163
- cart2pol : Cartesian to Polar ................................ 164
- cart2sph : Cartesian to spherical .......................... 164
- ellipj : Jacobian elliptic functions ....................... 165
- ellipke : Complete elliptic integrals ....................... 165
- expint : Exponential integral ................................ 165
- factor : Factorize ........................................... 166
- gamma : Gamma function .................................... 166
- gammainc : Incomplete gamma function ................... 167
- gammainc : Ln of the gamma function ...................... 167
- gcd : Greatest common divisor .............................. 167
- lcm : Least common multiply ................................. 168
- pol2cart : Polar to Cartesian ................................ 168
- primes : Prime list .......................................... 169
- rat : Rational approximation ................................ 169
- sph2cart : Spherical to Cartesian .......................... 169

21 Special matrices

- cauchy : Cauchy matrix ....................................... 171
- hadamard : Hadamard matrix ................................ 171
- hankel : Hankel matrix ...................................... 172
- hilb : Hilbert matrix ......................................... 172
- invhilb : Inverse Hilbert matrix ......................... 172
- ones : Ones matrix ........................................... 173
- pascal : Pascal matrix ....................................... 173
- pascalM : Pascal matrix ..................................... 173
- rosser : Rosser matrix ...................................... 174
- **toeplitz**: Toeplitz matrix ................................................................. 174
- **vander**: Vandermonde matrix ............................................................. 174
- **zeros**: Zeros matrix ............................................................................. 175

### 22 String functions 176
- **base2dec**: Convert number base .......................................................... 176
- **bin2dec**: Convert number base ............................................................. 176
- **blanks**: Make blank string ................................................................... 176
- **cellstr**: Make cell array of strings ......................................................... 177
- **char**: Create string ................................................................................ 177
- **deblank**: Erase trailing blanks .............................................................. 177
- **dec2base**: Convert number base ............................................................ 178
- **dec2bin**: Convert to binary ..................................................................... 178
- **dec2hex**: Convert to hexadecimal .......................................................... 178
- **findstr**: Search for substring .................................................................. 179
- **hex2dec**: Convert number base .............................................................. 179
- **hex2num**: Construct IEEE-754 float ...................................................... 179
- **int2str**: Convert integer to string ........................................................... 180
- **lower**: Lower case .................................................................................. 180
- **mat2str**: Matrix to string ....................................................................... 180
- **mchar**: Convert to string ......................................................................... 181
- **num2str**: Number to string ...................................................................... 181
- **setstr**: Create string ............................................................................... 181
- **sprintf**: Send formatted output to variable ............................................. 182
- **sscanf**: Get formatted input from string ............................................... 182
- **str2double**: String to number ................................................................. 183
- **str2mat**: Strings to string matrix ........................................................... 183
- **str2num**: String to number ..................................................................... 183
- **strcat**: String horizontal concatenation ................................................ 184
- **strcmp**: Compare strings ........................................................................ 184
- **strcmpi**: Compare strings insensitive ................................................... 184
- **strjust**: Right justify a string ................................................................. 185
- **strmatch**: Match strings ......................................................................... 185
- **strncmp**: Partial compare strings ............................................................ 185
- **strncpyi**: Partial compare strings insensitive ......................................... 186
- **strrep**: Search and replace ..................................................................... 186
- **strtok**: Tokenize string .......................................................................... 186
- **strvcat**: String vertical concatenation ................................................... 187
23 Trigonometric functions

- acos : Inverse cosine ........................................... 188
- acosh : Inverse hyperbolic cosine .................................. 188
- acot : Inverse cotangent ........................................... 188
- acoth : Inverse hyperbolic cotangent ................................ 189
- acsc : Inverse cosecant ............................................ 189
- acsch : Inverse hyperbolic cosecant ................................ 190
- asec : Inverse secant ................................................ 190
- asech : Inverse hyperbolic secant .................................. 190
- asin : Inverse sine ................................................... 191
- asinh : Inverse hyperbolic sine ..................................... 191
- atan : Inverse tangent ............................................... 192
- atan2 : Inverse tangent .............................................. 192
- atanh : Inverse hyperbolic tangent .................................. 192
- cos : Cosine ......................................................... 193
- cosh : Hyperbolic cosine ............................................ 193
- cot : Cotangent ....................................................... 194
- coth : Hyperbolic cotangent ........................................ 194
- csc : Cosecant ....................................................... 194
- csch : Hyperbolic cosecant ......................................... 195
- mcos : Cosine ....................................................... 195
- msin : Sine .......................................................... 196
- mtan : Tangent ....................................................... 196
- sec : Secant .......................................................... 196
- sech : Hyperbolic secant ............................................. 197
- sin : Sine ............................................................ 197
- sinh : Hyperbolic sine ............................................... 197
- tan : Tangent ......................................................... 198
- tanh : Hyperbolic tangent ............................................ 198

24 Time and date functions

- clock : Time clock .................................................. 199
- date : Current date .................................................. 199
- datenum : Serial date ................................................ 199
- datevec : Decompose Serial date .................................... 200
- eomday : Last day of month ........................................ 200
- etime: Elapsed time ........................................ 200
- mclock: Time clock ........................................ 201
- now: Current time ......................................... 201
- tic: Start stopwatch ....................................... 201
- toc: Stop stopwatch ...................................... 202
- weekday: Weekday from serial date ..................... 202

25 Two-dimensional plots ................................... 203
- area: Area plot ........................................... 203
- bar: Vertical bar plot .................................... 203
- barh: Horizontal bar plot .............................. 204
- compass: Compass plot .................................. 204
- contour: Plot contours .................................. 204
- contourf: Filled contour plot ......................... 205
- errorbar: Plot with error bars ......................... 205
- feather: Feather plot .................................... 205
- hist: Plot Histogram ...................................... 206
- loglog: Log-log plot ..................................... 206
- pie: Pie chart plot .................................... 207
- plot: Linear plot ........................................ 208
- plotyy: Plot with labels ................................ 209
- polar: Polar plots ....................................... 209
- quiver: Quiver plot ..................................... 209
- ribbon: Ribbon plot ...................................... 210
- rose: Rose plot ......................................... 210
- scatter: Scatter X-Y markers plot .................... 210
- semilogx: Log-linear plot ............................. 211
- semilogy: Linear-log plot .............................. 211
- spy: See sparsity structure ............................ 211
- stairs: Stairs plot ...................................... 212
- stem: Stem plot ......................................... 212

26 Three-dimensional plots .................................. 213
- bar3: 3D vertical bar plot ............................. 213
- bar3h: 3D horizontal bar plot ......................... 213
- contour3: Plot 3-dimensional contours ............... 213
- mesh: Draw 3-dimensional mesh ...................... 214
- meshc: Draw mesh and contour ......................... 214
• pcolor : Image plot of surface levels ........................................ 215
• pie3 : 3D pie chart plot ......................................................... 215
• plot3 : 3-dimensional plot .................................................... 215
• scatter3 : 3D plot of scatter X-Y points .................................. 216
• surf : 3D filled surface plot .................................................. 216
• surfc : Surface plot with contour below it ................................. 217
• surfl : 3D filled surface plot with light .................................... 217

27 Plot customization ................................................................. 218
• axis : Modify axis ranges ....................................................... 218
• clf : Clear figure ................................................................. 218
• close : Close window ......................................................... 218
• figure : Switch or create new window ...................................... 219
• grid : Toggle grid lines on axes ............................................ 222
• hold : Set hold state ............................................................ 223
• legend : Add legend to plot ................................................ 223
• print : Print figure .............................................................. 224
• subplot : Create several plots ............................................. 224
• text : Place text on plot ....................................................... 224
• title : Add plot title ............................................................. 227
• xlabel : Add label to the x axis ............................................ 227
• ylabel : Add label to the y axis ............................................ 227
• zlabel : Add label to the z axis ............................................ 228
• zoom : Zoom into plot ........................................................ 228

28 Color functions ................................................................. 229
• autumn : Autumn colormap .................................................. 229
• bone : Bone colormap ........................................................ 229
• brighten : Brightens the current colormap ............................... 229
• caxis : Set color axis values for plots and images ...................... 230
• colorbar : Display colormap values ...................................... 230
• colordef : Set figure default colors to black or white ................ 231
• colormap : Set and get the colormap .................................... 231
• contrast : Enhance contrast of image .................................... 231
• cool : Cool colormap .......................................................... 232
• copper : Copper colormap ................................................... 232
• flag : Flag colormap .......................................................... 232
• gray : Gray colormap .......................................................... 233
• grey : Grey colormap ........................................... 233
• hot : Hot colormap ........................................... 233
• hsv : HSV colormap ......................................... 234
• jet : Jet colormap ........................................... 234
• lines : Colormap of default plot line colors .......... 234
• pink : Pink colormap ....................................... 235
• prism : Prism colormap .................................... 235
• rgbplot : Colormap plot .................................... 235
• shading : Set shading for surface plot ................. 236
• spring : Spring colormap ................................... 236
• summer : Summer colormap ................................ 236
• white : White colormap .................................... 237
• whitebg : Set figure background ......................... 237
• winter : Winter colormap ................................... 237

29 Imaging ......................................................... 239
• bmpread : Read bitmap image from file ................. 239
• bmpwrite : Write image to BMP file .................... 239
• getimage : Get image from graphic object ............. 239
• gifread : Read image from GIF file ..................... 240
• gifwrite : Write image to GIF file ....................... 240
• im2double : Convert image to double ................... 240
• image : Display bitmap image ............................. 241
• imagesc : Create scaled image object ................. 241
• imread : Read image and colormap from file ........ 242
• imshow : Show true-color or indexed image ........... 242
• imshg : Show true-color or indexed image ............. 242
• imwrite : Write image and colormap to file .......... 243
• pcxread : Read image from PCX file ................... 243
• pcxwrite : Write image to PCX file ..................... 243
• subimage : Prepare image in multi-images figure .... 244
• tifread : Read image from TIFF file ................... 244
• tiffwrite : Write image to TIFF file .................... 244
• truesize : Rescale figure to true size ................... 245

30 User-interface dialogs .......................................... 246
• dialog : Generic dialog ..................................... 246
• errordlg : Error dialog .................................... 246
• helpdlg : Help dialog ....................................... 246
31 Handle graphics functions

- allchild : Child list .................................................. 253
- axes : Create a axes for subsequent plots ....................... 253
- axesposition : Axes position ........................................ 256
- box : Make box around plot ........................................ 257
- capture : Capture figure area ....................................... 257
- cla : Clear axes .......................................................... 257
- clabel : Add contour heights labels ................................ 258
- clg : Clear figure .......................................................... 258
- clruprop : Clear user property ........................................ 258
- datetick : Label ticks with dates .................................... 259
- delete : Delete graphics object ...................................... 259
- deleteobj : Delete graphics object ................................... 259
- drawnow : Force graphics update .................................... 260
- figure_close : Close window .......................................... 260
- fill : Filled polygons with color .................................... 260
- findall : Find all graphics objects .................................. 261
- findfigs : Find hidden figures ........................................ 261
- findobj : Find graphics object ........................................ 261
- gca : Get current axes ................................................ 262
- gcbf : Get callback originating figure ............................. 262
- gcbo : Get callback originating object ............................. 262
- gcf : Get current figure ................................................ 263
- gco : Get current object ................................................. 263
- get : Get object property ........................................... 263
- getframe : Capture figure area ..................................... 264
- getpts : Visual selection of points ................................. 264
- getuprop : Get user property ...................................... 264
- ginput : Select axes point using mouse .......................... 265
- gtext : Place text on axes using mouse ......................... 265
- hwcontrol : Create a hardware/communication object ........ 266
- ishandle : Check if a valid handle ............................... 267
- ishold : Get hold state ............................................ 268
- line : Create low level line plot ................................ 268
- mget : Get object property ....................................... 271
- mline : Create low level line plot .............................. 271
- mset : Set object property ....................................... 272
- newplot : Prepare axes for additional plot .................... 272
- orient : Orientation of paper for printing ...................... 272
- patch : Create low level polygon plot .......................... 273
- peaks : Generate data for 3-D plots ............................. 276
- popupstr : Label string of menu item ........................... 276
- rectangle : Create low level rectangle object ................. 276
- refresh : Refresh graphics on figure ............................ 279
- reset : Reset graphics object .................................... 279
- root : Graphics root object ...................................... 280
- rotate3d : 3D rotation of plot .................................. 281
- set : Set object property ....................................... 281
- setstatus : Set UI string ......................................... 281
- setuprop : Set user property ..................................... 282
- shg : Show graph .................................................. 282
- surface : Create low level surface plot ......................... 282
- textwrap : Wrap text to fit in object ............................ 285
- uicontextmenu : Create a context sensitive menu ............ 286
- uicontrol : Create a user interface object ...................... 287
- uimenu : Create a menu or submenu ............................. 290
- uiresume : Continue after uiwait ............................... 291
- uiwait : Pause until uiresume .................................... 291
- umtoggle : Toggles state of the menu item .................... 292
- vbaxes : VB axes ............................................... 292
- vbrefresh : Refresh VB figure .................................... 292
- view : Change viewpoint ......................................... 293
- waitfor : Wait for an object to change ............................................. 293
- waitforbuttonpress : Wait for keyboard or mouse click .............. 293
- watchoff : Set mouse pointer to arrow .......................................... 294
- watchon : Set mouse pointer to hourglass ........................................ 294
2 Introduction

This Reference Guide describes the functions available for programming using MatrixVB.

Further documentation can be found in the User’s Guide or from our web site http://www.matrixvb.com/.
3 Limitations

Some functions are not yet available in MatrixVB. These include the ODE solvers, quad function integrators, functions optimization functions and graphics input functions:

- ODE*
- quad*
- constr
- curvefit
- fmin
- fmins
- fminu
- fsolve
- fzero
- leastsq
- minimax
- getpts
- ginput
- gtext
- uicontextmenu
- uicontrol
- uimenu
- uiresume
- uiwait
- umtoggle
- waitforbuttonpress

We hope to provide these functions in a future version of MatrixVB. In addition, the functions load and save were replaced by vbload and vbsave.
## 4 Operators table

In the table below, common mathematical and logical operators are presented, along with their MatrixVB equivalent function names. Documentation for an operator is found at the corresponding function entry.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>a+b</td>
<td>plus(a,b)</td>
</tr>
<tr>
<td>a-b</td>
<td>minus(a,b)</td>
</tr>
<tr>
<td>a.*b</td>
<td>times(a,b)</td>
</tr>
<tr>
<td>a^b</td>
<td>mtimes(a,b)</td>
</tr>
<tr>
<td>a./b</td>
<td>rdivide(a,b)</td>
</tr>
<tr>
<td>a\b</td>
<td>ldivide(a,b)</td>
</tr>
<tr>
<td>a/b</td>
<td>mrdivide(a,b)</td>
</tr>
<tr>
<td>a\b</td>
<td>mldivide(a,b)</td>
</tr>
<tr>
<td>a.^ b</td>
<td>power(a,b)</td>
</tr>
<tr>
<td>a^ b</td>
<td>mpower(a,b)</td>
</tr>
<tr>
<td>a'</td>
<td>ctranspose(a)</td>
</tr>
<tr>
<td>a.'</td>
<td>transpose(a)</td>
</tr>
<tr>
<td>a:b:c</td>
<td>colon(a,b,c)</td>
</tr>
<tr>
<td>a&lt;b</td>
<td>lt(a,b)</td>
</tr>
<tr>
<td>a&lt;=b</td>
<td>le(a,b)</td>
</tr>
<tr>
<td>a&gt;b</td>
<td>gt(a,b)</td>
</tr>
<tr>
<td>a&gt;=b</td>
<td>ge(a,b)</td>
</tr>
<tr>
<td>a==b</td>
<td>eq(a,b)</td>
</tr>
<tr>
<td>a~=b</td>
<td>ne(a,b)</td>
</tr>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>a&amp;b</td>
<td>and(a,b)</td>
</tr>
<tr>
<td>~a</td>
<td>not(a)</td>
</tr>
<tr>
<td>√a</td>
<td>msqrt(a)</td>
</tr>
<tr>
<td>[a b]</td>
<td>horzcat(a,b)</td>
</tr>
<tr>
<td>[a;b]</td>
<td>vertcat(a,b)</td>
</tr>
</tbody>
</table>
5 Operators

• **and** ............................................ Logical AND

  **Description**

  \[ z = \text{and}(x, y) \]

  \[ z = x \& y \]

  and is named ```mand``` in Visual Basic programs.

  **See Also**

  all, any, logical, mand, not, or, xor

• **colon** ....................................... Colon operator

  **Description**

  \[ y = \text{colon}(j, i, k) \]

  return a row vector ranging from \( j \) to \( k \) with \( i \) increments. The last value is never larger than \( k \), but may be smaller if \( k - j \) is not divisible by \( i \).

  **See Also**

  find, indexing, linspace, logspace, ramp

• **ctranspose** ........................... Matrix conjugate transpose

  **Description**

  \[ y = \text{ctranspose}(x) \]

  is the matrix conjugate transpose, \( x' \).

  **See Also**

  flipdim, rot90, transpose
• **eq** .........................................................Equal

Description

\[ z = \text{eq}(x, y) \] is 1 if \( x == y \) and 0 otherwise.

See Also

all, any, ge, gt, isequal, le, logical, lt, ne

• **ge** .......................................................Greater than or equal

Description

\[ z = \text{ge}(x, y) \] is 1 if \( x \geq y \) and 0 otherwise.

See Also

all, any, eq, gt, le, logical, lt, ne

• **gt** .........................................................Greater than

Description

\[ z = \text{gt}(x, y) \] is 1 if \( x > y \) and 0 otherwise.

See Also

all, any, eq, ge, le, logical, lt, ne
indexing .................................. Access submatrices

Description

\[ y = x(v) \] returns \( x \) indexed by \( v \).

\[ x(v) = y \] sets \( x \) values, indexed by \( v \) to be \( y \) values. If \( y \) is an empty matrix, the indexed elements are deleted from the matrix \( x \).

Indexing can be in more than one dimension, for instance \( x(v,w) \). In this case, \( v \) index the rows and \( w \) index the columns.

Indices may be of three types:

**regular**

Regular index contains numbers, the indices into the indexed matrix.

**logical**

Logical matrices are the result of functions giving true-false results, such as \(<,>,\text{all,any} \). A logical index is a mask on the indexed matrix, where all non-zeros in the logical index affect the corresponding elements in the indexed matrix.

**colon**

Indexing with : is same as selecting the whole dimension.

The \texttt{end} keyword in an indexing expression means the last index in that dimension.

See Also

braces, colon, logical, parentheses

ldivide  .................................. Inverse division

Description

\[ z = ldivide(x,y) \] returns the result of dividing \( y \) by \( x \) element-by-element, \( x \setminus y \).
See Also

minus, mrdivide, plus, rdivide, times

• le  ......................................................... Less than or equal

Description

z = le(x, y) is 1 if \( x \leq y \) and 0 otherwise.

See Also

all, any, eq, ge, gt, islogical, logical, lt, ne

• lt  ......................................................... Less than

Description

z = lt(x, y) is 1 if \( x < y \) and 0 otherwise.

See Also

all, any, eq, ge, gt, le, logical, ne

• mand  ................................................. Logical AND

Description

and is named mand in Visual Basic programs.
See Also

and

• **minus** .................................Subtraction

Description

\[ z = \text{minus}(x, y) \] returns the result of subtracting \( y \) from \( x \), \( x - y \).

See Also

ldivide, plus, rdivide, times

• **mldivide** .............................Matrix inverse division

Description

\[ z = \text{mldivide}(x, y) \] returns the result of matrix division of \( y \) by \( x \), \( x \backslash y \).

See Also

colmmd, colperm, inv, lu, mldivide, mtimes, qr, rem, times

• **mnot** .................................Logical NOT

Description

\text{not} is named \text{mnot} in Visual Basic programs.
See Also
not

• **mor** ........................ Logical OR

Description
or is named mor in Visual Basic programs.

See Also
or

• **mpower** ........................ Matrix power

Description
\( z = \text{mpower}(x,y) \) is \( x \) raised to the \( y \)-th power, \( x^y \).

See Also
log,power,powM

• **mrdive** ........................ Matrix division

Description
\( z = \text{mrdive}(x,y) \) returns the result of matrix division of \( x \) by \( y \), \( x/y \).
See Also
det, inv, ldivide, lu, mldivide, mtimes, rank, times

- **mtimes** .......................... Matrix multiplication
  **Description**
  \[ z = \text{mtimes}(x, y) \] returns the result of matrix multiplication of \( x \) and \( y \), \( x \times y \).

  **See Also**
cross, dot, inv, kron, mldivide, mrdivide, sqrtm, times

- **ne** ........................................ Not equal
  **Description**
  \[ z = \text{ne}(x, y) \] is 1 if \( x = y \) and 0 otherwise.

  **See Also**
all, any, eq, ge, gt, le, logical, lt

- **not** ................................. Logical NOT
  **Description**
  \[ y = \text{not}(x) \] returns the logical NOT of \( x \).
  \text{not} is named \text{mnot} in Visual Basic programs.
See Also
and, logical, mnot, or, xor

- or .............................................. Logical OR

Description
\[
z = \text{or}(x, y) \text{ returns } x \lor y.
\]

or is named mor in Visual Basic programs.

See Also
all, and, any, logical, mor, not, xor

- plus ............................................. Addition

Description
\[
z = \text{plus}(x, y) \text{ returns the result of adding } x \text{ and } y, x + y.
\]

See Also
ldivide, minus, rdivide, times, uminus, uplus

- power ........................................ Power

Description
\[
z = \text{power}(x, y) \text{ is } x \text{ raised to the } y\text{-th power, element by element, } x^y.
\]
See Also
log, log10, mpower, pow2, realpow

• ramp  ........................................... Integer ramp

Description
y = ramp(j, k) return a row vector ranging from j to k with unit steps. ramp(j, k) is identical to colon(j, 1, k).

See Also
colon, linspace, logspace

• rdivide  ........................................... Division

Description
z = rdivide(x, y) returns the result of dividing x by y element-by-element, x./y.

See Also
ldivide, minus, plus, times

• times  .......................................... Multiplication

Description
z = times(x, y) returns the result of multiplying x by y element-by-element, x.*y.
See Also

ctranspose, flipdim, repmat, rot90

• transpose .................. Matrix transpose

Description

$y = \text{transpose}(x)$ is the matrix transpose, $x.'$.

See Also

ctranspose, flipdim, repmat, rot90

• uminus .................. Unary minus

Description

$y = \text{uminus}(x)$ returns $-x$.

See Also

plus, uplus

• uplus .................. Unary plus

Description

$y = \text{uplus}(x)$ returns $+x$. 
See Also

plus, uminus
6 Functions

• abs ................................. Absolute value

  Description
  abs(x) is the absolute value of x, |x|. If x is complex, the return value is computed according to
  \[ \text{abs}(x) = \sqrt{\text{real}(x)^2 + \text{imag}(x)^2}. \]
  without overflow.

  abs is named mabs in Visual Basic programs.

  See Also
  angle,conj,cplxpair,imag,mabs,polar,polyfit,real,setstr,sign

• all ................................. All non zero elements

  Description
  y=all(x) returns logical 1 if all x elements along a dimension are non-zeros, and logical 0 otherwise.
  The dimension can be explicitly given as an optional argument, after all other input arguments, as in y=all(...,dimension). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

  See Also
  and,any,eq,ge,gt,isequal,le,logical,lt,ne,or
• **angle** ................................. Angle

   **Description**

   \( y = \text{angle}(x) \) is the phase angle of \( x \),

   \[ angle(x) = \text{atan2}(\text{imag}(x), \text{real}(x)). \]

   If \( x \) is real, the result is 0 for non-negative \( x \) and \( \pi \) otherwise.

   **See Also**

   abs, cart2pol, cart2sph, conj, imag, pol2cart, polar, real, sph2cart, unwrap

• **any** ................................. Any non zero elements

   **Description**

   \( y = \text{any}(x) \) returns logical 1 if any of the elements along a dimension is non-zero, and logical 0 otherwise.

   The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{any}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

   any is named many in Visual Basic programs.

   **See Also**

   all, and, eq, ge, gt, isequal, islogical, le, logical, lt, many, ne, or

• **automesh** ............................. Are inputs meshable

   **Description**

   \( a = \text{automesh}(x, y) \) or \( a = \text{automesh}(x, y, z) \) returns 1 if the input variables should be auto-meshed.
See Also
mesh

• **cat** .......................... Concatenate matrices

Description

\[ z=\text{cat}(x,y,\text{dim}) \] returns the matrices \( x \) and \( y \) concatenated along dimension \( \text{dim} \).

See Also

brackets, horzcat, repmat, rot90, size, vertcat

• **ceil** ................................. Round upward

Description

\[ y=\text{ceil}(x) \] returns an integer such that \( y - 1 < x \leq y \).

See Also

fix, floor, round

• **complex** .......................... Construct complex numbers

Description

\[ z=\text{complex}(a,b) \] returns \( a+ib \) where \( i=\sqrt{-1} \).

\text{complex} \text{ is named } \text{mcomplex} \text{ in Visual Basic programs.}
See Also
imag, mcomplex, real

• **computer** ................................. Platform

Description

\( y = \text{computer} \) returns the platform on which the program currently runs. This information can be useful to access platform-specific features.

\([y, m] = \text{computer} \) also returns the maximal matrix size in \( m \).

See Also

gteenv, isieee, isunix, version

• **conj** ............................. Conjugate

Description

\( y = \text{conj}(x) \) returns the complex conjugate of \( x \).

See Also

abs, angle, cplxpair, imag, isreal, polar, real

• **cplxpair** ........................... Pair complex numbers

Description

\( y = \text{cplxpair}(x, \text{tolerance}) \) sorts the vector \( x \) so that all complex conjugate pairs appear first, followed by the real values. \text{tolerance} set the threshold for deciding if a number is real or
complex. If \( x \) is a matrix, \texttt{cplxpair} works along its columns.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y=cplxpair(\ldots,\text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

**See Also**

\texttt{abs,conj}

• **cross** ................................. Cross product

**Description**

\( z=\text{cross}(x,y) \) is the cross product of the vectors \( x \) and \( y \), defined as

\[
\begin{align*}
    z(1) &= x(2)y(3) - x(3)y(2) \\
    z(2) &= x(3)y(1) - x(1)y(3) \\
    z(3) &= x(1)y(2) - x(2)y(1)
\end{align*}
\]

**See Also**

\texttt{dot,mtimes,subspace,sum}

• **cumprod** ................................. Cumulative product

**Description**

\( y=\text{cumprod}(x) \) is the cumulative product of \( x \), computed along a dimension.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y=\text{cumprod}(\ldots,\text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.
See Also

cumsum, prod

• **cumsum** .............................. Cumulative sum

Description

\[ y = \text{cumsum}(x) \]

is the cumulative sum, computed along a dimension of the matrix \( x \).

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{cumsum}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

See Also

cumprod, diff, sum, trapz

• **delaunay** ............................... Delaunay triangulation

Description

\[ z = \text{delaunay}(x, y) \]

is a \( m \) by 3 matrix describing the triangle connections between the \((x, y)\) data points, where there are \( m \) points. An optional third argument, \('\text{sorted}'\), specifies that the \((x, y)\) data points are sorted and unique. The argument causes \text{delaunay} to skip the sorting step.

See Also

dsearch, griddata, tsearch
• **diff**  .................................................... Difference

**Description**

\[ z = \text{diff}(x, n) \]

returns the difference between adjacent elements of \( x \), computed \( n \) times. On each iteration, \( x \) is shortened by one element along a dimension.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{diff}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

**See Also**

cumsum,sum,unwrap

• **dot**  ....................................................... Dot product

**Description**

\[ z = \text{dot}(x, y) \]

is the dot product of the vectors \( x \) and \( y \), defined as

\[ \text{dot}(x, y) = x^T \cdot y \]

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{dot}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

**See Also**

cross,mtimes,subspace,sum
• **dsearch** ................................. Find nearest point

**Description**

\( n = dsearch(x, y, \text{tri}, xi, yi) \) returns the index \( n \) of the point \((x(n), y(n))\), which is nearest to the given point \((xi, yi)\). The parameter \( \text{tri} \) is obtained from calling \text{delaunay}.

**See Also**

delaunay, tsearch

• **exp** ................................. Exponential

**Description**

\( y = \exp(x) \) returns the exponential of \( x \),

\[
\exp(x) = e^x
\]

where \( e = 2.71828... \).

If \( x \) is complex,

\[
\exp(x) = e^{\text{real}(x)} \cdot (\cos(\text{imag}(x)) + i \cdot \sin(\text{imag}(x)))
\]

\( \exp \) is named \text{mexp} in Visual Basic programs.

**See Also**

expm, log, log10, mexp

• **eye** ................................. Identity matrix

**Description**

\( y = \text{eye}(m, n) \) is a \( m \)-by-\( n \) identity matrix, composed of all zeros, except on the main diagonal where it is 1.

Also, \( \text{eye}(n) = \text{eye}(n, n) \) and \( \text{eye}([m \ n]) = \text{eye}(m, n) \).
See Also
ones, size, zeros

• find  ......................... Find non zero elements

Description
y = find(x) returns the linear indices of the non-zero elements in x.
[i, j, v] = find(x) returns the row and column indices of the non-zero elements in x, in i and j respectively, and the non-zero elements in v. The output variable v may be omitted.

See Also
colon, findstr, logical, nonzeros, sort, strrep, sub2ind

• fix  ........................ Return integer part

Description
y = fix(x) returns the integer part of x.
fix is named mfix in Visual Basic programs.

See Also
ceil, floor, mfix, round
• floor .......................... Round downward
  Description
  y=floor(x) returns an integer such that $y - 1 \leq x < y$.

  See Also
  ceil, fix, round

• flops .......................... Flops counter
  Description
  f=flops always return 0.

  See Also
  cputime

• griddata ............................ Fit to irregular data
  Description
  [XI,YI,ZI]=griddata(x,y,z,xi,yi,method) interpolates the points described by (xi, yi) pairs to the surface (x, y, z). The interpolated surface is returned in (XI, YI, ZI). If just one output argument is given, ZI is returned. Unlike interp2, x and y need not be on a regular grid. method selects the interpolation method, out of the possible 'linear' (default), 'nearest' or 'invdist'.

  See Also
  delaunay, interp1, interp2, interp3, spline
• **horzcat** .......................... Horizontal concatenation

**Description**

\( z = \text{horzcat}(x, y) \) returns the horizontal concatenation of \( x \) and \( y \), \([x \ y]\).

**See Also**

brackets, cat, repmat, rot90, str2mat, strcat, strvcat, vertcat

• **imag** ............................... Imaginary part

**Description**

\( y = \text{imag}(x) \) is the imaginary part of the number \( x \). If \( x \) is real, \( y = 0 \).

**See Also**

abs, angle, conj, i, iscomplex, isreal, j, polar, real, sign

• **interp1** ............................. 1-d interpolation

**Description**

\( y_{\text{int}} = \text{interp1}(x, y, x_{\text{int}}) \) returns an interpolation of \( x_{\text{int}} \) values between \((x, y)\) points.

The input arguments may be followed by an optional **method**, which determines the interpolation method used.

The methods are

'\text{linear}'  Linear interpolation (default).
'\text{nearest}'  Nearest point interpolation.
'spline'  Spline interpolation, same as the **spline** command.
See Also
griddata, interp2, interp3, spline

- **interp2** .......................... 2-d interpolation

Description

\[ \text{zint} = \text{interp2}(x, y, z, xint, yint) \]
returns an interpolation of \( z \) values between \((x, y, z)\) triplets.

\[ \text{zint} = \text{interp2}(z, N) \]
interpolates \( z \) to be \( 2^N \) times as dense as \( z \).

The input arguments may be followed by an optional method, which determines the interpolation method used.

The methods are

'linear' Linear interpolation (default).

'nearst' Nearest point interpolation.

'spline' Spline interpolation, same as the \text{spline} command.

See Also
griddata, interp1, interp3, spline

- **interp3** .......................... 3-d interpolation

Description

\[ \text{vint} = \text{interp3}(x, y, z, v, xint, yint, zint) \]
returns an interpolation of \( v \) values between \((x, y, z, w)\) points.

\[ \text{vint} = \text{interp3}(v, N) \]
interpolates \( v \) to be \( 2^N \) times as dense as \( v \).

The input arguments may be followed by an optional method, which determines the interpolation method used.

The methods are
'linear'  Linear interpolation (default).
'nearest' Nearest point interpolation.
'spline'  Spline interpolation, same as the spline command.

See Also
griddata, interp1, interp2, spline

• kron  Kronecker product

Description
z=kron(x,y) is the Kronecker product of x and y.

See Also
mtimes, prod

• lasterr  Last error message

Description
lasterr is the last error message displayed.
lasterr(‘’) clears the last error message.

See Also
error
• **linspace** .......................... Linearly-spaced vector

**Description**

\[ z = \text{linspace}(x, y, n) \]

returns a \( n \)-point linearly-spaced vector with values ranging from \( x \) to \( y \). If \( n \) is omitted, 100 points are returned.

**See Also**
colon, logspace, ramp

• **log** ................................. Natural logarithm

**Description**

\[ y = \log(x) \]

is the natural logarithm of \( x \) such that \( \exp(\log(x)) = x \) for positive \( x \). If \( x \) is non-positive, \( y \) is complex.

\log\ is named \text{mlog} in Visual Basic programs.

**See Also**
exp, log10, log2, logm, mlog, mpower, power, reallog

• **log10** ............................... base-10 logarithm

**Description**

\[ y = \log_{10}(x) \]

is the logarithm to base 10 of \( x \) such that \( 10^{\log_{10}(x)} = x \) for positive \( x \). If \( x \) is non-positive, \( y \) is complex.

**See Also**
exp, log, log2, power
• log2 ........................................... base-2 logarithm

**Description**

\( y = \log_2(x) \) is the logarithm to base 2 of \( x \).

\([f,e] = \log_2(x)\) returns the fraction and exponent of the floating point number \( x \) in \( f \) and \( e \), respectively.

**See Also**

log, log10, pow2

• logical ........................................... Turn on logical flag

**Description**

\( y = \text{logical}(x) \) returns the matrix \( x \), with the logical flag turned on. The result can be used for logical indexing.

\( \text{logical} \) is named \text{mlogical} in Visual Basic programs.

**See Also**

all, and, any, eq, find, ge, gt, indexing, islogical, le, lt, ne, not, or, xor

• logspace ........................................... Log-spaced vector

**Description**

\( z = \text{logspace}(x, y, n) \) returns a \( n \)-point log-spaced vector with values ranging from \( x \) to \( y \). If \( n \) is omitted, 50 points are returned.

If \( y = \pi \), the vector range between \( 10^x \) and \( \pi \).
See Also
colon,linspace,ramp

• **mabs** .......................... Absolute value
  Description
  abs is named mabs in Visual Basic programs.
  See Also
  abs

• **magic** .......................... Magic square
  Description
  y=magic(n) is a magic square of order n, composed of the integers 1 : n^2 with all column sums and row sums being equal.
  See Also
  pascal,sum

• **many** .......................... Any non zero elements
  Description
  any is named many in Visual Basic programs.
See Also

max

Description

$z = max(x)$ finds the maximal element value of $x$ along a dimension.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in $y = max(..., \text{dimension})$. If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

$z = max(x, y)$ returns the larger value of $x$ and $y$. Complex numbers are compared by their absolute value.

max is named mmax in Visual Basic programs.

See Also

min, mmax, mmin, sort, sum

mcomplex

Description

complex is named mcomplex in Visual Basic programs.

See Also

complex
• **mdouble** .......................... Convert to double  
Description  
double is named mdouble in Visual Basic programs.  

See Also  
double

• **meshdom** .......................... Mesh grid for plotting  
Description  
[xx,yy]=meshdom(x,y) returns same xx and up-down flipped yy as meshgrid does.  

See Also  
meshgrid

• **meshgrid** .......................... Mesh grid for plotting  
Description  
[xx,yy]=meshgrid(x,y) returns matrices xx and yy, composed of the rows of x and the columns of y, which can be used to plot two-dimensional functions instead of looping over x and y values. If y is omitted, y=x.  
Similarly, for three dimensions, [xx,yy,zz]=meshgrid(x,y,z)  

See Also  
mesh,meshc,meshdom,ndgrid,plot
• **mexp** ................................. Exponential

  **Description**

  exp is named mexp in Visual Basic programs.

  **See Also**

  exp

• **mfix** ................................. Return integer part

  **Description**

  fix is named mfix in Visual Basic programs.

  **See Also**

  fix

• **min** ................................. Minimum

  **Description**

  \( z = \text{min}(x) \) finds the minimal element of \( x \) value along a dimension.

  The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{min}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

  \( z = \text{min}(x, y) \) returns the smaller value of \( x \) and \( y \). Complex numbers are compared by the absolute value.

  min is named mmin in Visual Basic programs.
See Also
max, mmax, mmin, sort, sum

- **mlog** ................................. Natural logarithm
  Description
  log is named mlog in Visual Basic programs.

  See Also
  log

- **mlogical** ............................. Turn on logical flag
  Description
  logical is named mlogical in Visual Basic programs.

  See Also
  double

- **mmax** ..................................... Maximum
  Description
  max is named mmax in Visual Basic programs.
See Also
max, min

- **mmin** ................................. Minimum

Description

min is named mmin in Visual Basic programs.

See Also
max, min

- **mmod** ................................. Modulo

Description

mod is named mmod in Visual Basic programs.

See Also
mod

- **mod** ................................. Modulo

Description

\[ z = \text{mod}(x, y) \] returns the signed remainder of \( x \) divided by \( y \).

mod is named mmod in Visual Basic programs.
See Also
mod, rem

- mrem .................................................. Reminder

Description
rem is named mrem in Visual Basic programs.

See Also
rem

- msqrt ............................................. Square root

Description
sqrt is named msqrt in Visual Basic programs.

See Also
sqrt

- mxor ................................................ Logical XOR

Description
xor is named mxor in Visual Basic programs.
See Also
xor

- **ndgrid** .......................... N-dimensional mesh grid

Description

\[ [xx,yy]=\text{ndgrid}(x,y) \] or \[ [xx,yy,zz]=\text{ndgrid}(x,y,z) \]
returns matrices \( xx, yy \) and \( zz \), composed of \( x, y \) and \( z \), which can be used to evaluate two- or three- dimensional functions instead of looping.

See Also

meshgrid

- **nextpow2** .......................... Next power of two

Description

\( y=\text{nextpow2}(x) \) returns the next power of 2 which is not smaller than \( x \).

See Also

pow2

- **pow2** ................................ Power of 2

Description

\( z=\text{pow2}(x) \) is \( 2^x \).
\( z=\text{pow2}(f,e) \) is the floating point number whose fraction is \( f \) and exponent \( e \).
See Also
Inf, log2, nextpow2, power, realmax, realmin

• prod  ......................................................... Product

Description
y=prod(x) is the product of x, computed along a dimension of the matrix x. The dimension can be explicitly given as an optional argument, after all other input arguments, as in y=prod(...,dimension). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

See Also
cumprod, kron, sum

• quit  ......................................................... Abort program

Description
quit aborts the current program execution and exits the interpreter.

See Also
error, exit, pause, warning

• real  ......................................................... Real part

Description
y=real(x) is the real part of the number x. If x is real, y=x.
See Also
abs, angle, conj, i, imag, iscomplex, isreal, j, polar, sign

• rem  .................................................. Reminder

Description
z = rem(x, y) is the reminder of x divided by y.
rem is named mrem in Visual Basic programs.

See Also
mldivide, mod, mrem

• round  .................................................. Round

Description
y = round(x) returns the value of x, rounded to the nearest integer.

See Also
ceil, fix, floor

• sign  .................................................. Number sign

Description
y = sign(x) returns 1 for positive numbers, 0 for zero and -1 for negative numbers. If x is complex, sign(x) is a unit vector of the same direction as x.
See Also
abs, imag, real

• sort ................................. Sort matrix

Description
[y, index] = sort(x) return x sorted along a dimension. The optional index output variable indicates the new locations of the original elements.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in y = sort(..., dimension). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

Complex elements are sorted according to the absolute value.

See Also
find, max, min, sortrows

• sortrows ............................ Sort matrix rows

Description
[y, index] = sortrows(x) return the sorted rows of x.

See Also
sort
• **sound** ................................. Play sound

**Description**

`sound(y,f,b)` plays the vector `y` at sample rate `f` with `b` bits of precision. The default value for `f` is 8192 Hz and for `b` is 16 bits.

`sound` is named `soundM` in Visual Basic programs.

**See Also**

`image,soundM,soundsc,wavread,wavwrite`

• **soundM** ................................. Play sound

**Description**

`sound` is named `soundM` in Visual Basic programs.

**See Also**

`sound`

• **soundsc** ................................. Play scaled sound

**Description**

`soundsc` syntax is identical to `sound`, but the waveform is scaled to $[-1,1]$ before playing.

**See Also**

`sound`
• **spline**  .................. Interpolation by cubic spline

**Description**

\( \text{yint} = \text{spline}(x, y, xint) \) returns the cubic spline interpolation of \( xint \) values between \((x, y)\).

\( \text{pp} = \text{spline}(x, y) \) returns the pp-form of the interpolation polynomial without computing any interpolated points, which can be later computed using \( \text{ppval} \).

**See Also**

griddata, interp1, interp2, interp3, mkpp, ppval, unmkpp

• **sqrt**  .................................. Square root

**Description**

\( y = \sqrt{x} \) is the square root of \( x \). If \( x \) is complex or negative, \( y \) is complex.

\( \text{sqrt} \) is named \( \text{msqrt} \) in Visual Basic programs.

**See Also**

msqrt, realsqrt, sqrtm, times

• **sum**  .................................. Sum

**Description**

\( y = \text{sum}(x) \) is the sum of \( x \), computed along a dimension of the matrix \( x \).

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{sum}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.
See Also
cross,cumsum,diff,dot,magic,max,min,prod,trace,trapz

• trapz .......................... Approximate integral

Description

\[ z = \text{trapz}(x, y) \] is the approximate integral, computed using the trapezoidal method, of \( y \) integrated over \( x \). If \( x \) is omitted, the integration is over unit spacing.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{trapz}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

See Also
cumsum,sum

• tsearch ........................ Find enclosing triangle

Description

\[ n = \text{tsearch}(x, y, \text{tri}, xi, yi) \] returns the index \( n \) of the triangle \( \text{tri}(n) \), which encloses the given point \((xi, yi)\). If no such triangle was found, \text{tsearch} \ returns \text{NaN}. The parameter \text{tri} is obtained from calling \text{delaunay}.

See Also
delaunay,dsearch
• **vertcat** .......................... Vertical concatenation

**Description**

\[ z = \text{vertcat}(x, y) \] returns the vertical concatenation of \( x \) and \( y \), \([x;y] \).

**See Also**

brackets, cat, horzcat, repmat, rot90, strcat, strvcat

• **wilkinson** .......................... Wilkinson matrix

**Description**

\[ y = \text{wilkinson}(n) \] returns the Wilkinson matrix of order \( n \).

**See Also**

cauchy, eig, pascal

• **xor** .............................. Logical XOR

**Description**

\[ z = \text{xor}(x, y) \] returns the exclusive or of \( x \) and \( y \).

\( \text{xor} \) is named \text{mxor} in Visual Basic programs.

**See Also**

and, logical, mxor, not, or
7 Linear algebra functions

• \textit{axxbc}  \hspace{1cm} \textit{Solve sylvester equation}

\textbf{Description}

\texttt{x=axxbc(a,b,c)} returns a solution to the equation \texttt{ax+xb=c}.

\textbf{See Also}
lu,mldivide

• \textit{balance}  \hspace{1cm} \textit{Pre-eigenvalues computation}

\textbf{Description}

\texttt{y=balance(x)} returns a balanced version of the matrix \texttt{x}, whose eigenvalues can be computed more accurately by the \texttt{eig} function.

\textbf{See Also}
chol,condeig,eig,hess,qz

• \textit{cdf2rdf}  \hspace{1cm} \textit{Complex diagonal form to real diagonal form}

\textbf{Description}

\texttt{[v2,d2]=cdf2rdf(v,d)} converts any complex eigenvalues on \texttt{diag(d)} to real form so that

\[ x \cdot v2 = v2 \cdot d2. \]
See Also
eig

- chol ................................. Cholesky Factorization

Description
y=chol(x) returns the Cholesky factorization of x. An error occurs if x is not positive definite.  
[y,p]=chol(x) returns the Cholesky factorization of the largest submatrix x(1 : (p − 1),1 :  
(p − 1)) which is still positive definite. (p may be zero.)

See Also
balance,lu

- compan ................................. Companion Matrix

Description
y=compan(x) returns the companion matrix of x. If x is considered as a polynomial, then the  
eigenvalues of y are roots(x).

See Also
eig,poly,roots
• **cond** .......................... Matrix condition number

**Description**

\( y = \text{cond}(x, p) \) return the matrix condition number, indicating how badly-conditioned the matrix is. Higher values indicate that the matrix is close to singular, which means the inversion or linear solution operations will return unreliable results. \( p \) can be 1, 2, 'fro' or Inf, for which the 1-norm, 2-norm, Frobenius norm or Infinity norm condition numbers are returned, respectively.

\( \text{cond}(x) = \text{cond}(x, 2) \), which returns \( \max(\text{svd}(x)) / \min(\text{svd}(x)) \).

**See Also**

condeig, inv, norm, rcond, svd

• **condeig** .................. Matrix eigenvalue condition number

**Description**

\( y = \text{condeig}(x) \) returns the eigenvalue condition number, indicating how \( x \) is close to having multiple eigenvalues.

\([v, d, y] = \text{condeig}(x)\) also returns the eigenvectors in \( v \) and the eigenvalues in \( d \).

**See Also**

balance, cond, eig, hess

• **det** .......................... Determinant

**Description**

\( y = \text{det}(x) \) is the determinant of \( x \), determined from the LU factorization of the matrix \( x \).
See Also
inv,lu,mrdivide,trace

• **diag** ............................................. Matrix diagonal

Description

\[ y = \text{diag}(x) \] returns the diagonal of matrix \( x \), or, if \( x \) is a vector, constructs a matrix whose diagonal is \( x \).

An optional second argument specifies the diagonal location relative to the main diagonal. Positive values are above the main diagonal and negative values are below. The default value is 0 which is the main diagonal.

See Also
trace,tril,triu

• **eig** ........................................... Eigenvalues and eigenvectors

Description

\[ d = \text{eig}(x) \] returns the eigenvalues of \( x \) in the vector \( d \).

\[ [v,d] = \text{eig}(x) \] returns the eigenvalues on the matrix \( d \) diagonal, and the corresponding eigenvectors in matrix \( v \), such that

\[ x \cdot v = v \cdot d \]

The generalized eigenvalues and eigenvectors can be obtained from calling \[ d = \text{eig}(x,y) \] and \[ [v,d] = \text{eig}(x,y) \] respectively, such that

\[ x \cdot v = y \cdot v \cdot d \]

An optional third argument, 'nobalance', instructs \text{eig} to skip the balancing step.
See Also

balance,cdf2rdf,compan,condeig,hess,norm,qz,qzval,rosser,schur,wilkinson

• eig_D  ................................................. Eigenvalues

Description
d=eig_D(x) returns the eigenvalues of x.

See Also
eig

• eig_W  ................................................. Eigenvectors

Description
v=eig_W(x) returns the eigenvectors of x.

See Also

• expm  ................................................. Matrix exponential

Description
y=expm(x) returns the matrix exponential of x.

See Also
exp,logm
• **givens** ................................. Givens plane rotation

**Description**

\[ z = \text{givens}(a, b) \]  
returns the Givens plane rotation matrix of \( a \) and \( b \).

**See Also**

view


• **hess** ................................. Hessenbreg form

**Description**

\[ y = \text{hess}(x) \]  
is the Hessenberg form of \( x \).

\[ [z, y] = \text{hess}(x) \]  
also returns a unitary matrix \( z \) so that

\[ z' \cdot z = I \]

and

\[ z \cdot y \cdot z' = x. \]

**See Also**

balance,condeig,eig,qz,schur

• **inv** ................................. Matrix Inverse

**Description**

\[ y = \text{inv}(x) \]  
is the matrix inverse of \( x \) such that

\[ x \cdot \text{inv}(x) = I \]

\( x \) must be square matrix.
See Also
cond, det, invhilb, lu, mldivide, mrdivide, mtimes, pinv

• logm ........................................ Matrix logarithm

Description
[y, err] = logm(x) returns the matrix logarithm of x, such that

\[ \expm(\logm(x)) = x \]

for most matrices. err is the estimated error of the result.

See Also
expm, log, sqrtm

• lu .................................................LU factorization

Description
[l, u] = lu(x) return the LU factorization of matrix x, such that \( l \cdot u = x \).
[l, u, p] = lu(x) also return a permutation matrix, p, such that \( p \cdot x = l \cdot u \).

See Also
axxbc, chol, det, inv, mldivide, mrdivide
• **lu_L** ............................... L of LU decomposition
  
  **Description**
  
  $l = lu_L(x)$ returns $l$ from $[l, u] = lu(x)$.

  **See Also**
  
  lu

• **lu_U** ............................... U of LU decomposition
  
  **Description**
  
  $l = lu_U(x)$ returns $u$ from $[l, u] = lu(x)$.

  **See Also**
  
  lu

• **norm** ................................. Norm
  
  **Description**
  
  $z = norm(x, y)$ return the $y$-norm of $x$.

  The following norms exist for vectors:

<table>
<thead>
<tr>
<th>$y$</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inf</td>
<td>$max(abs(x))$</td>
</tr>
<tr>
<td>-Inf</td>
<td>$min(abs(x))$</td>
</tr>
<tr>
<td>otherwise</td>
<td>$sum(abs(x)^y)^{1/y}$</td>
</tr>
</tbody>
</table>

  The following norms exist for matrices:
MatrixVB Function Reference Guide

<table>
<thead>
<tr>
<th>y</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\max(\sum(\text{abs}(x)))$</td>
</tr>
<tr>
<td>2</td>
<td>$\max(\text{svd}(x))$</td>
</tr>
<tr>
<td>Inf</td>
<td>$\max(\sum(\text{abs}(x'))) \quad \text{Inf}$</td>
</tr>
<tr>
<td>'fro'</td>
<td>$\sqrt{\sum(\text{diag}(x' \ast x))}$</td>
</tr>
</tbody>
</table>

$y$ is optional - its default value is 2.

See Also
cond,eig,polyfit,qz,schur,svd

- **null** ........................................... Null space

  **Description**

  $y=	ext{null}(x)$ returns an orthonormal vector basis for the null space of matrix $x$.

  See Also

  orth,qr

- **orth** ......................................... Orthonormal basis

  **Description**

  $y=	ext{orth}(x)$ is a orthonormal basis for $x$.

  See Also

  null,svd
• **pinv** ................................. Pseudo Inverse

*Description*

\[ y = \text{pinv}(x, \text{tol}) \]

returns a pseudo inverse of the matrix \( x \), computed with \( \text{tol} \) tolerance. The default value of \( \text{tol} \) is \( \text{eps} \cdot \text{length}(x) \cdot \text{norm}(x) \).

*See Also*

inv, rank

• **planerot** ............................ Plane rotation

*Description*

\[ [o, y] = \text{givens}(x) \]

computes the Given's plane rotation matrix \( o \) and the rotated matrix

\[ y = o \cdot x \]

*See Also*

qrdelete, qrinsert

• **qr** .............................. QR factorization

*Description*

\[ [q, r] = \text{qr}(x) \]

returns the QR factorization of \( x \) such that \( q \) is unitary, \( r \) is upper triangular and

\[ x = q \cdot r \]

\[ [q, r, p] = \text{qr}(x) \]

returns the QR factorization of \( x \) such that \( q \) is unitary, \( r \) is upper triangular and

\[ x \cdot p = q \cdot r \]

\( p \) is a permutation matrix, computed so \( r \) diagonal values are decreasing in absolute value.
See Also
mldivide,null,qrdelete,qrinsert

- **qrdelete** .................. Update QR factorization
  Description
  \[\[q_2, r_2] = \text{qrdelete}(q, r, j)\] is the QR factorization of \(x\) with its \(j\) column removed, assuming \([q, r] = \text{qr}(x)\). This function is more economical than recomputing the QR factorization for the modified \(x\).

  See Also
  qr, qrinsert

- **qrinsert** .................. Update QR factorization
  Description
  \[\[q_2, r_2] = \text{qrinsert}(q, r, j, c)\] is the QR factorization of \(x\) with the column \(c\) inserted before the \(j\) column, assuming \([q, r] = \text{qr}(x)\). This function is more economical than recomputing the QR factorization for the modified \(x\).

  See Also
  qr, qrdelete
• **qz** .......................... QZ factorization

**Description**

\[ [a, b, q, z, v] = \text{qz}(x, y) \]

returns the generalized eigenvalues and generalized eigenvector \( v \). The left and right eigenvalues are found on \( a \) and \( b \) diagonals, respectively. The following holds:

\[
q \cdot x \cdot z = a \\
q \cdot y \cdot z = b \\
x \cdot v \cdot \text{diag}(b) = y \cdot v \cdot \text{diag}(a)
\]

**See Also**

balance, eig, hess, norm, schur

• **qzval** ........................ Generalized eigenvalues

**Description**

\[ e = \text{qzval}(a, b) \]

does the generalized eigenvalues of \( a \) and \( b \).

**See Also**

eig

• **rank** .......................... Matrix rank

**Description**

\[ y = \text{rank}(x, \text{tol}) \]

returns the number of singular values of \( x \) larger than \( \text{tol} \). This is an estimate for the rank of \( x \). The default value of \( \text{tol} \) is \( \text{eps} \cdot \text{length}(x) \cdot \text{norm}(x) \).
See Also
mrdivide,pinv,rcond,svd

- **rcond** ...... Estimate for the reciprocal condition number

  **Description**
  
y=rcond(x) is an estimate for the reciprocal of x condition number.

  **See Also**
  cond,rank

- **rref** ...................... Reduced row echelon form

  **Description**
  
y=rref(x) returns the reduced row echelon form of x.

  **See Also**
  qr,svd

- **rsf2csf** ............. Real to complex Schur factorization

  **Description**
  
  [u,t]=rsf2csf(u,t) converts the real result from schur into a complex result. In the real result, complex eigenvalues are stored with separate real and imaginary values. The complex result combines them.
See Also
schur

- **schur** ................................. Schur factorization

  **Description**
  
  \[ [u, \mathbf{t}] = \text{schur}(x) \]
  returns the Schur factorization of \( x \), as Schur matrix \( \mathbf{t} \) and unitary matrix \( u \), such that
  
  \[
  u' \cdot u = I \\
  x = u \cdot t \cdot u'
  \]

  **See Also**
  
  eig, hess, norm, qz, rsf2csf

- **schur_T** .............................. \( \mathbf{T} \) of Schur decomposition

  **Description**
  
  \( \mathbf{t} = \text{schur}_T(x) \)
  returns \( \mathbf{t} \) of \( [u, \mathbf{t}] = \text{schur}(x) \).

  **See Also**

- **schur_U** .............................. \( \mathbf{U} \) of Schur decomposition

  **Description**
  
  \( u = \text{schur}_U(x) \)
  returns \( u \) of \( [u, \mathbf{t}] = \text{schur}(x) \).
See Also
schur

- **sqrtm** .......................... Matrix square root

  Description
  
y=\text{sqrtm}(x)\text{ is the matrix square root of }x.

  See Also
  logm, mtimes, sqrt

- **subspace** ......................... Angle between subspaces

  Description
  
z=\text{subspace}(x,y)\text{ returns the angle between the subspaces spanned by }x\text{ and }y.

  See Also
  cos, cross, dot

- **svd** ............................... Singular values decomposition

  Description
  
s=\text{svd}(x)\text{ returns the singular values of }x\text{ in a vector.}
[u,s,v]=svd(x) returns the singular values of x on the diagonal of s. u and v are unitary matrices such that

\[ x = u \cdot s \cdot v' \]

A zero optional second argument indicates that u and s will not have more columns than x.

See Also
cond,norm,orth,rank,rref

- svd_S
  S of SVD decomposition
  Description
  s=svd_S(x) returns s of \([u,s,v]=svd(x)\).  
  See Also
  svd

- svd_U
  U of SVD decomposition
  Description
  u=svd_U(x) returns u of \([u,s,v]=svd(x)\).
  See Also
  svd
- **svd_W** .......................... V of SVD decomposition

  **Description**
  
  \(v = \text{svd}_W(x)\) returns \(v\) of \([u, s, v] = \text{svd}(x)\).

  **See Also**
  
  `svd`

- **trace** .......................... Matrix trace

  **Description**
  
  \(y = \text{trace}(x)\) is the matrix trace, \(\text{sum(diag(x))}\).

  **See Also**
  
  `det, diag, sum`

- **tril** .......................... Lower triangular matrix

  **Description**
  
  \(y = \text{tril}(x, k)\) returns the lower triangular part of \(x\) below a diagonal.

  An optional second argument specifies the diagonal location relative to the main diagonal. Positive values are above the main diagonal and negative values are below. The default value is 0 which is the main diagonal.

  **See Also**
  
  `diag, triu`
• **triu** .......................... Upper triangular matrix

**Description**

\[ y = \text{triu}(x, k) \]

returns the upper triangular part of \( x \) above a diagonal.

An optional second argument specifies the diagonal location relative to the main diagonal. Positive values are above the main diagonal and negative values are below. The default value is 0 which is the main diagonal.

**See Also**

diag, tril
8 Bit* functions

- bitand .................................................... Bitwise AND

  Description
  bitand(x,y) returns the bitwise AND of x and y.
  bitand is named mbitand in Visual Basic programs.

  See Also
  bitcmp, bitget, bitmax, bitor, bitreset, bitset, bitshift, bitxor, mbitand

- bitcmp ................................. Bitwise complement

  Description
  bitcmp(x,y) returns the bitwise complement of x as a y-digit number.

  See Also
  bitand, bitget, bitmax, bitor, bitreset, bitset, bitshift, bitxor

- bitget ................................. Bitwise get bit

  Description
  bitget(x,y) returns the y-th bit from the number x.

  See Also
  bitand, bitcmp, bitmax, bitor, bitreset, bitset, bitshift, bitxor
• **bitmax**  ............... Maximum representable integer

**Description**

`bitmax` returns the maximum integer number that can be represented as a floating point number. This is $2^m - 1$, where $m$ is the number of bits in the floating point mantissa.

**See Also**

`bitand`, `bitcmp`, `bitget`, `bitor`, `bitreset`, `bitset`, `bitshift`, `bitxor`

• **bitor**  ........................................ Bitwise OR

**Description**

`bitor(x, y)` returns the bitwise OR of $x$ and $y$.

`bitor` is named `mbitor` in Visual Basic programs.

**See Also**

`bitand`, `bitcmp`, `bitget`, `bitmax`, `bitreset`, `bitset`, `bitshift`, `bitxor`, `mbitor`

• **bitreset**  .............................. Bitwise reset bit

**Description**

`bitreset(x, y)` resets the $y$-th bit in the number $x$.

**See Also**

`bitand`, `bitcmp`, `bitget`, `bitmax`, `bitor`, `bitshift`, `bitxor`
• **bitset** ................................. Bitwise set bit

  **Description**  
  bitset(x, y, b) sets the y-th bit in the number x to b.  
  Also, bitset(x, y) = bitset(x, y, 1).

  **See Also**  
  bitand, bitcmp, bitget, bitmax, bitor, bitshift, bitxor

• **bitshift** ............................... Bitwise shift

  **Description**  
  bitshift(x, y) returns the number x shifted y bits, bitshift(x, y) = x \cdot 2^y but is computed directly, by changing the floating-point exponent. y > 0 shifts to the left ; y < 0 shifts to the right.

  **See Also**  
  bitand, bitcmp, bitget, bitmax, bitor, bitreset, bitset, bitxor

• **bitxor** ................................. Bitwise XOR

  **Description**  
  bitxor(x, y) returns the bitwise XOR of x and y.

  **See Also**  
  bitand, bitcmp, bitget, bitmax, bitor, bitreset, bitset, bitshift
• **mbitand** ................................. Bitwise AND

**Description**

`bitand` is named `mbitand` in Visual Basic programs.

**See Also**

`bitand`

• **mbitor** ................................. Bitwise OR

**Description**

`bitor` is named `mbitor` in Visual Basic programs.

**See Also**

`bitor`
9 Control functions

•  
  **ltifr** .................................. LTI frequency response
  
  **Description**
  
y=ltifr(a,b,s) is the frequency response of the system \( y(s) = (sI-a)\backslash b \).

  **See Also**
  
  ltitr

  
  
  
•  
  **ltitr** ............................... LTI time response
  
  **Description**
  
x=ltitr(a,b,y) is the time response of the system \( x(n+1) = Ax(n) + By(n) \).

  **See Also**
  
  ltifr

  
  
  
•  
  **ss2tf** .......................... State space to transfer function
  
  **Description**
  
  \([num,den]=ss2tf(a,b,c,d,iu)\) represents the system \( a,b,c,d \) and the input \( iu \) by its transfer function

  \[
  H(s) = \frac{num(s)}{den(s)}
  \]
See Also
ss2zp,tf2ss,tf2zp,zp2ss,zp2tf

- **ss2zp** .......................... State space to zero pole
  
  **Description**
  
  \[ [z,p,k] = ss2zp(a,b,c,d,iu) \]
  
  represents the system \( a,b,c,d \) and the input \( iu \) by its zeros in \( z \), poles in \( p \) and gain \( k \).
  
  \[
  H(s) = k \frac{(s - z_1)(s - z_n)}{(s - p_1)...(s - p_m)}
  \]
  
  where \( n=\text{length}(z) \) and \( m=\text{length}(p) \).

  **See Also**
  
  ss2tf,tf2ss,tf2zp,zp2ss,zp2tf

- **tf2ss** .......................... Transfer function to state space
  
  **Description**
  
  \([a,b,c,d] = tf2ss(num,den)\) represents the system

  \[
  H(s) = \frac{\text{num}(s)}{\text{den}(s)}
  \]

  by its state space equations

  \[
  x = Ax + Bu \quad y = Cx + Du
  \]

  where \( u \) is the system input and \( y \) is its output.
See Also
ss2tf, ss2zp, tf2zp, zp2ss, zp2tf

• **tf2zp**  ...................... Transfer function zero pole

Description

\[ [z, p, k] = \text{tf2zp}(\text{num}, \text{den}) \]

represents the system

\[ H(s) = \frac{\text{num}(s)}{\text{den}(s)} \]

by its zeros \((z)\), poles \((p)\) and gain \((k)\)

\[ H(s) = k \frac{(s - z_1)(s - z_n)}{(s - p_1)(s - p_m)} \]

See Also
ss2tf, ss2zp, tf2ss, zp2ss, zp2tf

• **zp2ss**  ......................... zero pole to state space

Description

\[ [a, b, c, d] = \text{zp2ss}(z, p, k) \]

represents the system

\[ H(s) = k \frac{(s - z_1)(s - z_n)}{(s - p_1)(s - p_m)} \]

by its state space equations

\[ x = Ax + Bu = Cx + Du \]

where \(u\) is the system input and \(y\) is its output.
See Also
ss2tf, ss2zp, tf2ss, tf2zp, zp2tf

- **zp2tf** ................... zero pole to transfer function

**Description**

\[ [\text{num}, \text{den}] = \text{zp2tf}(z, p, k) \]

represents the system

\[
H(s) = k \frac{(s - z_1)(s - z_2) \cdots (s - z_n)}{(s - p_1)(s - p_2) \cdots (s - p_m)}
\]

by its transfer function

\[
H(s) = \frac{\text{num}(s)}{\text{den}(s)}
\]

where \( u \) is the system input and \( y \) is its output.

See Also
ss2tf, ss2zp, tf2ss, tf2zp, zp2ss
10 Set operations

• **intersect** ........................ Sets intersection

  **Description**

  \[[z, idx_x, idx_y] = \text{intersect}(x, y)\] returns the set intersection between \(x\) and \(y\). In addition,

  \[
  z = x(idx_x) \\
  z = y(idx_y)
  \]

  **See Also**

  setxor, union, unique

• **ismember** ........................ Is member in set

  **Description**

  \([z = \text{ismember}(x, s)]\) return 1 if \(x\) is a member of the set \(s\) and 0 otherwise.

  **See Also**

  setdiff, setxor, unique

• **munion** ........................ Set union

  **Description**

  \(\text{union}\) is named \textit{munion} in Visual Basic programs.
See Also
union

- **setdiff** .......................... *Set difference*

  **Description**

  
  \[ [z, idx] = \text{setdiff}(x, y) \]

  returns the set difference between \( x \) and \( y \), elements from \( x \) which do not appear in \( y \). In addition, \( z = x(idx) \).

  **See Also**

  ismember, setxor, union

- **setxor** .......................... *Set XOR*

  **Description**

  
  \[ [z, idx_x, idx_y] = \text{setxor}(x, y) \]

  returns the set exclusive or between the set \( x \) and the set \( y \). In addition, \( z = \text{unique}([x(idx_x); y(idx_y)]) \).

  **See Also**

  setdiff, union, unique

- **union** .......................... *Set union*

  **Description**

  
  \[ [z, idx_x, idx_y] = \text{union}(x, y) \]

  returns the set union between the set \( x \) and the set \( y \). In addition, \( z = \text{unique}([x(idx_x); y(idx_y)]) \).
union is named munion in Visual Basic programs.

**See Also**
ismember,munion,setxor,unique

- **unique** ............................. Unique elements

**Description**

[y,x_idx,y_idx]=unique(x) returns the unique elements of x, sorted. In addition,

\[
y = x(x\_idx)
\]

\[
x = y(y\_idx)
\]

**See Also**
intersect,ismember,setxor,union
11 File and text I/O functions

- **cd** ................................. Change directory
  
  **Description**
  
  cd(dir) changes the current working directory to dir.

  **See Also**
  
  chdir, delete, dir, exist, mkdir, pwd, rmdir, system, type

- **chdir** ................................. Change directory
  
  **Description**
  
  chdir(dir) changes the current working directory to dir.

  **See Also**
  
  cd, mkdir, rmdir

- **copyfile** ................................. Copy file
  
  **Description**
  
  copyfile(src, dest) copies the file named src to a file named dest.

  **See Also**
  
  delete, exist
• `delete` .................................. Delete file

Description
`delete('filename')` erases the file named `filename`.
`delete` is named `deleteM` in Visual Basic programs.

See Also
cd, copyfile, deleteM, deleteobj, dir, exist, findobj

• `deleteM` .................................. Delete file

Description
`delete` is named `deleteM` in Visual Basic programs.

See Also
delete

d • `dir` ...................................... Show directory listing

Description
`dir` produces a directory listing, obtained by running the local dir command. d=dir returns the directory information in a structure, with name, date, bytes and isdir fields.

See Also
cd, delete, exist, fopen, system, type
• dos ................................. Run DOS command

Description
dos(cmd) runs the DOS command cmd. With two output arguments, [status,out]=dos(cmd), the return status is status and the command output is out.

See Also
unix

• error ............................... Abort with error

Description
error(msg) displays the error message msg and aborts the program. If msg is empty, the call is ignored.

See Also
disp,lasterr,quit,warning

• exist ............................... File existence

Description
exist(name) returns 1 if the given name is a variable in the current scope, 2 if a file named name exists, 7 if a directory named name exists and 0 otherwise.
In compiled program, exist only checks for file existence.

See Also
cd,clear,copyfile,delete,dir,fopen,global
•  **fclose**  ........................................... Close file handle

**Description**

\( y = \text{fclose}(\text{fid}) \) closes the file handle \( \text{fid} \). If \( \text{fid} \) is not a handle for an open file, a warning message is printed.

\( y \) is 0 upon success and -1 otherwise.

**See Also**

feof, ferror, fflush, fopen, fprintf, fread, fscanf, fseek, ftell, fwrite, printf, sprintf, sscanf

•  **feof**  ............................................ Is end of file

**Description**

\( y = \text{feof}(\text{fid}) \) returns 1 if the file \( \text{fid} \) have been completely read, and 0 otherwise.

**See Also**

fclose, ferror, fopen

•  **ferror**  ........................................ File I/O error message

**Description**

\( y = \text{ferror}(\text{fid}) \) returns a text error message associated with the file descriptor \( \text{fid} \). If no error had occurred, \( y \) is empty.

An optional second input argument, ’clear’, clears the error message.

**See Also**

fclose,feof, fgetl, fgets, fopen
• **fflush** .................................................Flush file

Description

`fflush(fid)` flushes the file `fid` without closing it.

See Also

fclose, fopen, fwrite

• **fgetl** ....................................................Get line from file

Description

`y=fgetl(fid)` returns the next input line from the file `fid`. The line terminator is not included in `y`.

See Also

ferror, fgets, fopen, fread, fscanf

• **fgets** ..................................................Get line from file

Description

`y=fgets(fid)` returns the next input line from the file `fid`. The line terminator is included in `y`. A second optional input argument, `n`, specifies the maximal number of characters which will be read.

See Also

ferror, fgetl, fopen, fread, fscanf
• **filesep** ................................. File seperator

  **Description**

  `f=filesep` is the character that is used to separate path parts.

  **See Also**

  `fullfile`, `pathsep`

• **fopen** ................................. Open file handle

  **Description**

  `fid=fopen(filename,mode,format)` opens the file `filename`. `mode` can be either `'r'` for reading the file, `'w'` for writing it, or `'a'` for appending to it.

  The `format` argument specifies the machine on which the file was created, which may one of `'native'` for the same machine, `'ieee-le'` for little-endian machines or `'ieee-be'` for big-endian machines.

  In case of error while opening the file, -1 is returned as file id. An optional output argument, `message`, also returns the error message.

  `fopen('all')` returns a vector of all open fids.

  `[filename,mode,format]=fopen(fid)` returns the filename, mode and machone format for `fid`.

  **See Also**

  `exist`, `fclose`, `feof`, `ferror`, `fflush`, `fgetl`, `fgets`, `fprintf`, `fread`, `fseek`, `ftell`, `fwrite`, `sscanf`
• `fprintf` .................. Send formatted output to a file

**Description**

`y=fprintf(fid,format,...)` formats the variables specified according to the format string and sends the text output to the file `fid`.

Documentation for the format string can be found in the C library documentation for the `printf` command.

**See Also**

disp, fclose, fopen, format, printf, sprintf
• **fread** ................................. Read from file

**Description**

\[ y=fread(fid,\text{maxelem},\text{datatype}) \]

does up to \text{maxelem} elements of type \text{datatype} from the file \text{fid}.

\text{maxelem} may be \text{Inf}, in which case all the file is read, an integer scalar or \([m \ n]\) where the elements read are stored in a \(m\)-by-\(n\) matrix in column-major filling order.

The element type, \text{datatype} may be one of:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>'char'</td>
<td>Character</td>
</tr>
<tr>
<td>'schar'</td>
<td>Signed character</td>
</tr>
<tr>
<td>'uchar'</td>
<td>Unsigned character</td>
</tr>
<tr>
<td>'int8'</td>
<td>Signed 8-bit integer</td>
</tr>
<tr>
<td>'uint8'</td>
<td>Unsigned 8-bit integer</td>
</tr>
<tr>
<td>'int16'</td>
<td>Signed 16-bit integer</td>
</tr>
<tr>
<td>'uint16'</td>
<td>Unsigned 16-bit integer</td>
</tr>
<tr>
<td>'int32'</td>
<td>Signed 32-bit integer</td>
</tr>
<tr>
<td>'uint32'</td>
<td>Unsigned 32-bit integer</td>
</tr>
<tr>
<td>'int64'</td>
<td>Signed 64-bit integer</td>
</tr>
<tr>
<td>'uint64'</td>
<td>Unsigned 64-bit integer</td>
</tr>
<tr>
<td>'float32'</td>
<td>IEEE-754 float</td>
</tr>
<tr>
<td>'float64'</td>
<td>IEEE-754 double</td>
</tr>
</tbody>
</table>

An optional input argument, \text{skip}, causes \text{fread} to skip the specified number of bytes after each read.

The number of items actually read is returned in an optional second output argument, \text{count}.

**See Also**

fclose,fgetl,fgets,fopen,fscanf,fwrite,save
MatrixVB Function Reference Guide

- **frewind** ................................................. Move file pointer

  **Description**

  `frewind(fid)` seeks to the start of the file `fid`. This is identical to `fseek('bof',0)`.

  **See Also**

  `fseek, ftell`

- **fscanf** ................................................. Get formatted input from file

  **Description**

  `y=fscanf(fid,format,maxelem)` returns the values, read from file `fid`, according to the `format` string.

  `maxelem` may be `Inf`, in which case all the file is read, an integer scalar or `[m n]` where the elements read are stored in a `m`-by-`n` matrix in column-major filling order.

  Documentation for the format string can be found in the C library documentation for the `scanf` command.

  **See Also**

  `fclose, fgetl, fgets, fopen, fread, fwrite, input, sscanf`

- **fseek** .................................................. Seek in file

  **Description**

  `fseek(fid,pos,where)` seeks to the position `pos` in the file `fid`. The `where` argument specifies how `pos` is interpreted:

  - `'bof'` From start of file.
  - `'cof'` From current file pointer.
'eof' From the end of the file.
The function returns 0 upon success and -1 otherwise.

See Also
fclose,fopen,frewind,ftell

• **ftell**  ......................Tell file position

Description
pos=ftell(fid) returns the current file pointer position. Upon error, -1 is returned.
A method for determining file size would be:
```
 fid=fopen(filename)
fseek(fid,0,'eof');
filesize=ftell(fid);
fclose(fid);
```

See Also
fclose,fopen,frewind,fseek

• **fullfile**  ......................Filename from parts

Description
p=fullfile(dir,filename) constructs a file name out of its concatenated parts. dir may be repeated as many times as needed.

See Also
filesep,pathsep
• **fwrite** .................................................. Write to file

  **Description**

  $y = fwrite(fid,x,datatype)$ writes $x$ (as $datatype$ elements) to the file $fid$.

  The element type, $datatype$ may be one of

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>'char'</td>
<td>Character</td>
</tr>
<tr>
<td>'schar'</td>
<td>Signed character</td>
</tr>
<tr>
<td>'uchar'</td>
<td>Unsigned character</td>
</tr>
<tr>
<td>'int8'</td>
<td>Signed 8-bit integer</td>
</tr>
<tr>
<td>'uint8'</td>
<td>Unsigned 8-bit integer</td>
</tr>
<tr>
<td>'int16'</td>
<td>Signed 16-bit integer</td>
</tr>
<tr>
<td>'uint16'</td>
<td>Unsigned 16-bit integer</td>
</tr>
<tr>
<td>'int32'</td>
<td>Signed 32-bit integer</td>
</tr>
<tr>
<td>'uint32'</td>
<td>Unsigned 32-bit integer</td>
</tr>
<tr>
<td>'int64'</td>
<td>Signed 64-bit integer</td>
</tr>
<tr>
<td>'uint64'</td>
<td>Unsigned 64-bit integer</td>
</tr>
<tr>
<td>'float32'</td>
<td>IEEE-754 float</td>
</tr>
<tr>
<td>'float64'</td>
<td>IEEE-754 double</td>
</tr>
</tbody>
</table>

  An optional input argument, $skip$, causes $fread$ to skip the specified number of bytes after each read.

  The number of items actually written is returned in an optional second output argument, $count$.

  **See Also**

  fclose, fflush, fopen, fread, fscanf, load

• **getenv** .................................................. Get environment variable

  **Description**

  $y = getenv(x)$ returns the value of the environment variable $x$. 
See Also

home

Description

come home homes the cursor to the top-left corner.

See Also
clc, disp

load

Description

cload(filename, var1, ...) loads the specified variables from the file filename. This command automatically determines the file type and version. var1, ... can be omitted if the file filename is available during compile time.

See Also
fwrite, loadcell, save, vbload

menu

Description

y=menu(title, option1,option2,...) presents the user with a menu titled title with options option1, option2,.... The number of the option selected by the user is returned.
See Also
disp,input,uimenu

- **mkdir** ............................... Create directory
  
  Description
  
  mkdir(dir) creates a subdirectory named dir under the current working directory.

  See Also
cd, chdir, rmdir

- **printf** ............................ Send formatted output to screen
  
  Description
  
  y=printf(format,...) formats the variables specified according to the format string and sends the text output to the screen.

  Documentation for the format string can be found in the C library documentation for the printf command.

  See Also
  blanks, disp, fclose, fopen, fprintf, sprintf

- **pwd** ............................... Get working directory
  
  Description
  
  dir=pwd returns the current working directory.
See Also

- `cd` - Change directory

- `rmdir` - Remove directory

**Description**

`rmdir(dir)` removes the subdirectory named `dir`.

**See Also**

`cd, chdir, mkdir`

- `save` - Save variables to file

**Description**

`save(filename,var1,...)` saves the specified variables to the file `filename`. The optional argument `-ascii`, saves the variable in ASCII format, while the optional argument `-V4`, saves the variable in MAT-4 format.

**See Also**

`fread, load, savecell, vbsave`

- `tempdir` - Temporary directory

**Description**

`d=tempdir` is the system temporary directory.
See Also
fopen,tempname

- **tempname** ................. Temporary file name

  Description
  
f=tempname is a name for a temporary file.

  See Also
  fopen,tempdir

- **vbload** .................. Load variables from file

  Description
  
  var1=load(filename) loads the first variable from the file filename. This command automatically determines the file type and version.

  See Also
  load,vbsave

- **vbsave** .................. Save variables to file

  Description
  
  save(filename,var1,...) saves the specified variables to the file filename, in the mat-5 binary file format.
See Also
save,vbload

• **warning** .......................... Display a warning

Description

**warning(msg)** displays the warning message *msg*. If *msg* is empty, the call is ignored.

See Also
disp,error,quit

• **wavread** .......................... Read wave file

Description

[y,f,b]=wavread(filename) reads the .wav file data into the column vector *y*. Stereo data is returned as a two-column vector. The sample frequency is returned in *f* and the number of bits per sample in *b*. The *f* and *b* output arguments are optional.

See Also
sound,wavwrite

• **wavwrite** .......................... Write wave file

Description

wavwrite(y,f,b,filename) writes the .wav file data from the column vector *y*. Stereo data is supplied as a two-column vector. The sample frequency is in *f* and the number of bits per
sample in b. f and b are optional: the default values are 44,100 Hz and 16 bits, respectively.

See Also

sound, wavread
12 Function functions

- **ode15s** .......................... Solve stiff ODE
  
  **Description**
  
  See odeNN.

  **See Also**
  
  ode23,ode23s,ode45,odeget,odeNN,odeset

- **ode23** .......................... Solve ODE with Runge-Kutta (2,3)
  
  **Description**
  
  See odeNN.

  **See Also**
  
  ode15s,ode23s,ode45,odeget,odeNN,odeset

- **ode23s** .......................... Solve stiff ODE
  
  **Description**
  
  See odeNN.

  **See Also**
  
  ode15s,ode23,ode45,odeget,odeNN,odeset
• **ode45**  .................Solve ODE with Runge-Kutta (4,5)

**Description**

See `odeNN`.

**See Also**

`ode15s,ode23,ode23s,odeget,odeNN,odeset`

• **odeget**  .......................Get ode* options

**Description**

val=odeget(options,'option',def) is the value of 'option' as set in the variable options. In case the option was not found, val=def.

Options for the ode* functions are -

'Abstol' specifies the absolute tolerance of the solution.

'Reltol' specifies the relative tolerance of the solution.

'InitialStep' is the initial step size used by the solver.

'Refine' produces interpolated output.

**See Also**

`ode15s,ode23,ode23s,ode45,odeNN,odeset`

• **odeNN**  .................Solve ordinary differential equation

**Description**

[t,y]=odeNN('f',trange,y0,options) solves the ordinary differential equation f, starting from initial condition y0 in the time range trange(1)...trange(2). If trange contains more than two points, f is solved for all trange values exactly. options may be used to control the
solution accuracy, see `odeset`. Optional arguments may be provided to \( f \) by passing them to \( \text{odeNN} \) after `options`.

**See Also**

`ode15s,ode23,ode23s,ode45,odeget,odeset`

- **odeset** ................................. Set ode* options

**Description**

\[
\text{options=odeset(options,name,value,name,value,...)}
\]

sets specified \((\text{name},\text{value})\) pairs as ode options. For a list of options, see `odeget`.

**See Also**

`ode15s,ode23,ode23s,ode45,odeget,odeNN`

- **quad** ................................. Integrate \( f(x) \)

**Description**

\[
\text{quad(f,a,b,tol)}
\]

integrates the function \( f \) from \( a \) to \( b \) with tolerance \( \text{tol} \). The integration is approximate and will be less accurate for non-smooth functions. Optional arguments may be provided to \( f \) by passing them to `quad` after `tol`, preceded by an empty matrix.

**See Also**

`quad8`
- **quad8** ..........................Integrate f(x)

**Description**

quad8 is identical to quad.

**See Also**

quad
13 Is* functions

- **finite** ........................................... Is finite

  **Description**
  
  finite is identical to isfinite.

  **See Also**
  
  isfinite

- **is_scalar** ......................................... Is scalar

  **Description**
  
  y=is_scalar(x) returns 1 if x is a scalar and 0 otherwise.

  **See Also**
  
  is_square, is_vector, size

- **is_square** ......................................... Is square matrix

  **Description**
  
  y=is_square(x) returns 1 if x is a square matrix and 0 otherwise.

  **See Also**
  
  columns, is_scalar, rows, size
- **is_struct** ........................................... Is structure
  
  **Description**
  
  \( y = \text{is}\_\text{struct}(x) \) returns 1 if \( x \) is a structure and 0 otherwise.

  **See Also**
  
  fieldnames, isa, struct

- **is_vector** .............................................. Is vector
  
  **Description**
  
  \( y = \text{is}\_\text{vector}(x) \) returns 1 if \( x \) is a vector and 0 otherwise.

  **See Also**
  
  columns, is_scalar, rows, size

- **iscell** .................................................. Is cell matrix
  
  **Description**
  
  \( y = \text{iscell}(x) \) returns 1 if \( x \) is a cell array and 0 otherwise.

  **See Also**
  
  cell, isa
• `ischar` ........................................... Is char matrix

Description

\[ y = \text{ischar}(x) \] returns 1 if \( x \) is a char matrix and 0 otherwise.

See Also

`cellstr`, `char`, `isa`, `iscellstr`

• `iscomplex` ........................................ Is matrix complex

Description

\[ y = \text{iscomplex}(x) \] returns 1 if \( x \) is complex and 0 otherwise.

See Also

`imag`, `isreal`, `real`

• `isempty` ........................................... Is empty matrix

Description

\[ y = \text{isempty}(x) \] returns 1 if \( x \) is an empty matrix and 0 otherwise.

See Also

`size`, `zeros`
• **isequal** .......................... Are arguments equal

  **Description**
  
  $z = \text{isequal}(x, y)$ returns 1 if $x$ and $y$ have the same size and contain same elements and 0 otherwise.

  **See Also**
  
  any, eq, size

• **isfinite** .......................... Is finite

  **Description**
  
  $y = \text{isfinite}(x)$ returns 1 if $x$ is finite and 0 otherwise.

  **See Also**
  
  finite, Inf, isinf, isnan, NaN

• **isinf** .......................... Is infinite

  **Description**
  
  $y = \text{isinf}(x)$ returns 1 if $x$ is not finite and 0 otherwise.

  **See Also**
  
  Inf, isfinite, isnan, NaN
• **isletter** ................................................. Is letter

**Description**
y=isletter(x) returns 1 if x is an alphabet letter and 0 otherwise.

**See Also**
char, isspace

• **islogical** ............................................... Is logical

**Description**
y=islogical(x) returns 1 if x is logical and 0 otherwise.

**See Also**
all, any, le, logical

• **islower** ............................................... Is lowercase

**Description**
y=islower(x) returns 1 if x is a lowercase letter and 0 otherwise.

**See Also**
isupper, lower, upper
• **isnan** .................................................................................. Is Not-A-Number

**Description**
y=isnan(x) returns 1 if x is Not-A-Number and 0 otherwise.

**See Also**
Inf,isfinite,isinf,NaN

• **isnumeric** .............................................................................. Is matrix numeric

**Description**
y=isnumeric(x) returns 1 if x is a numeric matrix and 0 otherwise.

**See Also**
char,double,num2str,str2num

• **isprime** .................................................................................. Is matrix prime

**Description**
y=isprime(x) returns 1 if x is a prime number and 0 otherwise.

**See Also**
factor,gcd,primes
• **isreal** .............................................. Is matrix real

**Description**

\( y = \text{isreal}(x) \) returns 1 if \( x \) is a real (non-complex) matrix and 0 otherwise.

**See Also**

conj, imag, iscomplex, real

• **isspace** ............................................. Is space

**Description**

\( y = \text{isspace}(x) \) returns 1 if \( x \) is an ASCII blank space character and 0 otherwise.

**See Also**

char, isletter, strtok

• **isupper** .............................................. Is uppercase

**Description**

\( y = \text{isupper}(x) \) returns 1 if \( x \) is a uppercase letter and 0 otherwise.

**See Also**

islower, lower, upper
14 Optimization functions

- **conls** .......................... Constrained least squares

  **Description**

  \( x = \text{conls}(a, b, c, d) \) returns the best solution, in the least-squares sense, to the problem

  \[
  \min_x \frac{1}{2} |a x - b|^2 \quad \text{s.t.} \quad c x \leq d
  \]

  Additional optional arguments specify lower and upper bounds for \( x \), initial guess and number of equalities.

  **See Also**

  lp, nnls, qp

- **constr** .......................... Constrained optimization

  **Description**

  \( x = \text{constr}('f', x0, \text{options}, \text{xlow}, \text{xhigh}, 'g') \) finds the value of \( x \) for which the first output of \( f \) is minimal in the range constrained by \( f \) second output. \( f(x) \) should return two outputs: the first output is minimized subject to the second output being non-positive.

  \( \text{options} \) is optional, giving the optimization arguments. If \( \text{options} \) is the empty matrix, default options are used.

  \( \text{xlow} \) and \( \text{xhigh} \) provide lower and upper bounds for \( x \) value. \( g(x) \) provides the optional gradient of the function \( f(x) \). Optional arguments may be provided to \( f \) by passing them to \( \text{constr} \) after '\( g \)'.

  **See Also**

  fmin, fmins, fminu
• **curvefit** ................................. Curve fitting

**Description**

\[ x = \text{curvefit}('f', x_0, x_1, y_1, \text{options}, 'g') \]

finds the value of \( x \) for which \( f(x, x_1) \) is closest to \( y_1 \). In other words, it fits the model \( f \) to the experimental data \((x_1, y_1)\).

\( \text{options} \) is optional, giving the optimization arguments. If \( \text{options} \) is the empty matrix, default options are used.

\( g(x) \) provides the optional gradient of the function \( f(x) \). Optional arguments may be provided to \( f \) by passing them to `curevfit` after `'g'`.

**See Also**

`constr`, `leastsq`

---

• **fmin** ................................. Find minimum of \( f(x) \)

**Description**

\[ x = \text{fmin}('f', x_{low}, x_{high}, \text{options}) \]

finds the value of \( x \) for which \( f(x) \) is minimal in the range \([x_{low}, x_{high}]\).

\( \text{options} \) is optional, giving the optimization arguments. If \( \text{options} \) is the empty matrix, default options are used.

Optional arguments may be provided to \( f \) by passing them to `fmin` after \( \text{options} \).

**See Also**

`fmns`, `foptions`, `fzero`
• **fmins**  ..................... Find minimum of \( f(x_1,x_2,...) \)

**Description**

\[ x=fmins('f',x0,options) \]

finds the value of vector \( x \) near \( x_0 \) for which \( f(x) \) is minimal.

*options* is optional, giving the optimization arguments. If *options* is the empty matrix, default options are used.

Optional arguments may be provided to \( f \) by passing them to \( fmin \) after *options*, preceded by an empty matrix.

**See Also**

fmin,foptions,fzero

• **fminu**  ..................... Unconstrained minimization

**Description**

\[ x=fminu('f',x0,options,'g') \]

finds the value of \( x \) for which \( f(x) \) is minimal.

*options* is optional, giving the optimization arguments. If *options* is the empty matrix, default options are used.

*\( g(x) \)* provides the optional gradient of the function \( f(x) \). Optional arguments may be provided to \( f \) by passing them to \( constr \) after 'g'.

**See Also**

constr,fmin,fmins,minimax
• **foptions** ............... Set options to function functions

**Description**

\( \text{options} = \text{foptions} \) returns the default arguments used by the function functions, as a vector of numbers. The more important entries are

- \( \text{options}(2) \) Tolerance of \( x \).
- \( \text{options}(3) \) Tolerance of \( f(x) \).
- \( \text{options}(14) \) Limit number of times the target function will be evaluated. Optimization will stop upon reaching that number.

**See Also**

fmin,fmins,fzero

• **fsolve** .................. Solve non-linear equations

**Description**

\( x = \text{fsolve}('f',x0,\text{options},'g') \) finds the value of \( x \) for which \( f(x) \) is zero.

\( \text{options} \) is optional, giving the optimization arguments. If \( \text{options} \) is the empty matrix, default options are used.

\( g(x) \) provides the optional gradient of the function \( f(x) \). Optional arguments may be provided to \( f \) by passing them to \text{fsolve} after \( 'g' \).

**See Also**

constr,fmin,fminu
• **fzero** ................................. Find zero of f(x)

**Description**

$x = \text{fzero}(f', x0, \text{tol})$ returns $x$ for which $f(x) = 0$. If no such $x$ is found, $x = \text{NaN}$. $x0$ serves as an initial guess. $\text{tol}$ is the tolerance for which a zero will be identified. Optional arguments may be provided to $f$ by passing them to $\text{fzero}$, with an extra empty matrix before them.

**See Also**

fmin, fmins, foptions

• **leastsq** ............................... Non-linear least squares

**Description**

$x = \text{leastsq}(f', x0, \text{options}, g')$ finds the value of $x$ which minimizes the sum of squares of $f(x)$ value.

$\text{options}$ is optional, giving the optimization arguments. If $\text{options}$ is the empty matrix, default options are used.

$g(x)$ provides the optional gradient of the function $f(x)$. Optional arguments may be provided to $f$ by passing them to $\text{leastsq}$ after $g'$.

**See Also**

constr, curvefit, fmin

• **lp** ...................................... Linear programming

**Description**

$x = \text{lp}(c, a, b)$ returns the best solution to the linear programming (LP) problem

$$\min_x cx \ s.t. \ ax \leq b$$
Additional optional arguments supply lower limits and upper limits for \( x \).

**See Also**

conls, nnls, qp

- **minimax**  ......................... Minimax optimization

  **Description**

  \( x = \text{minimax}('f', x_0, \text{options}, 'g') \) finds the value of \( x \) for which the maximal output of \( f(x) \) is minimal.

  \text{options} is optional, giving the optimization arguments. If \text{options} is the empty matrix, default options are used.

  \( g(x) \) provides the optional gradient of the function \( f(x) \). Optional arguments may be provided to \( f \) by passing them to \text{minimax} after \( 'g' \).

  **See Also**

  constr, fminu

- **nnls**  .............................. Non-linear least squares

  **Description**

  \( x = \text{nnls}(a, b, \text{tolerance}) \) returns the best solution, in the least-squares sense, to the problem

  \[
  \min_x |ax - b|^2 \ s.t. \ x \geq 0
  \]

  The optional argument \text{tolerance} specifies how much \( x \) can deviate from zero.
See Also
cons,lp,qp

• **qp** .......................... Quadratic programming

Description

\[ x = \text{qp}(q, c, a, b) \] returns the best solution to the quadratic programming (QP) problem

\[
\min_{x} \frac{1}{2} x'q x + c'x \text{ s.t. } ax \leq b
\]

Additional optional arguments supply lower limits and upper limits for \( x \).

See Also
cons,lp,nnls
15 Polynomials

- **mkpp** ................. Make piecewise polynomial

  **Description**

  $pp = \text{mkpp}(b, c)$ returns a piecewise polynomial with breaks $b$ and coefficients $c$.

  **See Also**

  ppval, spline, unmkpp

- **poly** ...................... Create polynomial

  **Description**

  $y = \text{poly}(x)$ returns

  - $x$ is vector
    The polynomial whose roots are $x$ values.

  - $x$ is square matrix
    The characteristic polynomial $|sI - x|$.
    A N-degree polynomial is represented by a row vector with N+1 elements. The leftmost element is the coefficient of the N-th power; the rightmost element is the polynomial constant.

  **See Also**

  compan, conv, deconv, polyder, polyfit, polyval, polyvalm, roots
• polyder  ..................... Derivative of polynomial

Description

z=polyder(x) is the derivative of polynomial x.
z=polyder(x,y) is the derivative of polynomial x·y.
[n,d]=polyder(x,y) is the derivative \( \frac{n}{d} \) of the polynomial ratio \( \frac{x}{y} \).

See Also

conv,poly,polyfit

• polyfit  ..................... Fit polynomial for data series

Description

z=polyfit(x,y,n) returns the polynomial of degree n most fit in a least-squares sense to the
data series (x,y).

See Also

abs,corrcoef,cov,norm,poly,polyder,polyval,polyvalm

• polyval  ............................ Polynomial value

Description

y=polyval(p,x) is the value of the polynomial p evaluated at point x element-wise.

See Also

poly,polyfit,polyvalm,roots
• **polyvalm** .................... Polynomial matrix value

  **Description**

  \( y = \text{polyvalm}(p, x) \) is the value of the matrix polynomial \( p \) evaluated matrix-wise at \( x \), which must be a square matrix.

  **See Also**

  poly, polyfit, polyval

• **ppval** ..................... Evaluate piecewise polynomial

  **Description**

  \( y = \text{ppval}(pp, x) \) returns the value of the piecewise polynomial \( pp \) at point \( x \).

  **See Also**

  mkpp, spline, unmkpp

• **roots** .......................... Polynomial roots

  **Description**

  \( y = \text{roots}(x) \) return the roots of the polynomial \( x \), for which \( \text{polyval}(x, y) = 0 \).

  **See Also**

  compan, poly, polyval
• **unmkpp** .................Evaluate piecewise polynomial

**Description**

\[ [b, c, 1, k] = \text{unmkpp}(pp) \]

returns the parts of the piecewise polynomial `pp`.

**See Also**

`mkpp`, `ppval`, `spline`
16 Signal processing functions

- **conv** .......................... Convolution

  **Description**
  
  \[ z = \text{conv}(x, y) \]
  
  returns the convolution of \( x \) and \( y \), defined by
  
  \[ z(i) = \sum_j x_j \cdot y_{i-j}. \]
  
  The summation is over all the range for which \( x \) and \( y \) are defined to have values, so that 
  \[ \text{length}(z) = \text{length}(x) + \text{length}(y) - 1. \]
  
  Convolution of two polynomial coefficient vectors is the same as multiplying them. In addition, 
  long convolution can be implemented more efficiently by transforming the vectors, multiplying 
  element-by-element and transforming back.

  **See Also**

  conv2, deconv, fft, filter, ifft, poly, polyder

- **conv2** .......................... 2-D Convolution

  **Description**
  
  \[ z = \text{conv2}(x, y) \]
  
  returns the two-dimensional convolution of \( x \) and \( y \). If \( x \) is separable, the com-
  putation can be made more efficient by calling \( z = \text{conv2}(yrow, ycol, x) \). \( x \) is convoluted with 
  \( yrow \) and \( ycol \) separately.
  
  An optional shape parameter determines what part of the result is returned.
  'full' returns the full result. This is the default.
  'same' returns result which is same size as \( x \).
  'valid' returns the submatrix of the result which was not a result of zero-extending one of the 
  arguments.

  **See Also**

  conv, deconv, fft2, filter2, ifft2
• **corrcoef** .................. Correlation Coefficients

**Description**

\[ z = \text{corrcoef}(x) \]

returns the correlation coefficient of \( x \), computed from

\[ z_{i,j} = \frac{x_{i,j}}{\sqrt{x_{i,i}x_{j,j}}} \]

Also, \( \text{corrcoef}(x,y) = \text{corrcoef}([x \ y]) \).

**See Also**

cov, mean, median, polyfit, std

• **COV** .................. Covariance matrix

**Description**

\[ z = \text{cov}(x) \]

returns the variance of vector \( x \), or covariance of matrix \( x \). The covariance function is defined as

\[ \text{cov}(x) = \frac{1}{m-1}(x - \text{mean}(x))^T \cdot (x - \text{mean}(x)) \]

where \( m \) is number of observations in \( x \).

Also, \( \text{cov}(x,y) = \text{cov}([x \ y]) \). If an additional optional argument, 1, is supplied, the result will be divided by \( m \) instead of \( m-1 \).

**See Also**

corrcoef, mean, median, polyfit, std

• **deconv** .................. Deonvolution

**Description**

\[ z = \text{deconv}(x,y) \]

returns the convolution of \( y \) out of \( x \).
Deconvolution of two polynomial coefficient vectors is the same as dividing them. In addition, long deconvolution can be implemented more efficiently by transforming the vectors, dividing element-by-element and transforming back.

See Also
conv, conv2, fft, filter, ifft, poly

- del2 ............................................. Discrete Laplacian

  Description

  y = del2(x, hx, hy, hz) computes an approximation to the laplacian operator applied to x. hx, hy are used for two-dimensional matrices and all three for three-dimensional matrices. If either hy or hz is omitted, its default value is hx. If hx is not given, its value is 1.

  See Also

  gradient

- dft ............................................. Discrete Fourier transform

  Description

  y = dft(x) is the discrete Fourier transform of x.

  See Also

  fft
• **fft** ................................. Fourier transform

**Description**

\[ y = \text{fft}(x) \] is the 1-dimensional fourier transform of \( x \), defined by

\[ X_k = \sum_{j=1}^{N} x_j \cdot e^{-i(j-1)(k-1)2\pi/N} \]

An optional input argument, \( N \), specify the \( N \)-point fourier transform. \( x \) is padded with zeros or truncated to make it \( N \) points.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{fft}(..., \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

The implementation is highly optimized.

**See Also**

conv, deconv, dft, fft, fftshift, hadamard, ifft, ifft2

• **fft2** ............................. 2-Dimensional fourier transform

**Description**

\( y = \text{fft2}(x) \) is the 2-dimensional fourier transform of \( x \), computed by performing fft over the rows followed by fft over the columns.

Two optional input arguments, \( m \) and \( n \), specify the \( m \)-by-\( n \) -point fourier transform. \( x \) is padded with zeros or truncated along both dimensions to make it \( m \)-by-\( n \) before the computation.

**See Also**

conv2, fft, fftshift, ifft, ifft2
- **fftshift** ................................................. Shift fft results

**Description**

\( y = \text{fftshift}(x) \) rearranges the results of \( \text{fft} \) or \( \text{fft2} \) so that the lower frequency components is in the center, with the higher components at the edges of \( y \).

**See Also**

fft, fft2, ifft, ifft2, ifftshift

- **filter** .................................................. Digital FIR

**Description**

\( y = \text{filter}(b,a,x) \) returns the vector \( x \) filtered by the digital FIR filter whose numerator coefficients are in \( b \) and denominator in \( a \).

An optional input argument, \( zi \), supplies the initial values for the filter delays. An optional second output argument, \( zf \), is assigned the final values of the delays.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{filter}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

**See Also**

conv, deconv, filter2

- **filter2** ............................................. Digital 2-dimensional FIR

**Description**

\( y = \text{filter2}(b,x) \) returns \( x \) filtered with the 2-dimensional FIR \( b \).

An optional shape parameter determines what part of the result is returned.
'full' returns the full result.
'same' returns result which is same size as x. This is the default.
'valid' returns the submatrix of the result which was not a result of zero-extending one of the arguments.

See Also
conv2,filter

• gradient .................................. Gradient

Description
dx=gradient(x) returns an approximation to the one-dimensional gradient of x.
[dx,dy]=gradient(x) returns an approximation to the two-dimensional gradient of x.

See Also
del2

• ifft  .................................. Inverse Fourier transform

Description
y=ifft(x) is the 1-dimensional inverse Fourier transform of x. y is scaled by \( \frac{1}{N} \) so \( \text{fft}(\text{ifft}(x))=x \).

An optional input argument, N, specify the N-point Fourier transform. x is padded with zeros or truncated to make it N points.
The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y=\text{ifft}(\ldots,\text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.
The implementation is highly optimized.
See Also
conv, deconv, fft, fft2, fftshift, ifft2

• **ifft2** ......... 2-Dimensional inverse Fourier transform

Description

\[ y = \text{ifft2}(x) \]

is the 2-dimensional inverse Fourier transform of \( x \), computed by \( \text{ifft} \) over the rows followed by \( \text{ifft} \) over the columns.

Two optional input arguments, \( m \) and \( n \), specify the \( m \) by \( n \) -point inverse Fourier transform. \( x \) is padded with zeros or truncated along both dimensions to make it \( m \) by \( n \) before the computation.

See Also
conv2, fft, fft2, fftshift, ifft

• **ifftshift**.......................... Inverse fftshift

Description

\[ x = \text{ifftshift}(y) \]

is the inverse operation of \( y = \text{fftshift}(x) \) so that

\[ x = \text{ifftshift}(\text{fftshift}(x)) \]

See Also
fftshift, ifft2
• **sinc**  ............................................. Sinc function

   **Description**

   \[ y = \text{sinc}(x) \] returns \[ \frac{\sin(\pi x)}{\pi x} \]

   except when \( x = 0 \) where \( \text{sinc} \) is 1.

   **See Also**

   \( \text{sin} \)


• **unwrap**  ............................................. Smooth jumps

   **Description**

   \[ y = \text{unwrap}(x, \text{tol}) \] smoothes any jumps larger than \( \text{tol} \) along a dimension \( \text{dim} \) in the input data. Such jumps are reduced mod \( 2\pi \).

   The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{unwrap}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

   **See Also**

   \( \text{angle, diff} \)
17 Sparse functions

- **colmmd** .................. Column permutation for sparsity

  Description
  
  `colmmd` is identical to `colperm`. A minimum-degree algorithm is not yet available.

  See Also
  
  `colperm,mldivide,sparse`

- **colperm** .................. Column permutation for sparsity

  Description
  
  `p=colperm(x)` return a column permutation `p` for `x`. The column is computed so that `x(:,p)` columns are sorted by their nonzeros count.

  See Also
  
  `colmmd,mldivide,sparse`

- **full** ........................ Make matrix full

  Description
  
  `y=full(x)` returns a full (dense) copy of the matrix `x`. If `x` was already full, `x` is returned.

  See Also
  
  `double,issparse,sparse,uint8`
• **issparse** .................................................Is sparse

  **Description**

  $y = \text{issparse}(x)$ returns 1 if $x$ is a sparse matrix and 0 otherwise.

  **See Also**

  full, isa, sparse

• **nnz** .......................... Number of non zero elements

  **Description**

  $y = \text{nnz}(x)$ is the number of nonzero elements in the matrix $x$.

  **See Also**

  nonzeros, nzmax, size, spalloc, sparse, spones

• **nonzeros** .......................... Non-zero elements

  **Description**

  $y = \text{nonzeros}(x)$ returns the non zero elements of $x$ in a column vector $y$.

  **See Also**

  find, nnz, nzmax, sparse
• **nzmax** ........ Allocated number of non zero elements

**Description**

\[ y = \text{nzmax}(x) \]

is the maximal number of nonzero elements that can be stored in the sparse matrix \( x \). If a larger number of non zeros is being stored in \( x \), the matrix memory will be reallocated and its contents copied.

**See Also**

nnz, nonzeros, size, spalloc

• **spalloc** ...................... Allocate sparse matrix

**Description**

\[ y = \text{spalloc}(m, n, \text{nzmax}) \]

returns an empty sparse matrix with \( m \) rows and \( n \) columns. \( \text{nzmax} \) elements are allocated for non zero elements.

**See Also**

nnz, nzmax, sparse

• **sparse** ........ Create sparse matrix from nonzero vectors

**Description**

\[ y = \text{sparse}(x) \]

returns the sparse representation of \( x \).

\[ y = \text{sparse}(i, j, s, m, n, \text{nzmax}) \]

creates a sparse matrix with \( m \) rows and \( n \) columns. The matrix nonzero elements are the elements of \( s \) at indices \((i,j)\). The default value for \( \text{nzmax} \) is \( \text{size}(s) \). Default values for \( m \) and \( n \) are \( \max(i) \) and \( \max(j) \), respectively.
See Also
colperm, full, issparse, nnz, nonzeros, spalloc, spconvert, speye, spones, sprand, sprandn, spy

- **spconvert** ............ Create sparse matrix from columns

**Description**

\( y = \text{spconvert}(x) \) creates a sparse matrix from the 3- or 4- column matrix \( x \). Each line of the matrix contains \((i, j, s)\) triplets. A line for which \( s = 0 \) specifies the matrix size. For complex data, a fourth column exists which gives the imaginary part of each value.

See Also
sp2coo, sparse

- **speye** ................................. Sparse identity matrix

**Description**

\( y = \text{speye}(m,n) \) is a \( m \)-by-\( n \) sparse identity matrix, composed of all zeros, except on the main diagonal where it is 1.

Also, \( \text{speye}(n) = \text{speye}(n,n) \) and \( \text{speye([m n])} = \text{speye}(m,n) \).

See Also
sparse, spones, sprand, spzeros
• **spones** ................................. Sparse ones

  **Description**

  \( y = \text{spones}(x) \) returns a sparse matrix with 1 for every nonzero in \( x \).

  **See Also**

  nnz, ones, sparse, speye, sprand, spzeros

• **sprand** ............................... Sparse random matrix

  **Description**

  \( y = \text{sprand}(x) \) returns a sparse matrix with a uniformly-distributed random value for every nonzero in \( x \).

  \( y = \text{sprand}(m, n, \text{density}) \) returns a sparse matrix with \( m \) rows and \( n \) columns. Each nonzero element is uniformly distributed. The percentage of the non zero elements is approximately \( \text{density} \).

  **See Also**

  rand, sparse, speye, spones, sprandn, spy

• **sprandn** ............................... Sparse random matrix

  **Description**

  \( y = \text{sprandn}(x) \) returns a sparse matrix with a uniformly-distributed random value for every nonzero in \( x \).

  \( y = \text{sprandn}(m, n, \text{density}) \) returns a sparse matrix with \( m \) rows and \( n \) columns. Each nonzero element is normally distributed. The percentage of the non zero elements is approximately \( \text{density} \).
See Also
randn, sparse, sprand

- **spzeros** ............................. Sparse zeros matrix

Description

\[ y = \text{spzeros}(m, n) \] is a \( m \)-by-\( n \) sparse zeros matrix, composed of zeros.

Also, \( \text{spzeros}(n) = \text{spzeros}(n, n) \) and \( \text{spzeros}([m \ n]) = \text{spzeros}(m, n) \).

See Also

speye, spones
18  Size and shape functions

- **columns** .............................. Number of columns

  **Description**

  \( y = \text{columns}(x) \) returns the number of \( x \) columns.

  **See Also**

  is_square, is_vector, rows, size

- **flipdim** ............................... Flip matrix at a dimension

  **Description**

  \( y = \text{flipdim}(x, \text{dim}) \) flips \( x \) along the specified dimension. It holds that

  \[
  \text{flipdim}(x, 1) = \text{flipud}(x) \\
  \text{flipdim}(x, 2) = \text{fliplr}(x)
  \]

  **See Also**

  ctranspose, fliplr, flipud, rot90, size, transpose

- **fliplr** .............................. Flip matrix left-right

  **Description**

  \( y = \text{fliplr}(x) \) returns the left-right flipped matrix \( x \).
See Also
flipdim,flipud,rot90

• flipud ............................ Flip matrix up-down

Description
y=flipud(x) returns the up-down flipped matrix x.

See Also
flipdim,flipud,rot90

• ind2sub ............................ Linear index to subscript

Description
[i1,i2,...]=ind2sub(sz,index) returns the equivalent subscripts of the linear index for an array whose size is sz.

See Also
sub2ind

• ipermute  ............ Inverse permute of matrix dimensions

Description
y=ipermute(x,order) undoes the function of x=permute(y,order).
See Also
permute,reshape

- **length** ............................. Length of matrix

  Description
  
y=length(x) is the largest dimension of x.

  See Also
  blanks,size

- **ndims** ............................. Number of dimensions

  Description
  
d=ndims(x) is the number of dimensions of x.

  See Also
  size

- **permute** ......................... Permute matrix dimensions

  Description
  
y=permute(x,order) permutes the matrix dimensions based on order, which contains the integer numbers 1:ndims(x).
See Also
ipermute, reshape

• repmat .................................. Replicate matrix

Description

$y = \text{repmat}(x, m, n)$ returns a big matrix $y$, composed of $m \cdot n$ replicated copies of $x$.

See Also
cat, horzcat, rot90, transpose, vertcat

• reshape .................................. Reshape matrix

Description

$y = \text{reshape}(x, m, n)$ returns $x$ reshaped to $m$ by $n$. The new reshaped size must be same as the original size of $x$.

Also, $\text{reshape}(x, [m, n]) = \text{reshape}(x, m, n)$.

See Also
ipermute, permute, size, squeeze

• rot90 .................................. Rotate matrix

Description

$z = \text{rot90}(x, y)$ returns $x$ rotated by $90 \cdot y$ degrees counter-clockwise. The default value for the integer variable $y$ is 1.
See Also
cat,ctranspose,flipdim,flipl,flipud,horzcat,repmat,transpose,vertcat

- **rows** .......................... Number of rows

  Description
  
y=rows(x) returns the number of rows in x.

  See Also
columns,is_square,is_vector,size

- **size** .......................... Matrix dimensions

  Description
  
y=size(x) is a row vector, composed of x dimensions.
y=size(x,n) returns the n-th dimension of x.
[m,n]=size(x) returns the number of x rows in m and the number of x columns in n.

  See Also
eye,flipdim isempty,length,ndims,nnz,ones,reshape,rows,squeeze

- **squeeze** ........................ Squeeze dimensions

  Description
  
y=squeeze(x) removes singleton dimensions from the matrix x. If x was a row vector, y=x.
See Also
reshape, size

- **sub2ind** ....................... Subscript to linear index

Description

\[
\text{index} = \text{sub2ind}(sz, i_1, i_2, \ldots) \]

returns the equivalent linear index for when \( i_1, i_2, \ldots \) index an array whose size is \( sz \).

See Also
find, ind2sub
19 Statistical and random functions

- **erf** .......................................................... Error function
  
  **Description**
  
y = erf(x) is the error function of x, defined as
  
  \[ erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt \]

  **See Also**
  
erfc, erfcx, erfinv, expint

- **erfc** .......................... Complementary error function
  
  **Description**
  
y = erfc(x) is the complementary error function of x, defined as
  
  \[ erfc(x) = 1 - erf(x) \]

  **See Also**
  
erf, erfcx, erfinv

- **erfcx** ................. Scaled complementary error function
  
  **Description**
  
y = erfcx(x) is the scaled complementary error function of x, defined as
  
  \[ erfcx(x) = erfc(x) \cdot e^{x^2} \]
See Also
erf, erfc, erfinv

- \textbf{erfinv} \hspace{1cm} \textbf{Inverse error function}

\textbf{Description}

\( y = \text{erfinv}(x) \) is the inverse error function of \( x \), defined by

\[
\text{erf}(\text{erfinv}(x)) = x
\]

See Also
erf, erfc, erfcx

- \textbf{mean} \hspace{1cm} \textbf{Average value}

\textbf{Description}

\( y = \text{mean}(x) \) returns the average of \( x \) values along a dimension.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{mean}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

See Also
corrcoef, cov, median, rand, randn, std
• **median** ............................ Median value

**Description**

\( y = \text{median}(x) \) returns the median of \( x \) values along a dimension. If \( x \) contains even number of elements, the average of the two center values is returned.

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{median}(..., \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

**See Also**

corrcoef, cov, mean, std

• **nchoosek** ............................. Combinations

**Description**

\( c = \text{nchoosek}(n, k) \) is the binomial coefficient

\[
C(n, k) = \binom{n}{k} = \frac{n!}{k!(n - k)!}
\]

If \( n \) is a vector, \text{nchoosek} returns all the \( k \)-element combinations from the vector.

**See Also**

perms

• **perms** .............................. Permutations

**Description**

\( p = \text{perms}(v) \) returns all the permutations of \( v \). The number of the permutations is \( n! \) where \( n = \text{length}(v) \).
See Also
nchoosek

- **rand** ............. Random numbers : uniform distribution

  Description

  \[ y = \text{rand}(m,n) \]
  is a \( m \)-by-\( n \) random matrix, composed of random numbers.
  Also, \( \text{rand}(n) = \text{rand}(n,n) \) and \( \text{rand}([m \ n]) = \text{rand}(m,n) \).
  \[ z = \text{rand}('\text{state}') \]
  returns the random number generator state in \( z \).
  \[ \text{rand}('\text{state}',z) \]
  sets it.
  \( \text{rand} \) is named \( \text{randm} \) in Visual Basic programs.

  See Also

  mean,randM,randn,randperm,sprand,std

- **randm** ............. Random numbers : uniform distribution

  Description

  \( \text{rand} \) is named \( \text{randm} \) in Visual Basic programs.

  See Also

  rand
• **randn** ............... Random numbers : normal distribution

**Description**

\[ y = \text{randn}(m,n) \]

is a \( m \)-by-\( n \) random matrix, composed of random numbers. Also, \( \text{randn}(n) = \text{randn}(n,n) \) and \( \text{randn}([m \ n]) = \text{randn}(m,n) \).

\[ z = \text{randn('state')} \]

returns the random number generator state in \( z \); \( \text{randn('state',z)} \) sets it.

**See Also**

mean, rand, sprandn, std

• **randperm** ......................... Random permutation

**Description**

\[ y = \text{randperm}(n) \]

returns a random permutation of the integer numbers 1 to \( n \),

**See Also**

rand

• **std** .......................... Standard deviation

**Description**

\[ y = \text{std}(x,\text{type}) \]

returns the standard deviation of \( x \) values along a dimension. If \( \text{type} \) is 0, the result is scaled by \( n-1 \). If \( \text{type} \) is 1, the result is scaled by \( n \).

The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{std}(\ldots,\text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

\text{std} is named \text{stdM} in Visual Basic programs.
### See Also

`corrcoef`, `cov`, `mean`, `median`, `rand`, `randn`, `stdM`

- **stdM** .......................... Standard deviation

### Description

`std` is named `stdM` in Visual Basic programs.

### See Also

`std`
20 Special functions

• **airy** ............................... Airy function

  **Description**

airy(x)=airy(0,x) is the Airy function, a solution to the differential equation

\[ \frac{\delta^2 f}{\delta Z^2} = Z \cdot f \]

airy(1,x) is the Airy function’s derivative.
airy(2,x) is the other solution to the equation.
airy(3,x) is the derivative of the other solution.

An optional second output argument, ierr, signals error conditions. On success, ierr=0. Upon failure, positive ierr value indicates a problem.

  **See Also**
besselh,besseli,besselj,besselk,bessely

• **bessel** ............................. Bessel function of the first kind

  **Description**

bessel is identical to besselj.

  **See Also**
besselj
MatrixVB Function Reference Guide

- **bessela**  ............... Bessel function of the first kind

  **Description**
  
  bessela is identical to besselj.

  **See Also**
  
  besselj

- **besselh**  .......................... Bessel functions

  **Description**
  
  y=besselh(nu,k,z) is the Bessel function of the third kind. k=1 or 2 returns one of the two solutions.

  besselh(nu,z) is a shorthand for besselh(nu,1,z).

  Calling besselh with an optional argument 1 causes the result to be scaled by the factor $e^{-iz}$ or $e^{iz}$ for k=1, 2 respectively.

  An optional second output argument, ierr, signals error conditions. On success, ierr=0. Upon failure, positive ierr value indicates a problem.

  **See Also**
  
  airy,besseli,besselj,besselk,bessely

- **besseli**  ............ Modified Bessel function of the first kind

  **Description**
  
  y=besseli(nu,z) is the modified Bessel function of the first kind, solution to the differential equation
  
  $$z^2 \frac{\delta^2 f}{\delta z^2} + z \frac{\delta f}{\delta z} - (z^2 + nu^2)f = 0$$
Calling the function with an optional argument 1 causes the result to be scaled by the factor $e^{-iz}$.

An optional second output argument, ierr, signals error conditions. On success, ierr=0. Upon failure, positive ierr value indicates a problem.

**See Also**
airy, besselh, besselj, besselk, bessely

• **besselj** ................. Bessel function of the first kind

**Description**

$y=besselj(nu,z)$ is the Bessel function of the first kind, solution to the differential equation

$$z^2 \frac{\delta^2 f}{\delta z^2} + z \frac{\delta f}{\delta z} + (z^2 - nu^2)f = 0$$

An optional second output argument, ierr, signals error conditions. On success, ierr=0. Upon failure, positive ierr value indicates a problem.

**See Also**
airy, bessel, bessela, besselh, besseli, besselk, bessely

• **besselk** ............ Modified Bessel function of the second kind

**Description**

$y=besselk(nu,z)$ is the modified Bessel function of the second kind, solution to the differential equation

$$z^2 \frac{\delta^2 f}{\delta z^2} + z \frac{\delta f}{\delta z} - (z^2 + nu^2)f = 0$$

Calling the function with an optional argument 1 causes the result to be scaled by the factor $e^{-iz}$.
An optional second output argument, ierr, signals error conditions. On success, ierr=0. Upon failure, positive ierr value indicates a problem.

See Also
airy, besselh, besseli, besselj, bessely

• bessely ............... Bessel function of the second kind

Description
y=bessely(nu,z) is the Bessel function of the second kind, solution to the differential equation

\[ z^2 \frac{\delta^2 f}{\delta z^2} + z \frac{\delta f}{\delta z} + (z^2 - nu^2)f = 0 \]

An optional second output argument, ierr, signals error conditions. On success, ierr=0. Upon failure, positive ierr value indicates a problem.

See Also
airy, besselh, besseli, besselj, besselk

• beta ................................. Beta function

Description
y=beta(z,w) is the beta function,

\[ \beta(z,w) = \int_0^1 t^{z-1}(1-t)^{w-1} dt. \]
See Also
betainc,betaln

• betainc ......................... Incomplete beta function

Description
\[ y = \text{betainc}(x, z, w) \] is the incomplete beta function,  
\[ \text{betainc}(x, z, w) = \frac{1}{\text{beta}(z, w)} \int_0^x t^{z-1}(1-t)^{w-1} \, dt, \]
where 0 ≤ x ≤ 1.

See Also
beta,betaln

• betaln ........................ Ln beta function

Description
\[ \text{betaln}(z, w) \] returns the natural logarithm of the beta function,  
\[ \text{betaln}(z, w) = \ln(\text{beta}(z, w)). \]
The computation is done directly, avoiding overflow or underflow problems.

See Also
beta,betainc
• **cart2pol** .......................... Cartesian to Polar

**Description**

\([\theta, \rho] = \text{cart2pol}(x, y)\) returns the transformation of the cartesian coordinates \((x, y)\) to polar coordinates \((\theta, \rho)\) according to the formulas below.

\([\theta, \rho, Z] = \text{cart2pol}(x, y, z)\) returns the transformation of the cartesian coordinates \((x, y, z)\) to cylindrical coordinates \((\theta, \rho, Z)\) according to the formulas

\[
\theta = \text{atan2}(y, x) \\
\rho = \sqrt{x^2 + y^2} \\
Z = z
\]

**See Also**

angle, atan2, cart2sph, pol2cart, polar, sph2cart

• **cart2sph** .......................... Cartesian to spherical

**Description**

\([\theta, \phi, r] = \text{cart2sph}(x, y, z)\) returns the transformation of the cartesian coordinates \((x, y, z)\) to spherical coordinates \((\theta, \phi, r)\) according to the formulas

\[
\theta = \text{atan2}(y, x) \\
\phi = \text{atan2}(z, \sqrt{x^2 + y^2}) \\
r = \sqrt{x^2 + y^2 + z^2}
\]

**See Also**

angle, atan2, cart2pol, pol2cart, sph2cart
• ellipj .......................... Jacobian elliptic functions

Description

[sn,cn,dn]=ellipj(u,m) evaluates the Jacobian elliptic functions where 0 ≤ m ≤ 1 and u is real.

See Also

ellipke

• ellipke .......................... Complete elliptic integrals

Description

[k,e]=ellipke(m) returns the complete elliptic integrals of the first and second kind in k and e respectively. They are defined by the integrals

\[
\begin{align*}
  k(m) &= \int_0^{\pi/2} \frac{dt}{\sqrt{1 - m \sin(t)^2}} \\
  e(m) &= \int_0^{\pi/2} \sqrt{1 - m \sin(t)^2} dt
\end{align*}
\]

See Also

ellipj

• expint ............................. Exponential integral

Description

y=expint(x) returns the exponential integral of x,

\[
\text{expint}(x) = \int_x^\infty \frac{e^{-t}}{t} dt
\]
See Also
erf, erfc

- **factor**  .............................................. Factorize

Description

`y = factor(x)` returns a list of `x` prime factors. The factorization is done with a brute-force approach. Better algorithms exist for large `x`.

See Also

isprime, lcm, primes

- **gamma**  ............................................. Gamma function

Description

`z = gamma(y)` is the gamma function of `y`, defined as

\[
\gamma(y) = \int_0^\infty e^{-t}t^{y-1}dt
\]

See Also

gammainc, gammaln
• **gammainc** ..................Incomplete gamma function

Description

\[ y = \text{gammainc}(x, y) \]

is the incomplete gamma function of \( x \) and \( y \), defined as

\[ \text{gammainc}(x, y) = \frac{1}{\gamma(y)} \int_0^x e^{-t} t^{y-1} dt \]

See Also

gamma, gammaln

• **gammaln** ......................Ln of the gamma function

Description

\[ z = \text{gammaln}(y) \]

is the natural logarithm of the gamma function,

\[ \text{gammaln}(x) = \ln(\gamma(x)) \]

The computation avoids precision problems.

See Also

gamma, gammainc

• **gcd** ............................Greatest common divisor

Description

\[ g = \text{gcd}(x, y) \]

is the greatest common divisor of \( x \) and \( y \).

\[ [g, c, d] = \text{gcd}(x, y) \]

also return \( c, d \) integers such that

\[ x \cdot c + y \cdot d = g. \]
See Also
isprime,lcm,primes

• lcm .......................... Least common multiply

Description
z=lcm(x,y) is the smallest integer number divisible by integers x and y.

See Also
factor,gcd,primes

• pol2cart .......................... Polar to Cartesian

Description
[x,y]=pol2cart(θ,ρ) returns the transformation of the 2-dimensional polar coordinates (θ,ρ) to cartesian coordinates (x,y) according to the formulas below.

[x,y,z]=pol2cart(θ,ρ,Z) returns the transformation of the cylindrical coordinates (θ,ρ,Z) to cartesian coordinates (x,y,z) according to the formulas

\[ x = ρ \cdot \cos(θ) \]
\[ y = ρ \cdot \sin(θ) \]
\[ Z = z \]

See Also
angle,atan2,cart2pol,cart2sph,polar,sph2cart
• **primes**  ........................................Prime list

**Description**
y=primes(x) returns the a list of the prime numbers less or equal than x.

**See Also**
factor,gcd,isprime,lcm

• **rat**  ...........................................Rational approximation

**Description**
[n,d]=rat(x,tol) returns a rational approximation to the floating point number x. The approximation accuracy is tol, whose default value is $1e-6 \cdot \text{sum(abs(x(:)))}$.

**See Also**
disp

• **sph2cart**  .................................Spherical to Cartesian

**Description**
[x,y,z]=sph2cart(θ,φ,r) returns the transformation of spherical coordinates (θ,φ,r) to the cartesian coordinates (x,y,z) according to the formulas

\[
x = r \cdot \cos(\phi) \cdot \cos(\theta) \\
y = r \cdot \cos(\phi) \cdot \sin(\theta) \\
z = r \cdot \sin(\phi)
\]
See Also

angle, atan2, cart2pol, cart2sph, pol2cart
21 Special matrices

- **cauchy** .......................... Cauchy matrix

  **Description**

  \[ z = \text{cauchy}(x, y) \] where \( x \) and \( y \) are vectors of the same length, returns the square matrix \( z \). Each element is computed according to

  \[
  z(i, j) = \frac{1}{x(i) + y(j)}
  \]

  The dimension can be explicitly given as an optional argument, after all other input arguments, as in \( y = \text{cat}(\ldots, \text{dimension}) \). If it is not specified, the default dimension is 1 for row vectors and 2 otherwise.

  **See Also**

  pascal, wilkinson

- **hadamard** .......................... Hadamard matrix

  **Description**

  \( y = \text{hadamard}(n) \) returns the Hadamard matrix of order \( n \). Hadamard matrices, \( H_n \) are composed of 1 and -1 only, and \( \text{inv}(H_n) = \frac{1}{n} H' \).

  **See Also**

  fft
• **hankel** .......................... Hankel matrix

**Description**

\[ z = \text{hankel}(x, y) \]

is the Hankel matrix with \( x \) as first column and \( y \) as last row. If \( y \) is omitted, the elements before the main anti-diagonal are zero.

**See Also**

toeplitz

• **hilb** .......................... Hilbert matrix

**Description**

\[ y = \text{hilb}(n) \]

is the Hilbert matrix of order \( n \), defined by

\[ y_{i,j} = \frac{1}{i + j - 1} \]

**See Also**

invhilb

• **invhilb** .......................... Inverse Hilbert matrix

**Description**

\[ y = \text{invhilb}(n) \]

is the inverse Hilbert matrix of order \( n \). This function is more accurate than computing \( \text{inv(hilb(n))} \) since the Hilbert matrix is ill-conditioned.
See Also

hilb, inv

- **ones** .................................................. Ones matrix

**Description**
y = ones(m, n) is a m-by-n ones matrix, composed of ones.
Also, ones(n) = ones(n, n) and ones([m n]) = ones(m, n).

See Also

cell, eye, size, spones, zeros

- **pascal** .................................................. Pascal matrix

**Description**
y = pascal(n) returns the Pascal triangle of integer order n.
pascal is named pascalM in Visual Basic programs.

See Also

cauchy, magic, pascalM, vander, wilkinson

- **pascalM** .................................................. Pascal matrix

**Description**
pascal is named pascalM in Visual Basic programs.
See Also
pascal

- rosser ..................................... Rosser matrix

Description
y=rosser returns the Rosser matrix.

See Also
eig

toeplitz ...................................... Toeplitz matrix

Description
z=toeplitz(x,y) is the Toeplitz matrix with x as first column and y as first row. If y is omitted, y=x, and the matrix is symmetric.

See Also
hankel

vander ...................................... Vandermonde matrix

Description
v=vander(c) is the Vandermonde matrix. The columns of v are powers of c, with c^0 as the rightmost column and c^{n-1} as the leftmost column.
See Also

pascal

- **zeros** .................................................. Zeros matrix

Description

\( y = \text{zeros}(m,n) \) is a \( m \)-by-\( n \) zeros matrix, composed of zeros.

Also, \( \text{zeros}(n) = \text{zeros}(n,n) \) and \( \text{zeros}([m\ n]) = \text{zeros}(m,n) \).

See Also

cell, eye, isempty, ones, size
22 String functions

- **base2dec** .......................... Convert number base

  **Description**

  \( y = \text{base2dec}(s, n) \) returns the base-10 representation of the number in string \( s \), which contains a base-\( n \) number.

  **See Also**

  bin2dec, dec2base, dec2bin, dec2hex, hex2dec, hex2num

- **bin2dec** .......................... Convert number base

  **Description**

  \( y = \text{bin2dec}(s) \) returns the base-10 representation of the number in string \( s \), which contains a base-2 number.

  **See Also**

  base2dec, dec2base, dec2bin, hex2dec

- **blanks** .......................... Make blank string

  **Description**

  \( \text{blanks}(n) \) returns a string composed of \( n \) blanks.

  **See Also**

  deblank, disp, length, printf, strjust
• **cellstr** ................................. Make cell array of strings

**Description**

\[ y = \text{cellstr}(x) \]

returns a cell array whose size is the number of rows of \( x \). Each cell of \( y \) contains one string line from the array \( x \).

**See Also**

char, isa, iscellstr, ischar

• **char** ................................. Create string

**Description**

\[ y = \text{char}(x) \]

translates each element of \( x \) to its ASCII character.

\[ y = \text{char}(x_1, x_2, x_3, \ldots) \]

returns a string matrix containing the string lines \( x_1, x_2, \ldots \).

char is named mchar in Visual Basic programs.

**See Also**

cellstr, double, ischar, isletter, isnumeric, isspace, mchar, setstr, strcat, strvcat, uint8

• **deblank** ............................... Erase trailing blanks

**Description**

\[ y = \text{deblank}(x) \]

is the string \( x \) without its trailing blanks. If \( x \) is a cell string, deblank operates on every line.

**See Also**

blanks, sprintf, strcat, strjust
• **dec2base** .......................... Convert number base

**Description**

$s=\text{dec2base}(x, n)$ returns the base-$n$ string representation of the base-10 number $x$.

**See Also**

base2dec, bin2dec, dec2bin, dec2hex, hex2dec

• **dec2bin** .......................... Convert to binary

**Description**

$s=\text{dec2bin}(x)$ returns the base-2 string representation of the base-10 number $x$.

**See Also**

base2dec, bin2dec, dec2base, dec2hex, hex2dec

• **dec2hex** .......................... Convert to hexadecimal

**Description**

$s=\text{dec2hex}(x)$ returns the base-16 string representation of the base-10 number $x$. The letters A-F are used to represent the numbers 10-15.

**See Also**

base2dec, dec2base, dec2bin, hex2dec
• **findstr** .......................... Search for substring

**Description**

\[ z = \text{find}(x, y) \] returns the indices of any appearance of \( x \) within \( y \) if \( x \) is shorter, or \( y \) within \( x \) if \( y \) is shorter. If no matches are found \texttt{findstr} returns an empty matrix.

**See Also**

find, strcmp, strmatch, strncmp, strtok

• **hex2dec** .......................... Convert number base

**Description**

\[ y = \text{hex2dec}(s) \] returns the base-10 representation of the number in string \( s \), which contains a base-16 (hex) number.

**See Also**

base2dec, bin2dec, dec2base, dec2bin, dec2hex

• **hex2num** .......................... Construct IEEE-754 float

**Description**

\[ y = \text{hex2num}(s) \] returns the IEEE-754 float number described by the hex string \( x \).

**See Also**

base2dec, sprintf
- **int2str** ........................................... Convert integer to string
  
  **Description**
  
y=int2str(x) returns the string representation of the integer number x.

  **See Also**
  
disp,num2str,sprintf,sscanf

- **lower** ........................................... Lower case
  
  **Description**
  
y=lower(x) returns string x, converted to lower-case.

  **See Also**
  
islower,isupper,upper

- **mat2str** ........................................... Matrix to string
  
  **Description**
  
s=mat2str(x) produces a string out of the matrix x, which when eval-ed will result in x.

  **See Also**
  
num2str
• **mchar** ........................................... Convert to string

**Description**

char is named mchar in Visual Basic programs.

**See Also**

char

• **num2str** ........................................... Number to string

**Description**

\[ y = \text{num2str}(x, \text{precision}) \]

returns the string representation of the number \( x \) with \( \text{precision} \) digits. The default precision is 4 digits.

**See Also**

disp,input,int2str,isnumeric,str2num

• **setstr** ........................................... Create string

**Description**

\[ y = \text{setstr}(x) \]

returns \( x \) with the string flag on. ASCII values are converted to the corresponding characters.

**See Also**

abs,char
• **sprintf**  ...............Send formatted output to variable

**Description**

\[ y = \text{sprintf}(\text{format}, \ldots) \]

formats the variables specified according to the format string and sends the text output to the variable.

Documentation for the format string can be found in the C library documentation for the `printf` command.

**See Also**

debblank, disp, fclose, fopen, fprintf, hex2num, int2str, printf

• **sscanf**  ...............Get formatted input from string

**Description**

\[ y = \text{sscanf}(\text{s}, \text{format}, \text{maxelem}) \]

returns the values, read from string \( s \), according to the format string.

\( \text{maxelem} \) may be \( \text{Inf} \), in which case all the file is read, an integer scalar or \([m n]\) where the elements read are stored in a \( m \)-by-\( n \) matrix in column-major filling order.

Documentation for the format string can be found in the C library documentation for the `scanf` command.

**See Also**

close, fopen, fscanf, input, int2str

• **str2double**  ......................String to number

**Description**

\[ y = \text{str2double}(x) \]

returns the number contained in string \( x \).
See Also
num2str,str2num

- **str2mat** .......................... Strings to string matrix
  Description
  s=str2mat(a1,a2,...) returns a string matrix composed of a1, a2,... as rows.
  See Also
  horzcat,strcat

- **str2num** .......................... String to number
  Description
  y=str2num(x,precision) returns the number contained in string x.
  See Also
  disp,input,isnumeric,num2str

- **strcat** ......................... String horizontal concatenation
  Description
  y=strcat(a1,a2,...) returns an horizontal concatenation of the string arguments.
See Also
char, deblank, horzcat, str2mat, strjust, strvcat, vertcat

• `strcmp` .................................................. Compare strings

Description
`z = strcmp(x, y)` returns 1 if the strings `x` and `y` are identical and 0 otherwise.

See Also
findstr, strcmpi, strmatch, strncmp, strtok

• `strcmpi` .......................... Compare strings insensitive

Description
`z = strcmpi(x, y)` returns 1 if the strings `x` and `y` are identical, except for case and 0 otherwise.

See Also
strcmp, strncmp

• `strjust` .......................... Right justify a string

Description
`y = strjust(x, how)` returns a modified version of the string matrix `x` where all rows are justified according to `how`, which may be 'left' (default), 'right' or 'center'.
See Also
blanks, debank, strcat

• **strmatch** .......................... Match strings

Description

`z=strmatch(x,y)` return the indices of `y` rows where `x` appear. An optional input argument, 'exact' causes `strmatch` to return only those lines which are identical to `x`.

See Also
findstr, strcmp, strncmp, strrep

• **strncmp** ......................... Partial compare strings

Description

`z=strncmp(x,y,n)` returns 1 if the first `n` characters of strings `x` and `y` are identical and 0 otherwise.

See Also
findstr, strcmp, strncmpi, strmatch, strncmpi

• **strncmpi** ........................ Partial compare strings insensitive

Description

`z=strncmpi(x,y,n)` returns 1 if the first `n` characters of strings `x` and `y` are identical except for case and 0 otherwise.
See Also
strncmp

- **strrep** ................................. Search and replace

Description

\[ y = \text{strrep}(x, \text{from}, \text{to}) \]

returns a copy of \( x \), in which all occurrences of the string \( \text{from} \) had been replaced by the string \( \text{to} \).

See Also
find, strmatch

- **strtok** ................................. Tokenize string

Description

\[ [\text{before}, \text{after}] = \text{strtok}(x, y) \]

return the characters before the first appearance of one of \( y \) value in \( \text{before} \). The rest of characters are returned in \( \text{after} \). Default value for \( y \) is the whitespace characters. \( \text{after} \) is optional.

See Also
findstr, isspace, strcmp

- **strvcat** ......................... String vertical concatenation

Description

\[ y = \text{strcat}(a_1, a_2, \ldots) \]

returns a vertical concatenation of its string arguments.
See Also
char, horzcat, strcat, vertcat

- **upper** ......................................... Upper case

Description

\[ y = \text{upper}(x) \] returns string \( x \), converted to upper-case.

See Also

islower, isupper, lower
23 Trigonometric functions

- **acos** ................................. Inverse cosine

  **Description**
  
  acos(x) is the inverse cosine of x. If x is complex, or outside the range \([-1, 1]\), a complex result is returned.

  **See Also**
  
  acosh,asin,atan,cos

- **acosh** ......................... Inverse hyperbolic cosine

  **Description**
  
  acosh(x) is the inverse hyperbolic cosine of x.

  **See Also**
  
  acos,cosh

- **acot** .......................... Inverse cotangent

  **Description**
  
  acot(x) is the inverse cotangent of x,

  \[ acot(x) = atan(1/x). \]
See Also
atan,cot,tan

• **acoth** .................... *Inverse hyperbolic cotangent*

  **Description**

  \( \text{acoth}(x) \) is the inverse hyperbolic cotangent of \( x \),

  \[
  \text{acoth}(x) = \text{atanh}(1/x).
  \]

  **See Also**

  atanh,coth,tanh

• **acsc** .......................... *Inverse cosecant*

  **Description**

  \( \text{acsc}(x) \) is the inverse cosecant of \( x \),

  \[
  \text{acsc}(x) = \text{asin}(1/x).
  \]

  **See Also**

  asin,csc,sin
• **acsch** .......................... Inverse hyperbolic cosecant

**Description**

`acsch(x)` is the inverse hyperbolic cosecant of `x`,

\[ \text{acsch}(x) = \text{asinh}(1/x). \]

**See Also**

`asinh,csch,sinh`

• **asec** ............................ Inverse secant

**Description**

`y=asec(x)` is the inverse secant of `x`,

\[ \text{asec}(x) = \text{acos}(1/x). \]

**See Also**

`asech,cos,sec`

• **asech** ......................... Inverse hyperbolic secant

**Description**

`y=asech(x)` is the inverse hyperbolic secant of `x`,

\[ \text{asech}(x) = \text{acosh}(1/x). \]
See Also
asec,cosh,sech

• **asin** .............................. Inverse sine

Description

asin(x) is the inverse sine of x,

\[ asin(x) = -i \cdot \log(i \cdot x + \sqrt{1 - x^2}) . \]

If x is complex, or outside the range \([-1, 1]\), a complex result is returned.

See Also
acos,acsc,asinh,sin

• **asinh** ............................ Inverse hyperbolic sine

Description

asinh(x) is the inverse hyperbolic sine of x,

\[ asinh(x) = \log(x + \sqrt{1 + x^2}) . \]

See Also
acsch,asin,sinh
• **atan** ................................. Inverse tangent

**Description**

`atan(x)` is the inverse tangent of `x`,

\[
atan(x) = \frac{i}{2} \log \frac{i + x}{i - x}.
\]

**See Also**

`acos`, `acot`, `atan2`, `atanh`, `tan`

• **atan2** ................................. Inverse tangent

**Description**

`atan(y,x)` is the inverse tangent of `y/x`, with the result positioned in the correct circle quadrant.

**See Also**

`atan`, `cart2pol`, `cart2sph`, `pol2cart`, `sph2cart`, `tan`

• **atanh** ............................... Inverse hyperbolic tangent

**Description**

`atanh(x)` is the inverse hyperbolic tangent of `x`,

\[
atanh(x) = \frac{1}{2} \log \frac{1 + x}{1 - x}.
\]
See Also
acoth,atan,tan,tanh

• **Cos** .........................................................Cosine

Description
\[ y = \cos(x) \] is the cosine of \( x \). If \( x \) is complex,
\[ \cos(x) = \cos(\text{real}(x)) \cdot \cos(\text{imag}(x)) - i \cdot \sin(\text{real}(x)) \cdot \sin(\text{imag}(x)) \]

\( \cos \) is named \( mcos \) in Visual Basic programs.

See Also
acos,asec,cosh,cot,mcos,sec,sin,subspace,tan

• **cosh** ............................Hyperbolic cosine

Description
\[ y = \cosh(x) \] is the hyperbolic cosine of \( x \),
\[ \cosh(x) = \frac{1}{2}(e^x + e^{-x}) \]

See Also
acosh,asech,cos,sech,sinh
• **cot** ......................................................... Cotangent

**Description**

\( y = \cos(x) \) is the cotangent of \( x \),

\[
cot(x) = \frac{1}{\tan(x)}
\]

**See Also**

acot, cos, coth, tan

• **coth** .............................. Hyperbolic cotangent

**Description**

\( y = \cosh(x) \) is the hyperbolic cotangent of \( x \),

\[
coth(x) = \frac{1}{\tanh(x)}
\]

**See Also**

acoth, cot, tanh

• **csc** ................................. Cosecant

**Description**

\( y = \csc(x) \) is the cosecant of \( x \),

\[
csc(x) = \frac{1}{\sin(x)}
\]
See Also
acsc,sin

• **csch** ....................................... Hyperbolic cosecant

Description

\[ y = \text{csch}(x) \] is the hyperbolic cosecant of \( x \),

\[ \text{csch}(x) = \frac{1}{\sinh(x)} \]

See Also
acsch,sinh

• **mcos** ............................................ Cosine

Description

\( \cos \) is named **mcos** in Visual Basic programs.

See Also
\( \cos \)
• **msin** .......................... Sine

  **Description**

  $\sin$ is named **msin** in Visual Basic programs.

  **See Also**

  $\sin$

• **mtan**  ......................... Tangent

  **Description**

  $\tan$ is named **mtan** in Visual Basic programs.

  **See Also**

  $\tan$

• **sec**  ............................ Secant

  **Description**

  $y=\sec(x)$ is the secant of $x$,

  \[
  \sec(x) = 1/\cos(x).
  \]

  **See Also**

  asec, cos, sech
• **sech** .................................. Hyperbolic secant

**Description**

\[ y = sech(x) \] is the hyperbolic secant of \( x \),

\[ sech(x) = \frac{1}{cosh(x)}. \]

**See Also**

asech, cosh, sec

• **sin** ............................................. Sine

**Description**

\[ y = \sin(x) \] is the sine of \( x \). If \( x \) is complex,

\[ \sin(x) = \sin(real(x)) \cdot \cos(imag(x)) + i \cdot \cos(real(x)) \cdot \sin(imag(x)) \]

\( \sin \) is named \texttt{msin} in Visual Basic programs.

**See Also**

acsc, asin, cos, csc, msin, sinc, sinh

• **sinh** ........................................ Hyperbolic sine

**Description**

\[ y = \sinh(x) \] is the hyperbolic sine of \( x \),

\[ \sinh(x) = \frac{1}{2}(e^x - e^{-x}) \]
See Also
acsch, asinh, cosh, csch, sin

- **tan** .................................................... Tangent

**Description**

\[ \tan(x) = \frac{\sin(x)}{\cos(x)}. \]

\( \tan \) is named \texttt{mtan} in Visual Basic programs.

See Also
acot, atan, atan2, atanh, cos, cot, mtan, tanh

- **tanh** .................................Hyperbolic tangent

**Description**

\[ \tanh(x) = \frac{\sinh(x)}{\cosh(x)}. \]

See Also
acoth, atanh, coth, tan
24  Time and date functions

- **clock** ........................................... Time clock
  
  **Description**
  
y=clock returns the realtime clock value.
y contains six values: [year month day hour minute second]
clock is named mclock in Visual Basic programs.

  **See Also**
clockM,cputime,etime,mclock,tic,toc

- **date** ........................................... Current date
  
  **Description**
  
date returns the current date, such as ’29-May-1999’.

  **See Also**
clock,etime,now

- **datenum** ....................................... Serial date
  
  **Description**
  
j=datenum(s) returns a serial date corresponding to the string s.
j=datenum(year,month,day,hour,minute,second) return the serial date corresponding to the input date. The last three inputs can be omitted, assumed zeros. The serial date 1 is the result of the date ’1-Jan-0000’.
See Also
date, now, tic, toc

- **datevec** .......................... Decompose Serial date

  **Description**
  
c = datevec(j) decomposes the serial date j into its six components in the vector c.

  **See Also**
clock, datenum

- **eomday** .......................... Last day of month

  **Description**
  
d = eomday(year, month) is the last day of month month in year year.

  **See Also**
clock, datevec

- **etime** .......................... Elapsed time

  **Description**
  
dt = etime(t1, t2) returns the elapsed time between t1 and t2. Both arguments should be the results of the clock function.
See Also
clock, disp, tic, toc

- **mclock** ................................. Time clock
  Description
clock is named mclock in Visual Basic programs.
  See Also
clock

- **now** ................................. Current time
  Description
c=now is the current time and date.
  See Also
clock, datenum, etime

- **tic** ................................. Start stopwatch
  Description
tic start a global stopwatch.
See Also
clock, etime, toc

- **toc** .......................... Stop stopwatch

Description
toc stops a global stopwatch and displays the elapsed time, which may be assigned to a variable: 
\[ dt = toc. \]

See Also
clock, etime, tic

- **weekday** ........................ Weekday from serial date

Description
d=weekday(j) returns the weekday of the serial date j. Weekdays are counted from Sunday=1.

See Also
clock, date, datenum, datevec, eomday
25 Two-dimensional plots

Under Unix and Linux, only a subset of the graphics commands is available - see the chapter 'Graphics functions under Unix'.

- **area** .................................................. Area plot

  **Description**

  \[ h=\text{area}(x,y) \] plots an area graph with heights \( y \) at points \( x \). If \( x \) is omitted, the default value is \( 1:\text{length}(y) \). Input parameters are similar to the \texttt{plot} function.

  \( h \) contains the handles for the created line object(s).

  **See Also**

  \texttt{plot}, \texttt{stairs}

- **bar** ................................................. Vertical bar plot

  **Description**

  \[ h=\text{bar}(x,y) \] plots a vertical bar graph with heights \( y \) at points \( x \). If \( x \) is omitted, its default value is \( 1:\text{length}(y) \). Input parameters are similar to the \texttt{plot} function.

  \( h \) contains the handles for the created line object(s).

  **See Also**

  \texttt{bar3}, \texttt{barh}, \texttt{errorbar}, \texttt{hist}, \texttt{plot}
MatrixVB Function Reference Guide

- **barh** .............................. Horizontal bar plot

  Description
  
  $h=\text{barh}(x,y)$ plots horizontal bar graph with length $y$ at vertical points $x$. If $x$ is omitted, its default value is $1:\text{length}(y)$. Input parameters are similar to the plot function.

  $h$ contains the handles for the created line object(s).

  See Also
  
  bar, bar3h, plot

- **compass** .......................... Compass plot

  Description
  
  $\text{compass}(x,y)$ plots a graph with arrows from the (0,0) location towards the (x,y) vectors.

  See Also
  
  rose

- **contour** ............................ Plot contours

  Description
  
  $[cm,hm]=\text{contour}(x,y,z,n)$ plots the contours of the matrix $z$ where $x$ and $y$ define the x- and y- axes, respectively. If $n$, the number of contour levels, is not specified, automatic choice is made.

  $x$ and $y$ are also optional, unit squares are used if they are not given.

  $hm$ contains the handles for the created contour line object(s). $cm$ contains the contour matrix height, which may be very large.
See Also
clabel, contour3, contourf, mesh, meshc, plot3, surf, view

- **contourf** .......................... Filled contour plot

**Description**

`contourf` is a contour plot with filled areas.

See Also
contour

- **errorbar** .......................... Plot with error bars

**Description**

`h=errorbar(x,y,l,u)` plots the curve `(x,y)` together with error bars which are `u` above and `l` below the curve.

`h` contains the handles for the created line object(s).

See Also
bar, plot, quiver, stem

- **feather** .......................... Feather plot

**Description**

`feather(x,y)` plots a graph with horizontal arrows towards the `(x,y)` vectors.
See Also

• hist  ......................................... Plot Histogram

Description

\[[x_n,x_c]=\text{hist}(y)\] calculates a histogram of \(y\) along 10 equally spaced bins. The return values are \(x_n\), the number of items per bin and \(x_c\), the centers of the bins. \(y\) can be a matrix, in which case the histogram is calculated per column vector.

\[[x_n,x_c]=\text{hist}(y,n)\] uses \(n\) bins; if \(n\) is a vector, it is treated as the centers of the beans.

\text{hist}(y)\) without any return arguments plots the histogram bar graph.

See Also

\text{bar,plot}

• loglog  .............................. Log-log plot

Description

\text{loglog} draws a plot graph, but with logarithmic x- and y- axes.

See Also

\text{plot,semilogx,semilogy}

• pie  .................................... Pie chart plot

Description

\text{h=pie}(y)\) plots a pie chart, with the \(y\) representing slices weights.
h contains the handles for the created pie object(s).

**See Also**

pie3, plot
• **plot** .................................................. Linear plot

**Description**

\[ h=\text{plot}(x,y,t) \] plot \( y \) against \( x \) using line type and color \( t \). The color and line type are specified by letters from the tables. If \( t \) is omitted, line type and color are automatically selected.

<table>
<thead>
<tr>
<th>Letter</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>Yellow</td>
</tr>
<tr>
<td>m</td>
<td>Magenta</td>
</tr>
<tr>
<td>c</td>
<td>Cyan</td>
</tr>
<tr>
<td>r</td>
<td>Red</td>
</tr>
<tr>
<td>g</td>
<td>Green</td>
</tr>
<tr>
<td>b</td>
<td>Blue</td>
</tr>
<tr>
<td>w</td>
<td>White</td>
</tr>
<tr>
<td>k</td>
<td>Black</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Letter</th>
<th>Marker/line type</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>Dot mark</td>
</tr>
<tr>
<td>o</td>
<td>Circle mark</td>
</tr>
<tr>
<td>x</td>
<td>X mark</td>
</tr>
<tr>
<td>+</td>
<td>+ mark</td>
</tr>
<tr>
<td>*</td>
<td>* mark</td>
</tr>
<tr>
<td>-</td>
<td>Solid line (default)</td>
</tr>
<tr>
<td>:</td>
<td>Dotted line</td>
</tr>
<tr>
<td>--</td>
<td>Dashed line</td>
</tr>
<tr>
<td>-.</td>
<td>Dash-dot line</td>
</tr>
</tbody>
</table>

\((x,y,s)\) triplets may be repeated several times in one plot to get several curves in the same plot.

\( h \) contains the handles for the created line object(s).

**See Also**

hold, legend, plot3, semilogx, semilogy, subplot, text, title, xlabel, ylabel, zoom
• **plotyy** .......................... Plot with labels

**Description**

`plotyy(x1,y1,x2,y2)` plots two graphs with two vertical axis and two horizontal axis.

**See Also**

plot

• **polar** .......................... Polar plots

**Description**

`h=polar(\theta,r,t)` plots a polar plot of $r(\theta)$ using $t$ line type.

$h$ contains the handles for the created line object(s).

**See Also**

abs, angle, cart2pol, conj, imag, plot, pol2cart, real

• **quiver** .......................... Quiver plot

**Description**

`h=quiver(dx,dy,s)` plots a quiver plot at unit spaced points of line type $s$. An arrow is plotted at each point, pointing to $(dx,dy)$.

$h$ contains the handles for the created line object(s).

**See Also**

errorbar, plot, stem
• **ribbon** .......................... Ribbon plot

  **Description**

  `ribbon(x,y)` plots a line graph where each line is a 3D ribbon.

  **See Also**

• **rose** .......................... Rose plot

  **Description**

  `rose(phi,r)` plots a polar histogram of the values of `phi`. `r`, if exists, can specify either the number of beans of the centers of the beans.

  **See Also**

  `hist`, `polar`

• **scatter** .......................... Scatter X-Y markers plot

  **Description**

  `h=scatter(x,y)` plots a scatter X-Y plot of points.

  `h` contains the handles for the created line object(s).

  **See Also**

  `plot`, `scatter3`
• **semilogx** ........................................Log-linear plot

**Description**

`semilogx` draws a plot graph, but with a logarithmic x axis.

**See Also**

loglog, plot, semilogy

• **semilogy** ........................................Linear-log plot

**Description**

`semilogy` draws a plot graph, but with a logarithmic y axis.

**See Also**

loglog, plot, semilogx

• **spy** ............................................See sparsity structure

**Description**

`spy(x)` shows the sparsity structure of a sparse matrix x in graphic form.

**See Also**

sparse, sprand
**Stairs plot**

**Description**

\( h=\text{stairs}(x,y) \) plots a stair graph with heights \( y \) at points \( x \). If \( x \) is omitted, the default value is \( 1:\text{length}(y) \). Input parameters are similar to the \text{plot} function.

\( h \) contains the handles for the created line object(s).

**See Also**

area, plot

---

**Stem plot**

**Description**

\( h=\text{stem}(x,y) \) plots a horizontal lines graph with heights \( y \) at points \( x \). If \( x \) is omitted, its default value is \( 1:\text{length}(y) \). Input parameters are similar to the \text{plot()} function.

\( h \) contains the handles for the created line object(s).

**See Also**

errorbar, plot, quiver
26 Three-dimensional plots

- **bar3** .......................... 3D vertical bar plot
  
  **Description**
  
  \[ h = \text{bar3}(x,y) \]
  plots a 3D bar graph with heights \( y \) at points \( x \). If \( x \) is omitted, its default value is \( 1:length(y) \). Input parameters are similar to the \texttt{plot} function.

  \( h \) contains the handles for the created line object(s).

  **See Also**
  
  bar, bar3h, plot

- **bar3h** .......................... 3D horizontal bar plot
  
  **Description**
  
  \[ h = \text{bar3h}(x,y) \]
  plots a 3D horizontal bar graph with length \( y \) at vertical positions \( x \). If \( x \) is omitted, its default value is \( 1:length(y) \). Input parameters are similar to the \texttt{plot} function.

  \( h \) contains the handles for the created line object(s).

  **See Also**
  
  bar3, barh, plot

- **contour3** ..................... Plot 3-dimensional contours
  
  **Description**
  
  \[ \text{contour3}(x,y,z,n) \] plots a 3-dimensional contour map, similar to \texttt{contour}. 
See Also

contour, mesh, plot3

• mesh .......................... Draw 3-dimensional mesh

Description

h = mesh(x, y, z) draws a 3-dimensional mesh surface described by (x, y, z). If x and y are omitted, unit spacing is used.

h contains the handles for the created surface object(s).

See Also

automesh, contour, contour3, meshc, meshgrid, plot3, shading, surf, view

• meshc .......................... Draw mesh and contour

Description

meshc draws a contour map and a mesh on the same figure.

h contains the handles for the created surface object(s).

See Also

contour, mesh, meshgrid
• **pcolor** .......................... *Image plot of surface levels*

**Description**

h=pcolor(z) draws an image with z values as the surface levels.

h contains the handles for the created image object(s).

**See Also**

image,surface

• **pie3** .............................. *3D pie chart plot*

**Description**

h=pie3(y) plots a 3D pie chart, with the y representing slices weights.

h contains the handles for the created line object(s).

**See Also**

pie

• **plot3** .............................. *3-dimensional plot*

**Description**

h=plot3(x,y,z,t) plots the 3-dimensional curve (x,y,z) using line style t. See plot for all available markers, lines and colors settings.

h contains the handles for the created line object(s).

**See Also**

contour,contour3,mesh,plot,scatter3,view,zlabel
• **scatter3** ................. 3D plot of scatter X-Y points

**Description**

\[ h = \text{scatter}(x, y, z, t) \]

plots the 3-dimensional \((x, y, z)\) points. See \texttt{plot} for all available markers, lines and colors settings.

\( h \) contains the handles for the created line object(s).

**See Also**

\[ \text{plot, plot3, scatter} \]

• **surf** ......................... 3D filled surface plot  

**Description**

\[ h = \text{surf}(x, y, z) \]

draws a 3-dimensional colored surface described by \((x, y, z)\). If \(x\) and \(y\) are omitted, unit spacing is used.

\( h \) contains the handles for the created surface object(s).

**See Also**

\[ \text{caxis, mesh, rotate3d, shading, surfc} \]

• **surfc** .................. Surface plot with contour below it  

**Description**

\texttt{surfc} draws a surface and a contour below it.

**See Also**

\[ \text{contour, surf} \]
- **surfl** ..................... 3D filled surface plot with light

  **Description**

  `surfl` is exactly as `surf`. Light objects are not supported yet.

  **See Also**

  `surf`
27  Plot customization

•  **axis**  ......................................................... Modify axis ranges

  **Description**

  `axis([xmin xmax ymin ymax])` and `axis([xmin xmax ymin ymax zmin zmax])` set the range of the x-, y- and z- axes to the values supplied. This turns off automatic ranging, which can be turned back on using `axis('manual')`.

  **See Also**

  plot

•  **clf**  .......................................................... Clear figure

  **Description**

  `clf(fig)` clears the figure fig. If fig is omitted, the current figure is cleared.

  **See Also**

  cla,clc,clg,figure

•  **close**  ......................................................... Close window

  **Description**

  `close(h)` closes the figure h or the current figure if h is omitted. `close all` closes all figures. `close` is named `figure_close` in Visual Basic programs.

  **See Also**

  figure,figure_close,hold
**figure** ................................. *Switch or create new window*

**Description**

$h=\text{figure}(n)$ switches the graphic focus to the window $n$. If the window did not exist, a new window is created. $h=\text{figure}$ creates a new figure window.

$h$ contains the handles for the created figure object(s).

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BackgroundColor</td>
<td>Background color of the object. See also the ForegroundColor and Color properties.</td>
<td>$[r ; g ; b]$ or one character color</td>
</tr>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>Function to be called when the mouse button is pressed down while the pointer is over the object.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>Close</td>
<td>Closes the figure or COM port. COM port must be closed after using it.</td>
<td></td>
</tr>
<tr>
<td>CloseReqFcn</td>
<td>Callback function to be performed before closing figures. If the value is 'closereq', the figure immediately closes. The 'close' property can be used to force closing the figure.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Color</td>
<td>Object color.</td>
<td>$[r ; g ; b]$ or one character color</td>
</tr>
<tr>
<td>ColorMap</td>
<td>Object colormap.</td>
<td>Matrix with red, green and blue columns 'on' or 'off' callback string character</td>
</tr>
<tr>
<td>ControlBar</td>
<td>Sets whether to display the control bar of the figure.</td>
<td></td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td></td>
</tr>
<tr>
<td>CurrentCharacter</td>
<td>Value of the last character pressed on the figure.</td>
<td>[x,y]</td>
</tr>
<tr>
<td>CurrentPoint</td>
<td>The last point in which the mouse was clicked on the figure or axes. If on axes, the point is in axis data scales.</td>
<td></td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>DesignMode</td>
<td>Is GUI designer active.</td>
<td>0,1</td>
</tr>
<tr>
<td>Enable</td>
<td>Is object enabled.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>EraseMode</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Extent</td>
<td>Identical to Position.</td>
<td></td>
</tr>
<tr>
<td>FontAngle</td>
<td>Angle of the font.</td>
<td>Normal, italic or oblique string</td>
</tr>
<tr>
<td>FontName</td>
<td>Sets the name of the font.</td>
<td>string</td>
</tr>
<tr>
<td>FontSize</td>
<td>Sets the size of font. The size unit is set by the FontUnits property.</td>
<td>positive integer</td>
</tr>
<tr>
<td>FontUnits</td>
<td>Units for the FontSize property.</td>
<td>'inches', 'centimeters', 'normalized', 'points', 'pixels'</td>
</tr>
<tr>
<td>FontWeight</td>
<td>Weight of font.</td>
<td>'normal', 'light', 'demi', 'bold'</td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Foreground color of the object. For some objects, the foreground color may be the font color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HitTest</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HWND</td>
<td>Read only. Windows handle of the figure.</td>
<td>number</td>
</tr>
<tr>
<td>IconFile</td>
<td>Figure icon file.</td>
<td>legal filename</td>
</tr>
<tr>
<td>Interruptible</td>
<td>Note implemented.</td>
<td></td>
</tr>
<tr>
<td>KeyPressFcn</td>
<td>Action to be performed when a key is pressed on the figure.</td>
<td>callback string</td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>MenuAbout</td>
<td>Visibility of the figure About menu item.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>MenuBar</td>
<td>This property hides the default menu when set to 'none'.</td>
<td>'figure', 'none'</td>
</tr>
<tr>
<td>MouseDownFcn</td>
<td>Action to occur after the mouse is clicked on the object.</td>
<td>callback string</td>
</tr>
<tr>
<td>MoveIntoScreen</td>
<td>Repositions the figure so that it is completely contained inside the screen area.</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Title of figure.</td>
<td>string</td>
</tr>
<tr>
<td>NumberTitle</td>
<td>Figure title is 'Figure NN' if 'on' or string contained in property Name if 'off'.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>OpenGL</td>
<td>Enable OpenGL use for the figure.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>PaperOrientation</td>
<td>Orientation of paper for printing.</td>
<td>'portrait', 'landscape'</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
<tr>
<td>Pointer</td>
<td>Select mouse pointer image.</td>
<td>'arrow', 'watch', 'crosshair', 'top', 'cross', 'ibeam', 'topl'</td>
</tr>
<tr>
<td>Position</td>
<td>Position of the object in terms of the units defined in the Units property.</td>
<td>[left, bottom, width, height]</td>
</tr>
<tr>
<td>Print</td>
<td>Prints the figure.</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next shg if 1.</td>
<td>0, 1</td>
</tr>
<tr>
<td>Resize</td>
<td>Enable resizing the figure.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>ResizeFcn</td>
<td>Function to be called when the figure is resized.</td>
<td>callback string</td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>SetFont</td>
<td>Shows a font selection dialog box.</td>
<td></td>
</tr>
<tr>
<td>Show</td>
<td>Brings the figure into focus.</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
<td>string</td>
</tr>
<tr>
<td>Style</td>
<td>Style of a object. See the uicontrol and hwcontrol functions for available styles.</td>
<td>string</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>TooltipString</td>
<td>Hint that appears when the mouse rests over the object.</td>
<td>string</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td>string</td>
</tr>
<tr>
<td>Units</td>
<td>Units for the Position property. 'Normalized' is relative to the parent size in the range zero to one.</td>
<td>'normalized', 'inches', 'centimeters', 'points', 'characters', 'pixels'</td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td>any data type</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td>Value of object. For true/false objects (radio, checkbox, toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid. number</td>
<td></td>
</tr>
<tr>
<td><strong>Visible</strong></td>
<td>Object visibility. 'on' or 'off'</td>
<td></td>
</tr>
<tr>
<td><strong>WhiteBackgroundColorPrint</strong></td>
<td>Sets the color of the figure and all underlying objects. Useful when printing a figure, so that the background will be white. [r g b] or one character color</td>
<td></td>
</tr>
<tr>
<td><strong>WindowButtonDownFcn</strong></td>
<td>Function to be called when the mouse button is pressed down on the figure. callback string</td>
<td></td>
</tr>
<tr>
<td><strong>WindowButtonMotionFcn</strong></td>
<td>Sets the function to be called when the mouse is being moved over the figure. This function may be called many times when the mouse is moved, so lengthy operations should not be performed. callback string</td>
<td></td>
</tr>
<tr>
<td><strong>WindowButtonUpFcn</strong></td>
<td>Function to be called when the mouse button is released. callback string</td>
<td></td>
</tr>
<tr>
<td><strong>WindowStyle</strong></td>
<td>'modal' sets figure to be always on top. 'modal','normal'</td>
<td></td>
</tr>
</tbody>
</table>

**See Also**
axes, clf, close, colordef, dialog, gcf, plot

- **grid** .......................... Toggle grid lines on axes

**Description**
grid('on') turns on grid lines.
grid('off') turns off grid lines.

**See Also**
title, xlabel, ylabel
- **hold** ............................... Set hold state

  **Description**

  `hold('on')` turns on hold state.
  `hold('off')` turns off hold state.

  When hold is on, plotting new graph does not erase the older one. Graphics are accumulated on the current window.

  **See Also**

  close,ishold,newplot,plot

- **legend** ........................... Add legend to plot

  **Description**

  `legend(legend1,legend2,...)` adds the given legends to the current plot. An optional position parameter specifies where the legend will be placed.

<table>
<thead>
<tr>
<th>Position</th>
<th>Where</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>Outside</td>
</tr>
<tr>
<td>0</td>
<td>Upper right</td>
</tr>
<tr>
<td>1</td>
<td>Upper right</td>
</tr>
<tr>
<td>2</td>
<td>Upper left</td>
</tr>
<tr>
<td>3</td>
<td>Lower left</td>
</tr>
<tr>
<td>4</td>
<td>Lower right</td>
</tr>
</tbody>
</table>

  **See Also**

  plot,title,xlabel,ylabel,zlabel
• **print** ........................................... Print figure

**Description**

`print` shows a print dialog, preparing to print the current figure. `print(filename)` prints to the file with the graphics format according to the file extension.

**See Also**

`printdlg`

• **subplot** ...................................... Create several plots

**Description**

`subplot(m,n,p)` creates `m x n` subplots in the current viewable area and switches the focus to the `p`-th plot.

**See Also**

`newplot,plot,subimage`

• **text** ....................................... Place text on plot

**Description**

`text(x,y,s)` places the string `s` at the coordinates `(x,y)` on the current plot.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Property</td>
<td>Description</td>
<td>Value Type</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is</td>
<td>Callback string</td>
</tr>
<tr>
<td></td>
<td>clicked or a character arrives to a COM port.</td>
<td></td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>Color</td>
<td>Object color.</td>
<td>[r g b] or one character</td>
</tr>
<tr>
<td></td>
<td></td>
<td>color</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>Enable</td>
<td>Is object enabled.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>EraseMode</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Extent</td>
<td>Identical to Position.</td>
<td></td>
</tr>
<tr>
<td>FontAngle</td>
<td>Angle of the font.</td>
<td></td>
</tr>
<tr>
<td>FontName</td>
<td>Sets the name of the font.</td>
<td>Normal, italic, or oblique</td>
</tr>
<tr>
<td>FontSize</td>
<td>Sets the size of font. The size unit is set by the FontUnits property.</td>
<td>positive integer</td>
</tr>
<tr>
<td>FontUnits</td>
<td>Units for the FontSize property.</td>
<td>'inches', 'centimeters', 'normalized', 'points', 'pixels'</td>
</tr>
<tr>
<td>FontWeight</td>
<td>Weight of font.</td>
<td>'normal', 'light', 'demi', 'bold'</td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Foreground color of the object. For some objects, the foreground color may be</td>
<td>[r g b] or one character</td>
</tr>
<tr>
<td></td>
<td>the font color.</td>
<td>color</td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HitTest</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Interruptible</td>
<td>Note implemented.</td>
<td></td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>MouseDownFcn</td>
<td>Action to occur after the mouse is clicked on the object.</td>
<td>callback string</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Position</td>
<td>Position of the object in terms of the units defined in the Units property.</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next shg if 1.</td>
<td></td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>SetFont</td>
<td>Shows a font selection dialog box.</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
<td></td>
</tr>
<tr>
<td>Style</td>
<td>Style of a object. See the uicontrol and hwcontrol functions for available styles.</td>
<td></td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
<td></td>
</tr>
<tr>
<td>TooltipString</td>
<td>Hint that appears when the mouse rests over the object.</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td></td>
</tr>
<tr>
<td>Units</td>
<td>Units for the Position property. 'Normalized' is relative to the parent size in the range zero to one.</td>
<td></td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td>Value of object. For true/false objects (radio,checkbox,toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid.</td>
<td></td>
</tr>
<tr>
<td>Visible</td>
<td>Object visibility.</td>
<td></td>
</tr>
</tbody>
</table>

**See Also**

gtext, plot, title, xlabel, ylabel, zlabel
• **title** ........................................ Add plot title

**Description**
title(s) sets the plot title to s.

**See Also**
grid, legend, plot, text, xlabel, ylabel, zlabel

• **xlabel** ............................. Add label to the x axis

**Description**
xlabel(s) sets the x axis label to s.

**See Also**
grid, legend, plot, text, title, ylabel, zlabel

• **ylabel** ............................. Add label to the y axis

**Description**
ylabel(s) sets the y axis label to s.

**See Also**
grid, legend, plot, text, title, xlabel, zlabel
• **zlabel** .................................Add label to the z axis

**Description**

`zlabel(s)` sets the z axis label to `s`.

**See Also**

`legend, plot3, text, title, xlabel, ylabel`

• **zoom** .................................Zoom into plot

**Description**

`zoom(a)` sets the zooming on the current axes. `a` may be `'on'`, `'off'` or a numeric zoom factor.

**See Also**

`plot, rotate3d`
28 Color functions

- **autumn** .............................. Autumn colormap

  **Description**

  \(c = \text{autumn}(N)\) returns an \(N\)-by-3 colormap of autumn colors. If \(N\) is omitted, it is set to the number of colors in current colormap(figure).

  **See Also**

colormap, summer, winter

- **bone** .............................. Bone colormap

  **Description**

  \(c = \text{bone}(N)\) returns an \(N\)-by-3 colormap of shades of bone colors. If \(N\) is omitted, it is set to the number of colors in current colormap(figure).

  **See Also**

colormap

- **brighten** .......................... Brightens the current colormap

  **Description**

  `brighten(b)` brightens the current colormap by the factor \(b\) if \(b\) positive, or darkens it if \(b\) negative. \(b\) can be -1 to 1. `brighten(map,b)` operates on the supplied colormap, and can also return the modified colormap.
See Also
contrast

- **caxis** ............ Set color axis values for plots and images

Description
caxis(c) sets the color axis minimum and maximum values for 3-dimensional plots and images.
c should be of the format \([c_{\text{min}} \; c_{\text{max}}]\), where \(c_{\text{min}}\) is the color to be mapped to the start of the colormap and \(c_{\text{max}}\) is the color to be mapped to the end of the colormap. \([c_{\text{min}},c_{\text{max}}]=\text{caxis}\) returns the minimum and maximum values. \(c=\text{caxis}\) returns the minimum and maximum values as a vector of two elements.

See Also
image,surf

- **colorbar** .................. Display colormap values

Description
colorbar or colorbar('vert') adds a vertical colorbar of the current colormap next to the current axes. colorbar('horiz') adds an horizontal colorbar.
See Also

- **colordf** .......Set figure default colors to black or white

  **Description**

  `colordf white` or `colordf black` sets the default color scheme of figures or plots to the color specified. `colordf(f,color)` sets the color scheme of figure `f` to the color specified. `colordf('new',color)` creates a new figure with the specified color scheme.

  **See Also**

  `figure,whitebg`

- **colormap** ................. Set and get the colormap

  **Description**

  `c=colormap` returns the current colormap. `colormap(c)` sets the current colormap to `c`. The current colormap should have 3 columns, which are the intensity of red, green and blue for each entry.

  **See Also**

  `black,cool,hot,hsv,jet,pink,prism,white`

- **contrast** ................. Enhance contrast of image

  **Description**

  `map=contrast(img,1)` returns 1 x 3 colormap that improves the contrast of image `img`. 
See Also

brighten

cool ............................................. Cool colormap

Description

c=cool(N) returns an N-by-3 colormap of cool set of colors. if N is omitted, it is set to the number of colors in current colormap(figure).

See Also

colormap

copper ................................. Copper colormap

Description

c=copper(N) returns an N-by-3 colormap of copper colors. if N is omitted, it is set to the number of colors in current colormap(figure).

See Also

colormap

flag ......................................... Flag colormap

Description

c=flag(N) returns an N-by-3 colormap of red, black, blue and white colors. if N is omitted, it is set to the number of colors in current colormap(figure).
See Also

colormap

- **gray**  
  Gray colormap

  **Description**

  \( c=\text{gray}(N) \) returns an \( N \)-by-3 colormap of shades of gray. if \( N \) is omitted, it is set to the number of colors in current colormap(figure).

  **See Also**

colormap, grey

- **grey**  
  Grey colormap

  **Description**

  \( c=\text{grey}(N) \) returns an \( N \)-by-3 colormap of shades of gray. if \( N \) is omitted, it is set to the number of colors in current colormap(figure).

  **See Also**

colormap, gray

- **hot**  
  Hot colormap

  **Description**

  \( c=\text{hot}(N) \) returns an \( N \)-by-3 colormap of hot colors (shades of red). if \( N \) is omitted, it is set to the number of colors in current colormap(figure).
See Also
colormap

• hsv ............................... HSV colormap

Description
c=hsv(N) returns an N-by-3 colormap of Hue-Saturation-Value colors. If N is omitted, it is set to the number of colors in current colormap(figure).

See Also
colormap,jet

• jet ................................. Jet colormap

Description
c=jet(N) returns an N-by-3 colormap of hsv variant. If N is omitted, it is set to the number of colors in current colormap(figure).

See Also
colormap,hsv

• lines  .................. Colormap of default plot line colors

Description
c=lines(N) returns an N-by-3 colormap of default plot line style colors. If N is omitted, it is set to the number of colors in current colormap(figure).
See Also
colormap

• pink .................................................. Pink colormap

Description
c=pink(N) returns an N-by-3 colormap of pink colors. If N is omitted, it is set to the number of colors in current colormap(figure).

See Also
colormap

• prism .................................................. Prism colormap

Description
c=prism(N) returns an N-by-3 colormap of circulating colors. If N is omitted, it is set to the number of colors in current colormap(figure).

See Also
colormap

• rgbplot .............................................. Colormap plot

Description
rgbplot(m) plots the colormap m as three red, green and blue lines with values between 0 and 1.
See Also

- **shading** .......................... Set shading for surface plot

  **Description**

  shading facelet adds a grid to the current surface plot. shading none removes the grid from the current surface plot. shading interp smoothes the current surface plot. The smoothing is performed by interpolating the current values, which will considerably slow redraw speed.

  **See Also**

  mesh, surf

- **spring** .......................... Spring colormap

  **Description**

  c=spring(N) returns an N-by-3 colormap of spring colors. if N is omitted, it is set to the number of colors in current colormap(figure).

  **See Also**

  colormap

- **summer** .......................... Summer colormap

  **Description**

  c=summer(N) returns an N-by-3 colormap of summer colors. if N is omitted, it is set to the number of colors in current colormap(figure).
See Also
autumn, colormap

• white ............................ White colormap

Description
c=white(N) returns an N-by-3 colormap of all-white colors. If N is omitted, it is set to the number of colors in current colormap(figure).

See Also
black, colormap

• whitebg .......................... Set figure background

Description
whitebg(fig,c) sets the color of the figure fig to c. If c is omitted, the color is white.

See Also
colordef

• winter ............................. Winter colormap

Description
c=winter(N) returns an N-by-3 colormap of winter colors. If N is omitted, it is set to the number of colors in current colormap(figure).
See Also

autumn, colormap
29 Imaging

• **bmpread**  ................. Read bitmap image from file

  **Description**
  
  \[ \text{[img, map]} = \text{bmpread(filename)} \]
  
  reads the bitmap stored in `filename` and returns it as image values `img` and the colormap `map` of the image.

  **See Also**
  
  `bmpwrite`

• **bmpwrite**  .................. Write image to BMP file

  **Description**
  
  `bmpwrite(img, map, filename)` writes the image represented by image data `img` and colormap `map` into file `filename`.

  **See Also**
  
  `bmpread`

• **getimage**  ................. Get image from graphic object

  **Description**
  
  `img = getimage(obj)` returns the image representing the graphics object `obj`.

  **See Also**
  
  `image`
• **gifread** .......................... Read image from GIF file

**Description**

\[
\text{[img,map]=gifread(filename)}
\]

reads the image stored in `filename` and returns it as image values `img` and the colormap `map` of the image.

**See Also**

gifwrite

• **gifwrite** .......................... Write image to GIF file

**Description**

\[
gifwrite(img,map,filename)
\]

writes the image represented by image data `img` and colormap `map` into file `filename`.

**See Also**

gifread

• **im2double** .......................... Convert image to double

**Description**

\[
im2double(im)
\]

converts the matrix `im` to a image of type `double`.
See Also

- **image** .......................... Display bitmap image

  **Description**
  
h=image(x,y,img,...) displays the bitmap img with coordinates (x,y). If the coordinates are omitted, unit spacing is assumed. Each value of img is an index into the current color map. If img is omitted, an empty image is created.
  
  Extra parameters following img are properties of the image object.
  
h contains the handles for the created image object(s).

  **See Also**
  
bmpwrite,caxis,colormap,getimage,imagesc,imshow,imwrite,pcolor,sound,truesize

- **imagesc** ............................ Create scaled image object

  **Description**
  
h=imagesc(img) or h=imagesc(x,y,img) shows an image, scaled so that the full colormap is used. imagesc(img,c) or imagesc(x,y,img,c) shows a scaled image, with color scales limited between c(1) and c(2).
  
h contains the handles for the created image object(s).

  **See Also**
  
  image,imshow
• **imread** .............. Read image and colormap from file

**Description**

\[ \text{[img, map]} = \text{imread}(\text{filename}, \text{type}) \]

reads the image from file \text{filename} and returns the image values in \text{img} and the colormap in \text{map}. \text{type} is optional, containing the type (extension) of image.

**See Also**

bmpread, bmpwrite, imwrite

• **imshow** ............... Show true-color or indexed image

**Description**

\text{imshow(img,m)} shows the intensity image \text{img}. If \text{img} is an RGB image, it is displayed with colormap of \text{m}. If \text{img} is an indexed (256 colors) image, the colormap depends upon \text{m} with the following options:

\text{m is N-by-3}

Colormap is \text{m}.

\text{m is scalar}

gray colormap with \text{m} entries.

\text{m is 2-element vector}

gray colormap with \([\text{m}(1) \ \text{m}(2)]\) color scaling.

**See Also**

image, imagesc, subimage
• **imwrite** ...............Write image and colormap to file

**Description**

imwrite(img,map,filename,type) writes the image data img and colormap map into the file filename. type is optional, specifying file type.

imwrite(img,filename,type) writes the image data img into file filename. For indexed-color images, the current colormap is used.

**See Also**

bmpwrite,image,imread

• **pcxread** ..................Read image from PCX file

**Description**

[img,map]=pcxread(filename) reads the image from file filename and returns it as image values img and colormap map of the image.

**See Also**

pcxwrite

• **pcxwrite** ..................Write image to PCX file

**Description**

pcxwrite(img,map,filename) writes the image data img and colormap map into file filename.

**See Also**

pcxread
• **subimage** ........ Prepare image in multi-images figure

**Description**

`subimage` performs the same functionality of `imshow` for multiple-image figures.

**See Also**

`imshow`, `subplot`

---

• **tiffread** .................. Read image from TIFF file

**Description**

```
[img, map] = tiffread(filename)
```

reads the image from file `filename` and returns image values `img` and colormap `map` of the image.

**See Also**

tiffwrite

---

• **tiffwrite** .................. Write image to TIFF file

**Description**

```
tiffwrite(img, map, filename)
```

writes the image data `img` and colormap `map` into file `filename`.

**See Also**

tiffread
- **truesize** .......................... Rescale figure to true size

**Description**

`truesize(fig)` scales figure `fig` to fit to the true size of the image it contains. If `fig` is omitted, the current figure is used.

**See Also**

`image`
30 User-interface dialogs

- dialog .................................................... Generic dialog
  
  **Description**

dialog(...) opens a dialog figure which is a figure with no menu. The arguments to dialog are interpreted as properties to the created figure object.

  **See Also**

figure,inputdlg,msgbox,waitbar

- errordlg .................................................. Error dialog

  **Description**

errordlg(m,t) shows an error dialog with title t and message m.

  **See Also**

msgbox

- helpdlg ................................................ Help dialog

  **Description**

helpdlg(m,t) shows a help dialog with title t and message m.

  **See Also**

msgbox
• **helpwin** .......................... Show help window

Description

`helpwin(text,title)` shows a dialog with the title and text. `helpwin(filename)` shows a dialog with the help inside the file `filename`.

See Also

• **inputdlg** .......................... Input dialog

Description

`a=inputdlg(prompt,title,lineno,defval)` shows an input dialog, requesting the user to enter values into the various fields. Input arguments can be matrices for single field input or cell arrays of matrices for multiple-field input dialog.

`prompt` is the description of each input field; `title` is the title of the dialog box; `lineno` is the number of lines to present for each input field; `defval` is an optional default values to display for the input fields. Th function returns the values typed or an empty string matrix if the Cancel button was pressed.

See Also

dialog

• **makemenu** .......................... Create user menu

Description

This command is not implemented. The way to create menus is to use the `uimenu` command. Here is a simple example of creating menus using `uimenu`: 
Examples

```matlab
m1=uimenu('label','menu1');
m11=uimenu(m1,'label','menu11','callback','disp(''menu11'')');
m12=uimenu(m1,'label','menu12','callback','disp(''menu12'')');
m2=uimenu('label','menu2','callback','disp(''menu2'')');
```

See Also

uimenu

- **menu** ........................................ Menu dialog

Description

`menu(title,but1,but2,...,butn)` presents a menu for the user to choose from. `title` is the title of the menu, and `but1...butn` are the texts on the buttons. The function returns the number of the button pressed.

See Also

disp,input,uimenu

- **msgbox** ........................................ Message box

Description

`msgbox(m,t,type)` shows a dialog with title `t` and message `m`. The icon on the dialog is set by the `type` string, which may be one of 'help', 'error', 'warn', 'none' or 'quest'.

See Also

dialog,errordlg,helpdlg,questdlg,warndlg
• **printdlg** ............................... Print dialog

**Description**

`printdlg` shows a print dialog before printing.

**See Also**

`print`

• **questdlg** ................................. Question dialog

**Description**

`c=questdlg(m,t)` shows a question dialog with title `t`, message `m` and three buttons: `Yes`, `No` and `Cancel`.

The return value `c` is `'Yes'`, `'No'` or `'Cancel'` according to the button pressed.

**See Also**

`msgbox`

• **uigetfile** ................................. Open file dialog

**Description**

`[name,path]=uigetfile(initname,title)` displays a standard open file dialog. `initname` is the initial filename to display. `title` is the title of the load dialog box. `uigetfile` returns the `name`, the name of the selected file and `path`, the path of the selected file.

**See Also**

`uiputfile`
• **uiputfile** .......................... Save file dialog

**Description**

\[ \text{[name,path]=uiputfile(initname,title)} \]

displays a standard save file dialog. \text{initname} is the initial filename to display. \text{title} is the title of the save dialog box. \text{uiputfile} returns the \text{name}, the name of the selected file and \text{path}, the path of the selected file.

**See Also**

uigetfile

• **uisetcolor** .......................... Color selection dialog

**Description**

\[ \text{c=uisetcolor(h,t)} \]

displays a color selection dialog box with the title \text{t}. If \text{h} is an RGB vector, it is the default color; if \text{h} is a valid handle, \text{uisetcolor} sets the color of the object \text{h}. \text{c} is the RGB vector of the color selected.

**See Also**

uisetfont

• **uisetfont** .......................... Font selection dialog

**Description**

\[ \text{uisetfont(h)} \]

shows a font selection dialog box and sets the font of object \text{h} to the selected font.

**See Also**

uisetcolor
- **waitbar** ................................. Progress bar dialog

  **Description**

  h=waitbar(x,title) displays a progress dialog with title caption bar and initial value of x, which must be between 0 and 1. Further calls to waitbar(x) modify the progress bar position. h is the handle to the created figure. close(h) will remove the waitbar.

  **See Also**

dialog

- **warndlg** ................................. Warning dialog

  **Description**

  warndlg(m,t) shows a warning dialog with the title t and message m.

  **See Also**

  msgbox
31 Handle graphics functions

- **allchild** ............................................. Child list
  
  **Description**
  
  allchild(h) returns a vector containing all the childs of the graphics object h.

  **See Also**

- **axes** ...................... Create a axes for subsequent plots
  
  **Description**
  
  axes(h) makes axes h the current axes on the current figure. h=axes('prop1','value1',...) creates an axes object and sets the properties to this object. h is the handle to the created axes object.

  **Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Box</td>
<td>Determines if there is a box around the axes.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>Function to be called when the mouse button is pressed down while the pointer is over the object.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Cdata</td>
<td>Color data of the image.</td>
<td>2D palleted matrix or 3D RGB matrix. handles vector</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td></td>
</tr>
<tr>
<td>Clear</td>
<td>Clear the axes.</td>
<td></td>
</tr>
<tr>
<td>Clim</td>
<td>Stretch color vector for images or surfaces on the axes.</td>
<td>[mincolor maxcolor]</td>
</tr>
</tbody>
</table>
| **Color**             | Object color.                                      | \[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color
| **CreateFcn**        | Function to call when the object is created.       | callback string
| **DeleteFcn**        | Function to call when the object is deleted.       | callback string
| **Editchart**        | Displays the plot editor.                          | ‘on’ or ‘off’
| **Enable**           | Is object enabled.                                 | ‘on’ or ‘off’
| **EraseMode**        | Not implemented.                                   | ‘on’ or ‘off’
| **Extent**           | Identical to **Position**.                         | ‘on’ or ‘off’
| **ForegroundColor**  | Foreground color of the object. For some objects,  | \[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color
| **GridLineStyle**    | The grid line style of the axes.                   | ‘on’ or ‘off’
| **HandleVisibility**| Not implemented.                                   | ‘on’ or ‘off’
| **HitTest**          | Not implemented.                                   | ‘on’ or ‘off’
| **Interruptible**    | Not implemented.                                   | ‘on’ or ‘off’
| **Kill**             | Kills the object.                                  | ‘on’ or ‘off’
| **Legend**           | Show or hide the legend on axes.                   | ‘on’ or ‘off’
| **LegendPos**        | Legend position.                                   | ‘on’ or ‘off’
| **LegendString**     | Legend strings.                                    | ‘on’ or ‘off’
| **MouseDownFcn**     | Action to occur after the mouse is clicked on the  | ‘on’ or ‘off’
| **NextPlot**         | Next plot will ‘add’ to the current axes or ‘replace’ all existing plots on the axes. | ‘on’ or ‘off’
| **Parent**           | The object parent. Determines where the object will appear. | ‘on’ or ‘off’
| **PlotBoxAspectRatio** | Aspect ratio of the axes box.                     | ‘on’ or ‘off’
| **Pointer**          | Select mouse pointer image.                        | ‘on’ or ‘off’
| **Position**         | Position of the object in terms of the units defined in the **Units** property. | ‘on’ or ‘off’
| **Projection**       | Projection mode of the depth axis of the axes.     | ‘on’ or ‘off’

\[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color

\[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color

\[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color

\[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color

\[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color

\[
\begin{array}{c}
[r \ g \ b] \\
on \\
\end{array}
\] or one character color
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>RealPosition</td>
<td>Exact setting of the axes position, in pixel units. Intended for use by MFC applications that use <code>winaxes</code>.</td>
<td>[left top width height]</td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next <code>shg</code> if 1.</td>
<td>0,1</td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
<td>string</td>
</tr>
<tr>
<td>Style</td>
<td>Style of an object. See the <code>uicontrol</code> and <code>hwcontrol</code> functions for available styles.</td>
<td>string</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for <code>findobj</code> commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>TickDir</td>
<td>Direction of the ticks: inside or outside the axes.</td>
<td>'in', 'out'</td>
</tr>
<tr>
<td>Title</td>
<td>Handle of title object.</td>
<td>handle</td>
</tr>
<tr>
<td>TooltipString</td>
<td>Hint that appears when the mouse rests over the object.</td>
<td>string</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td>string</td>
</tr>
<tr>
<td>Units</td>
<td>Units for the <code>Position</code> property. 'Normalized' is relative to the parent size in the range zero to one.</td>
<td>'normalized',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'inches',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'centimeters',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'points',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'characters',</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'pixels'</td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td>any data type</td>
</tr>
<tr>
<td>Value</td>
<td>Value of object. For true/false objects (radio, checkbox, toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid.</td>
<td>number</td>
</tr>
<tr>
<td>View</td>
<td>View mode of the axes to 2D, 3D or 3D specific angles.</td>
<td>2,3 or [azim elev]</td>
</tr>
<tr>
<td>Visible</td>
<td>Object visibility.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Xcolor, Ycolor,</td>
<td>Axis color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>Zcolor</td>
<td></td>
<td>date format</td>
</tr>
<tr>
<td>Xdate, Ydate, Zdate</td>
<td>Date format for axis ticks.</td>
<td>string</td>
</tr>
<tr>
<td>Xdir, Ydir, Zdir</td>
<td>Invert axis.</td>
<td>'normal', 'inverse'</td>
</tr>
<tr>
<td>Xgrid, Ygrid, Zgrid</td>
<td>Enable a grid along the axis.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Xlabel, Ylabel,</td>
<td>Handle of label.</td>
<td>handle</td>
</tr>
<tr>
<td>Zlabel</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MatrixVB Function Reference Guide

Xlim, Ylim, Zlim
Axis display limits.
Automatically determine display limits or use limits from the Xlim, Ylim, Zlim properties.
[min max]

XlimMode, YlimMode, ZlimMode
Scale for axis values.

Xscale, Yscale, Zscale
Axes position.

Xtick, Ytick, Ztick
Ticks location on the axis.

XTickInc, YTickInc, ZTickInc
Increment of axes ticks.

XTickIncMode, YTickIncMode, ZTickIncMode
Use automatic tick positions or values from XTickInc, YTickInc, ZTickInc properties.

XTickLabel, YTickLabel, ZTickLabel
Axis labels.

XTickLabelMode, YTickLabelMode, ZTickLabelMode
Show labels automatically or use values from XTickLabel, YTickLabel, ZTickLabel properties.

XTickMode, YTickMode, ZTickMode
Determine ticks on axis automatically or from Xtick, Ytick, Ztick properties.

ZoomFactor
Control figure zoom. 'on' or 'off': enable mouse zooming. 'out': zoom out to 1:1. Any other number number: zoom factor. 1-200, 'on', 'off', 'out'

See Also
box, cla, figure, gca, newplot

axesposition

Axes position

axesposition(left, top, width, height) sets the plot position in a C++ window or VB form.

See Also
winaxes
• **box** .......................... Make box around plot

**Description**

box('on') shows a box around the current axes. box('off') removes the box.

**See Also**

axes

• **capture** .......................... Capture figure area

**Description**

[img,map]=capture(fig) captures the bitmap of the figure fig into bitmap img and colormap map.

**See Also**

getframe

• **cla** .......................... Clear axes

**Description**

cla removes all objects on current axes. cla reset also reset current axes to its default state.

**See Also**

axes,clf
• **clabel** .......................... Add contour heights labels

  **Description**
  
  `clabel(cmatrix)` adds height text labels to contours represented by `cmatrix`, which may be the output of `contour`.

  **See Also**
  `contour`

• **clg** ............................ Clear figure

  **Description**
  
  `clg` is identical to `clf`.

  **See Also**
  `clc`, `clf`

• **clruprop** ....................... Clear user property

  **Description**
  
  `clruprop(h,name)` clears the value of the `name` property of the object `h`.

  **See Also**
  `getuprop`, `setuprop`
• **datetick** .......................... Label ticks with dates

**Description**

datetick(axis,date_format) sets the labels for the axis axis to the date format specified in date_format. axis can be one of 'x', 'y' or 'z'. date_format can contain combinations of 'y','yy','yyyy','m','mm','mmm','dd',':','hh','mm','ss'.

**See Also**
date,datenum

• **delete** .......................... Delete graphics object

**Description**
delete(h) delete the object h.
delete is named deleteobj in Visual Basic programs.

**See Also**
cd,copyfile,deleteM,deleteobj,dir,exist,findobj

• **deleteobj** .......................... Delete graphics object

**Description**
delete is named deleteobj in Visual Basic programs.

**See Also**
delete,findobj
• **drawnow** .......................... Force graphics update

**Description**

*drawnow* draws all the queued graphics operations for all figures.

**See Also**

refresh, shg

• **figure_close** .......................... Close window

**Description**

*close* is named *figure_close* in Visual Basic programs.

**See Also**

close

• **fill** .......................... Filled polygons with color

**Description**

*fill(x,y,c)* creates a filled polygon in the x, y points and color c.
See Also

- **findall** .................................Find all graphics objects

  Description
  returns all childs of object h.

  See Also
  allchild, findobj

- **findfigs** .................................Find hidden figures

  Description
  findfigs finds all figures positioned off-screen and returns them to the screen area.

  See Also

- **findobj** .................................Find graphics object

  Description
  findobj(‘prop1’,’name1’,...) finds all objects having property ‘prop1’ with value ‘name1’. If no property pairs are given, findobj shows all objects recursively.
  findobj(h,...) begins searching from object(s) h and down the children tree.
  An optional ‘flat’ argument cause findobj to search only the given objects, skipping the recursive search.
See Also
delete, deleteobj, get, ishandle, set

• gca ................................. Get current axes

Description
gca returns the handle of the current axes or image in the figure, or creates new axes if no axes exists.

See Also
axes, gcf

• gcbf ................................. Get callback originating figure

Description
h=gcbf returns the figure handle in which the control that originated the callback exists.

See Also
gcbo, gcf, gco

• gcbo ................................. Get callback originating object

Description
[obj, fig]=gcbo returns obj, the handle to the control that originated the callback, and fig, the handle of the figure originating the callback.
See Also

gcf,gco

- **gcf** .......................... Get current figure

**Description**
gcf returns the handle of the current figure.

See Also

figure(gca,gcf,gcbo,gco)

- **gco** .......................... Get current object

**Description**
gco returns the object whose callback is currently processing.

See Also

gcbf,gcbo,gcf

- **get** .......................... Get object property

**Description**

get(h,name) gets the value of property name of the object h. name may be cell array, in which case a cell array with the respective properties values is returned.

get is named mget in Visual Basic programs.
See Also
findobj,mget,set

- **getframe** .......................... Capture figure area

  Description

  \[ \text{[img,map]} = \text{getframe}(h) \]

  captures the bitmap of the graphic object \( h \) into bitmap \( \text{img} \) and colormap \( \text{map} \). \( h \) can be handle to a figure or an axes.

  See Also

capture

- **getpts** .......................... Visual selection of points

  Description

  \[ \text{[x y]} = \text{getpts}(h) \]

  allows the user to interactively select points from the handle \( h \), which can be figure or axes. The coordinates of the points are returned in the \( x \) and \( y \) vectors.

  See Also

ginput

- **getuprop** .......................... Get user property

  Description

  \( \text{getuprop}(h,\text{name}) \) gets the value of the \text{name} property of the object \( h \).
See Also
cruprop,setuprop

• **ginput** ................. Select axes point using mouse

**Description**

\([x,y,b]=\text{ginput}(n)\) requests the user to click \(n\) times on current axes. **ginput** returns the \(x\) and \(y\) data coordinates of the clicked points, and the button pressed \(b\). The values in \(b\) specify which button was pressed.

<table>
<thead>
<tr>
<th>Value of (b)</th>
<th>Button</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Left</td>
</tr>
<tr>
<td>2</td>
<td>Middle</td>
</tr>
<tr>
<td>3</td>
<td>Right</td>
</tr>
</tbody>
</table>

See Also
gtext,input

• **gtext** .................. Place text on axes using mouse

**Description**

\(h=\text{gtext}(s)\) requests the user to click on a point on current axes, and places the string \(s\) there. \(h\) is the handle to the created text object.

See Also
ginput, text
•  **hwcontrol**  .... Create a hardware/communication object

**Description**

h=hwcontrol(p,'prop1','value1',...) creates a hardware or communication control as a child of object p and returns its handle. If p is omitted, the control is created for the current figure. The type of control is specified by the 'Style' property.

Special properties for COM port: Open, Write, Read, Callback. Special properties for audio device: SampleRate, BitsPerSample, Channels, Play, Record, Callback, Data, Stop

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BaudRate</td>
<td>Baud rate for the COM port.</td>
<td>110, 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 56000, 57600, 115200</td>
</tr>
<tr>
<td>BitsPerSample</td>
<td>How many bits are stored for each audio sample.</td>
<td>8, 16</td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Channels</td>
<td>Number of channels for the audio device.</td>
<td>'mono', 'stereo'</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>Close</td>
<td>Closes the figure or COM port. COM port must be closed after using it.</td>
<td></td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>Data</td>
<td>Stores the recorded audio data.</td>
<td></td>
</tr>
<tr>
<td>DataBits</td>
<td>Number of data bits for the COM port.</td>
<td>4, 5, 6, 7, 8</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>FlowControl</td>
<td>Flow control setting of the COM port.</td>
<td>none, rtscts or xonxoff</td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>Open</td>
<td>Open the COM port, must be set prior to using the COM port.</td>
<td></td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
</tbody>
</table>
### MatrixVB Function Reference Guide

**Parity**
Parity checking for the COM port.  
'none', 'even', 'odd', 'space', 'mark'

**Play**
Play the audio device. The vector may be one-column vector for mono sound or two-columns vector for stereo sound.

**Port**
Select communication port, n=1,2,3,4.  
'COMn', 'LPTn'

**Read**
Reading this property of a COM port object returns the received data.

**Record**
Record length samples from the audio device using a buffer of size buffer. After the recording has completed, the function stored in Callback property will be called. Recorder sound is stored in the Data property.

**SampleRate**
Samples per second rate of the audio device.  
11025, 22050, 44100

**Stop**
Stops the play or record action of the audio device.

**StopBits**
Number of COM port stop bits.  
1, 1.5, 2

**Tag**
Stores a tag to identify the object. Useful for findobj commands.

**Type**
Type of object.  
string

**UserData**
Store user specified data in the object. Any data type

**Write**
Writes the data vector to the port.  
Data vector

### See Also

- uicontrol
- ishandle

**ishandle** ................................. Check if a valid handle

**Description**

ishandle(h) returns 1 if h is a valid graphics object handle valid or 0 if not.
See Also
findobj

- **ishold** ........................................... Get hold state

Description

\[ z = \text{ishold} \]

\( z = \text{ishold} \) returns 1 if the current plot is being held and 0 otherwise.

See Also
hold, plot

- **line** ............................................ Create low level line plot

Description

`line(...)` creates a line object. Parameters to `line` are interpreted as properties to the newly created line.

`line` is named `mline` in Visual Basic programs.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>Function to be called when the mouse button is pressed down while the pointer is over the object.</td>
<td></td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td></td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the children of the object.</td>
<td>handles vector [r g b] or one character color</td>
</tr>
<tr>
<td>Color</td>
<td>Object color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>ContourPosition</td>
<td>Contour position in the plot of the contour series.</td>
<td>data value</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>Curvation</td>
<td>Appearance of the rectangle object: 0 or (0,0) is a rectangle; 1 or (1,1) is a circle.</td>
<td>0, 1, (0,0), (1,1)</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>EdgeColor</td>
<td>Edge color of the series.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>Enable</td>
<td>Is object enabled.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>EraseMode</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Extent</td>
<td>Identical to Position.</td>
<td></td>
</tr>
<tr>
<td>FaceColor</td>
<td>Edge color of the series.</td>
<td></td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Foreground color of the object. For some objects, the foreground color may be the font color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HitTest</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HorizontalAlignment</td>
<td>Text alignment.</td>
<td>left, middle or right</td>
</tr>
<tr>
<td>Interruptible</td>
<td>Note implemented.</td>
<td></td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>LineStyle</td>
<td>Series line style.</td>
<td></td>
</tr>
<tr>
<td>LineWidth</td>
<td>Series line width.</td>
<td></td>
</tr>
<tr>
<td>Marker</td>
<td>Marker style of the series.</td>
<td></td>
</tr>
<tr>
<td>MarkerEdgeColor</td>
<td>Marker edge color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>MarkerFaceColor</td>
<td>Marker face color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td>Type</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>MarkerSize</td>
<td>Marker size of the series.</td>
<td>positive number</td>
</tr>
<tr>
<td>MouseDownFcn</td>
<td>Action to occur after the mouse is clicked on the object.</td>
<td>callback string</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
<tr>
<td>Position</td>
<td>Position of the object in terms of the units defined in the Units property.</td>
<td>[left bottom width height]</td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next shg if 1.</td>
<td>0,1</td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Shading</td>
<td>Shading style of the series.</td>
<td>'faceted', 'interp', 'flat'</td>
</tr>
<tr>
<td>Stacktype</td>
<td>Stack type for bar series.</td>
<td>'stacked','normal'</td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
<td></td>
</tr>
<tr>
<td>Style</td>
<td>Style of a object. See the uicontrol and hwcontrol functions for available styles.</td>
<td>string</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>TooltipString</td>
<td>Hint that appears when the mouse rests over the object.</td>
<td>string</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td>string</td>
</tr>
<tr>
<td>Units</td>
<td>Units for the Position property. 'Normalized' is relative to the parent size in the range zero to one.</td>
<td>'normalized', 'inches', 'centimeters', 'points', 'characters', 'pixels'</td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td>any data type</td>
</tr>
<tr>
<td>Value</td>
<td>Value of object. For true/false objects (radio, checkbox, toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid.</td>
<td>number</td>
</tr>
<tr>
<td>Visible</td>
<td>Object visibility.</td>
<td>'on' or 'off'</td>
</tr>
</tbody>
</table>
MatrixVB Function Reference Guide

XAxisLocation, YAxisLocation, ZAxisLocation
---
Location of the axis for the series. 'bottom', 'top', 'left', 'right'

Xdata, Ydata, Zdata
---
Series values. vector

See Also
patch, surface

• **mget** .............................. Get object property

Description
get is named mget in Visual Basic programs.

See Also
get

• **mline** ............................ Create low level line plot

Description
line is named mline in Visual Basic programs.
See Also

- **mset** .................. Set object property
  
  **Description**
  
  set is named mset in Visual Basic programs.

  **See Also**
  
  set

- **newplot** ................. Prepare axes for additional plot
  
  **Description**
  
  newplot prepares axes for plotting. newplot combines the logic of the commands subplot, images, axes and hold to decide where to put the axes and whether to use existing axes.

  **See Also**
  
  axes, hold, plot, subplot

- **orient** ................. Orientation of paper for printing
  
  **Description**
  
  a=orient or a=orient(fig) returns the orientation of the figure. orient(or) sets the orientation of the current figure. or can be 'portrait' or 'landscape'. orient(fig,or) sets the orientation of a specific figure.
See Also
print

• patch ..................... Create low level polygon plot

Description

patch(...) creates a patch object. Parameters to patch are interpreted as properties to the newly created patch.

patch is partially supported.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td>Callback string</td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>Function to be called when the mouse button is pressed down while the pointer is over the object.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>Color</td>
<td>Object color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>ContourPosition</td>
<td>Contour position in the plot of the contour series.</td>
<td>data value</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>Curvature</td>
<td>Appearance of the rectangle object: 0 or (0,0) is a rectangle; 1 or (1,1) is a circle.</td>
<td>0, 1, (0,0), (1,1)</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>EdgeColor</td>
<td>Edge color of the series.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>Enable</td>
<td>Is object enabled.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>EraseMode</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Extent</td>
<td>Identical to Position.</td>
<td></td>
</tr>
<tr>
<td>FaceColor</td>
<td>Edge color of the series.</td>
<td></td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Foreground color of the object. For some objects, the foreground color may be the font color.</td>
<td></td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HitTest</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HorizontalAlignment</td>
<td>Text alignment.</td>
<td></td>
</tr>
<tr>
<td>Interruptible</td>
<td>Note implemented.</td>
<td></td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>LineStyle</td>
<td>Series line style.</td>
<td></td>
</tr>
<tr>
<td>LineWidth</td>
<td>Series line width.</td>
<td></td>
</tr>
<tr>
<td>Marker</td>
<td>Marker style of the series.</td>
<td></td>
</tr>
<tr>
<td>MarkerEdgeColor</td>
<td>Marker edge color.</td>
<td></td>
</tr>
<tr>
<td>MarkerFaceColor</td>
<td>Marker face color.</td>
<td></td>
</tr>
<tr>
<td>MarkerSize</td>
<td>Marker size of the series.</td>
<td></td>
</tr>
<tr>
<td>MouseDownFcn</td>
<td>Action to occur after the mouse is clicked on the object.</td>
<td></td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determined where the object will appear.</td>
<td></td>
</tr>
<tr>
<td>Position</td>
<td>Position of the object in terms of the units defined in the Units property.</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next shg if 1.</td>
<td></td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>Shading</td>
<td>Shading style of the series.</td>
<td>'faceted', 'interp', 'flat'</td>
</tr>
<tr>
<td>Stacktype</td>
<td>Stack type for bar series.</td>
<td>'stacked', 'normal'</td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
<td>string</td>
</tr>
<tr>
<td>Style</td>
<td>Style of an object. See the uicontrol and hwcontrol functions for available styles.</td>
<td>string</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>TooltipString</td>
<td>Hint that appears when the mouse rests over the object.</td>
<td>string</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td>'normalized', 'inches', 'centimeters', 'points', 'characters', 'pixels'</td>
</tr>
<tr>
<td>Units</td>
<td>Units for the Position property. 'Normalized' is relative to the parent size in the range zero to one.</td>
<td>any data type</td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td>any data type</td>
</tr>
<tr>
<td>Value</td>
<td>Value of object. For true/false objects (radio, checkbox, toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid.</td>
<td>number</td>
</tr>
<tr>
<td>Visible</td>
<td>Object visibility.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>XAxisLocation, YAxisLocation, ZAxisLocation</td>
<td>Location of the axis for the series.</td>
<td>'bottom', 'top', 'left', 'right'</td>
</tr>
<tr>
<td>Xdata, Ydata, Zdata</td>
<td>Series values.</td>
<td>vector</td>
</tr>
</tbody>
</table>
- **peaks** ...................... Generate data for 3-D plots

  **Description**
  
  peaks(n) generates n x n data for demonstrating 3-D plots.

  **See Also**

- **popupstr** ...................... Label string of menu item

  **Description**
  
  popupstr(h) returns the text of the label of the selected menu item h.

  **See Also**

- **rectangle** ............... Create low level rectangle object

  **Description**
  
  rectangle(...) creates a rectangle object. Parameters to rectangle are interpreted as properties to the newly created rectangle. 'position', [left bottom width height] set location of the rectangle and 'curvation', [a, a] set its appearance.

  **Examples**
  
  rectangle('position', [0.1 0.1 0.6 0.4], 'curvation', [1 1], 'b') creates a blue circle while rectangle('position', [0.1 0.1 0.8 0.8], 'curvation', [0 0], 'r') draws a red rectangle.

  **Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Property</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>Function to be called when the mouse button is pressed down while the pointer is over the object.</td>
<td></td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td></td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td></td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the child of the object.</td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td>Object color.</td>
<td></td>
</tr>
<tr>
<td>ContourPosition</td>
<td>Contour position in the plot of the contour series.</td>
<td></td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td></td>
</tr>
<tr>
<td>Curvature</td>
<td>Appearance of the rectangle object: 0 or (0,0) is a rectangle; 1 or (1,1) is a circle.</td>
<td></td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td></td>
</tr>
<tr>
<td>EdgeColor</td>
<td>Edge color of the series.</td>
<td></td>
</tr>
<tr>
<td>Enable</td>
<td>Is object enabled.</td>
<td></td>
</tr>
<tr>
<td>EraseMode</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Extent</td>
<td>Identical to Position.</td>
<td></td>
</tr>
<tr>
<td>FaceColor</td>
<td>Edge color of the series.</td>
<td></td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Foreground color of the object. For some objects, the foreground color may be the font color.</td>
<td></td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HitTest</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HorizontalAlignment</td>
<td>Text alignment.</td>
<td></td>
</tr>
<tr>
<td>Interruptible</td>
<td>Note implemented.</td>
<td></td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>LineStyle</td>
<td>Series line style.</td>
<td></td>
</tr>
<tr>
<td>LineWidth</td>
<td>Series line width.</td>
<td></td>
</tr>
</tbody>
</table>

- `[r g b]` or one character color
- `data value` callback string `0, 1, (0,0), (1,1)`
- `callback string` `[r g b]` or one character color
- `'on' or 'off'`
- `left, middle or right`
- `positive integer`
<table>
<thead>
<tr>
<th>Marker</th>
<th>Marker style of the series.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MarkerEdgeColor</td>
<td>Marker edge color.</td>
</tr>
<tr>
<td>MarkerFaceColor</td>
<td>Marker face color.</td>
</tr>
<tr>
<td>MarkerSize</td>
<td>Marker size of the series.</td>
</tr>
<tr>
<td>MouseDownFcn</td>
<td>Action to occur after the mouse is clicked on the object.</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
</tr>
<tr>
<td>Position</td>
<td>Position of the object in terms of the units defined in the Units property.</td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next shg if 1.</td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
</tr>
<tr>
<td>Shading</td>
<td>Shading style of the series.</td>
</tr>
<tr>
<td>Stacktype</td>
<td>Stack type for bar series.</td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
</tr>
<tr>
<td>Style</td>
<td>Style of a object. See the uicontrol and hicontrol functions for available styles.</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
</tr>
<tr>
<td>TooltipString</td>
<td>Hint that appears when the mouse rests over the object.</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
</tr>
<tr>
<td>Units</td>
<td>Units for the Position property. 'Normalized' is relative to the parent size in the range zero to one.</td>
</tr>
</tbody>
</table>

- Marker styles: ., o, x, +, *, s, d, ?, v, >, <, p, h
- Marker edge color: [r g b] or one character color
- Marker face color: [r g b] or one character color
- Marker size: positive number
- MouseDownFcn: callback string
- Parent: handle
- Position: [left bottom width height]
- Refresh: 0,1
- Shading styles: 'faceted', 'interp', 'flat', 'stacked', 'normal'
- Stack type: 'stacked', 'normal'
- Units: 'Normalized' is relative to the parent size in the range zero to one.
<table>
<thead>
<tr>
<th>UserData</th>
<th>Store user specified data in the object.</th>
<th>any data type number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Value of object. For true/false objects (radio, checkbox, toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid.</td>
<td>number</td>
</tr>
<tr>
<td>Visible</td>
<td>Object visibility.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>XAxisLocation, YAxisLocation, ZAxisLocation</td>
<td>Location of the axis for the series.</td>
<td>'bottom', 'top', 'left', 'right'</td>
</tr>
<tr>
<td>Xdata, Ydata, Zdata</td>
<td>Series values.</td>
<td>vector</td>
</tr>
</tbody>
</table>

**See Also**

line, patch

- **refresh** .......................... Refresh graphics on figure

**Description**

refresh(h) performs all queued windows events for the figure h. If h is omitted, the current figure is assumed.

**See Also**

drawnow, shg, vbrefresh

- **reset** ............................... Reset graphics object

**Description**

reset(h) resets the graphics object h.
See Also

- **root** ............................. Graphics root object

Description

The main graphics object. The root object has handle number zero, which can be used to retrieve information about the system, like screen size and color depth. There is no `root` command or function.

Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallbackObject</td>
<td>Read only. Handle of the object whose callback is now executing.</td>
<td>handle</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>CurrentFigure</td>
<td>Current figure handle.</td>
<td>number</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td>handle</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
<tr>
<td>PixelsPerInch</td>
<td>Read only. Number of pixels per screen inch.</td>
<td>number</td>
</tr>
<tr>
<td>ScreenDepth</td>
<td>Read only. Color depth of the screen in bits.</td>
<td>number</td>
</tr>
<tr>
<td>ScreenSize</td>
<td>Read only. Screen size in pixels.</td>
<td>[0 0 width height]</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for <code>findobj</code> commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td>string</td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td>any data type</td>
</tr>
</tbody>
</table>

See Also

get
• **rotate3d** .......................... 3D rotation of plot

**Description**

rotate3d(a) enables 3D rotation and control panel. a can be 'on' or 'off'.

**See Also**

surf, zoom

• **set** .......................... Set object property

**Description**

set(h,name,value) sets the property name of the object h to value. name and value may be cell arrays, in which case each (name, value) pair is set in the object whose handle is h.

set is named mset in Visual Basic programs.

**See Also**

findobj, get, mset

• **setstatus** .......................... Set UI string

**Description**

setstatus(fig,s) finds the object with tag property 'Status' and sets its string to s.
See Also

- **setuprop** .......................... Set user property

  **Description**
  
  `setuprop(h,name,value)` sets the name property of the object h to value.

  **See Also**
  
  `clruprop,getuprop`

- **shg** .......................... Show graph

  **Description**
  
  `shg` performs all queued windows events. If no figure is open, `shg` will create one.

  **See Also**
  
  `drawnow,refresh`

- **surface** .......................... Create low level surface plot

  **Description**
  
  `surface(...)` creates a surface object. Parameters to `surface` are interpreted as properties to the newly created surface.

  `surface` is partially supported.
## Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>Function to be called when the mouse button is pressed down while the pointer is over the object.</td>
<td></td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>Color</td>
<td>Object color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>ContourPosition</td>
<td>Contour position in the plot of the contour series.</td>
<td>data value</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>Curvation</td>
<td>Appearance of the rectangle object: 0 or (0,0) is a rectangle; 1 or (1,1) is a circle.</td>
<td>[0, 1, (0,0), (1,1)]</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>EdgeColor</td>
<td>Edge color of the series.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>Enable</td>
<td>Is object enabled.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>EraseMode</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Extent</td>
<td>Identical to Position.</td>
<td></td>
</tr>
<tr>
<td>FaceColor</td>
<td>Edge color of the series.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Foreground color of the object. For some objects, the foreground color may be the font color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HitTest</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HorizontalAlignment</td>
<td>Text alignment.</td>
<td>left, middle or right</td>
</tr>
<tr>
<td>Interruptible</td>
<td>Note implemented.</td>
<td></td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>LineStyle</td>
<td>Series line style.</td>
<td>-, :,-,--</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>LineWidth</td>
<td>Series line width.</td>
<td>positive integer</td>
</tr>
<tr>
<td>Marker</td>
<td>Marker style of the series.</td>
<td>., o, x, +, * , s, d, †, v, &gt;, &lt;, p, h</td>
</tr>
<tr>
<td>MarkerEdgeColor</td>
<td>Marker edge color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>MarkerFaceColor</td>
<td>Marker face color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>MarkerSize</td>
<td>Marker size of the series.</td>
<td>positive number</td>
</tr>
<tr>
<td>MouseDownFcn</td>
<td>Action to occur after the mouse is clicked on the object.</td>
<td>callback string</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
<tr>
<td>Position</td>
<td>Position of the object in terms of the units defined in the Units property.</td>
<td>[left bottom width height]</td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next shg if 1.</td>
<td>0,1</td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Shading</td>
<td>Shading style of the series.</td>
<td>'faceted', 'interp', 'flat', 'stacked', 'normal'</td>
</tr>
<tr>
<td>Stacktype</td>
<td>Stack type for bar series.</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
<td></td>
</tr>
<tr>
<td>Style</td>
<td>Style of a object. See the uicontrol and hwcontrol functions for available styles.</td>
<td>string</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>TooltipString</td>
<td>Hint that appears when the mouse rests over the object.</td>
<td>string</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td></td>
</tr>
<tr>
<td>Units</td>
<td>Units for the Position property. 'Normalized' is relative to the parent size in the range zero to one.</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td>Value of object. For true/false objects (radio, checkbox, toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid.</td>
<td></td>
</tr>
<tr>
<td>Visible</td>
<td>Object visibility.</td>
<td></td>
</tr>
<tr>
<td>XAxisLocation, YAxisLocation, ZAxisLocation</td>
<td>Location of the axis for the series.</td>
<td></td>
</tr>
<tr>
<td>Xdata, Ydata, Zdata</td>
<td>Series values.</td>
<td></td>
</tr>
</tbody>
</table>

See Also

line, patch, pcolor

- **textwrap**          ......................... Wrap text to fit in object

Description

This command is not implemented.
See Also

- **uicontextmenu** ......... Create a context sensitive menu

**Description**

\[ h = \text{uicontextmenu}('prop1', 'value1', ...) \]

creates a context menu for an object. A context menu is a floating menu which appears after a right click on an object.

A contextmenu is assigned to an object \( \text{obj} \) using \( \text{set}(\text{obj}, 'uicontextmenu', h) \). Using the handle \( h \) and further \text{uimenu} commands, a series of cascaded menus can be created as needed.

**Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checked</td>
<td>True if menu is checked.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>Label</td>
<td>Menu string.</td>
<td>string</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
<tr>
<td>Separator</td>
<td>Is menu item is a separator.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td>string</td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td>any data type</td>
</tr>
</tbody>
</table>

**See Also**

uimenu
• **uicontrol** .................. Create a user interface object

**Description**

`h=uicontrol(p,'prop1','value1',...)` creates a graphics user interface control as a child of object `p` and returns its handle. If `p` is omitted, the control is created for the current figure. The type of control is specified by the `'Style'` property, described below.

<table>
<thead>
<tr>
<th>Object</th>
<th>Style</th>
<th>Special properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Listbox</td>
<td><code>'listbox'</code></td>
<td>Value property is item index, 1-based.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ListBoxTop property is item on top.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String can accept a cell string.</td>
</tr>
<tr>
<td>Checkbox</td>
<td><code>'checkbox'</code></td>
<td>Property Value is checked state of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>check box.</td>
</tr>
<tr>
<td>Text</td>
<td><code>'text'</code></td>
<td>HorizontalAlignment property is available.</td>
</tr>
<tr>
<td>Edit</td>
<td><code>'edit'</code></td>
<td>If the Max property is larger than the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Min property, the edit box will accept</td>
</tr>
<tr>
<td></td>
<td></td>
<td>newlines.</td>
</tr>
<tr>
<td>Radiobutton</td>
<td><code>'radiobutton'</code></td>
<td>Property Value is selected state of the</td>
</tr>
<tr>
<td>ComboBox</td>
<td><code>'popupmenu'</code></td>
<td>radio button.</td>
</tr>
<tr>
<td>Pushbutton</td>
<td><code>'pushbutton'</code></td>
<td>Value property is item index, 1-based.</td>
</tr>
<tr>
<td>Frame</td>
<td><code>'frame'</code></td>
<td>ListBoxTop property is item on top.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>String can accept a cell string.</td>
</tr>
<tr>
<td>Slider</td>
<td><code>'slider'</code></td>
<td>The Min and Max properties set slider</td>
</tr>
<tr>
<td></td>
<td></td>
<td>limits. Value is the slider position.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SliderStep controls slider steps.</td>
</tr>
<tr>
<td>Progress bar</td>
<td><code>'progressbar'</code></td>
<td>The Min and Max properties set the bar</td>
</tr>
<tr>
<td></td>
<td></td>
<td>limits. Value is the bar position.</td>
</tr>
<tr>
<td>Grid,Spreadsheet</td>
<td><code>'grid'</code></td>
<td>Available properties are RowCount, ColCount, RowHeight, ColWidth, FixedRows, FixedCols. GridValue property set the entire grid from a matrix. The Row, Col and Value properties set value to a specific cell. Enable property enables user editing of cells.</td>
</tr>
<tr>
<td>Scope</td>
<td><code>'scope'</code></td>
<td>Value is current scope value.</td>
</tr>
<tr>
<td>Indicator</td>
<td><code>'indicator'</code></td>
<td>Value is indicator height.</td>
</tr>
<tr>
<td>Led</td>
<td><code>'led'</code></td>
<td>Value is 'on' or 'off' controls LED.</td>
</tr>
<tr>
<td>Lcd</td>
<td><code>'lcd'</code></td>
<td>Length is available number of digits.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value is value shown in LCD.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Meter</td>
<td>'meter'</td>
<td>The Min and Max properties set the meter limits. Value is meter value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toggle Button</td>
<td>'togglebutton'</td>
<td>Value is 0,1 controls toggle state.</td>
</tr>
<tr>
<td>Switch</td>
<td>'switch'</td>
<td>Value is 0,1 controls switch state.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Properties</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Values</td>
</tr>
<tr>
<td>BusyAction</td>
<td>Not implemented.</td>
<td>Callback string</td>
</tr>
<tr>
<td>ButtonDownFcn</td>
<td>Function to be called when the mouse button is pressed down while the pointer is over the object.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Callback</td>
<td>Action to occur after the object is signaled. For example after a button is clicked or a character arrives to a COM port.</td>
<td>Callback string</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the childs of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>ColCount</td>
<td>Number of columns in the grid.</td>
<td>number</td>
</tr>
<tr>
<td>Color</td>
<td>Object color.</td>
<td>[r g b] or one character color</td>
</tr>
<tr>
<td>ColWidth</td>
<td>Width of every column.</td>
<td>positive number</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>Enable</td>
<td>Is object enabled.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>EraseMode</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>Extent</td>
<td>Identical to Position.</td>
<td></td>
</tr>
<tr>
<td>FixedCols</td>
<td>Number of fixed columns.</td>
<td>non-negative integer</td>
</tr>
<tr>
<td>FixedRows</td>
<td>Number of fixed rows.</td>
<td>non-negative integer</td>
</tr>
<tr>
<td>FontAngle</td>
<td>Angle of the font.</td>
<td>Normal, italic or oblique string</td>
</tr>
<tr>
<td>FontName</td>
<td>Sets the name of the font.</td>
<td></td>
</tr>
<tr>
<td>FontSize</td>
<td>Sets the size of font. The size unit is set by the FontUnits property.</td>
<td>positive integer</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>FontUnits</td>
<td>Units for the FontSize property.</td>
<td></td>
</tr>
<tr>
<td>FontWeight</td>
<td>Weight of font.</td>
<td></td>
</tr>
<tr>
<td>ForegroundColor</td>
<td>Foreground color of the object. For some objects, the foreground color may be the font color.</td>
<td></td>
</tr>
<tr>
<td>HandleVisibility</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HitTest</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>HorizontalAlignment</td>
<td>Text alignment.</td>
<td></td>
</tr>
<tr>
<td>Interruptible</td>
<td>Note implemented.</td>
<td></td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>Number of digits to display in the LCD control.</td>
<td></td>
</tr>
<tr>
<td>ListboxTop</td>
<td>Index of the top item in the listbox.</td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>Maximum value of the object.</td>
<td></td>
</tr>
<tr>
<td>Min</td>
<td>Minimum value of the object.</td>
<td></td>
</tr>
<tr>
<td>MouseDownFcn</td>
<td>Action to occur after the mouse is clicked on the object.</td>
<td></td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td></td>
</tr>
<tr>
<td>Position</td>
<td>Position of the object in terms of the units defined in the Units property.</td>
<td></td>
</tr>
<tr>
<td>Refresh</td>
<td>Immediately refresh the object appearance if 0 or to wait until next shg if 1.</td>
<td></td>
</tr>
<tr>
<td>RowCount</td>
<td>Number of rows in the grid.</td>
<td></td>
</tr>
<tr>
<td>RowHeight</td>
<td>Height of every row.</td>
<td></td>
</tr>
<tr>
<td>Selected</td>
<td>Not implemented.</td>
<td></td>
</tr>
<tr>
<td>SetFont</td>
<td>Shows a font selection dialog box.</td>
<td></td>
</tr>
<tr>
<td>SliderStep</td>
<td>Small and large steps of the slider. The steps are values between zero and one, corresponding to the range Min to Max.</td>
<td></td>
</tr>
<tr>
<td>String</td>
<td>String on object.</td>
<td></td>
</tr>
</tbody>
</table>
### Style
- **Style of a object.** See the `uicontrol` and `hwcontrol` functions for available styles.

### Tag
- **Stores a tag to identify the object.** Useful for `findobj` commands.

### TooltipString
- **Hint that appears when the mouse rests over the object.**

### Type
- **Type of object.**

### Units
- **Units for the `Position` property.** 'Normalized' is relative to the parent size in the range zero to one.

### UserData
- **Store user specified data in the object.** Any data type

### Value
- **Value of object.** For true/false objects (radio, checkbox, toggle) the value reflects the object state. For combo/listboxes, the value is the index of the selected item, 1-based. For slider, value is the position of the slider in the range zero to one. For grid, value is the matrix to display on the grid.

### Visible
- **Object visibility.** 'on' or 'off'

### See Also
- `hwcontrol`, `uimenu`

---

#### uimenu

**Create a menu or submenu**

**Description**

`h=uimenu(p,'prop1','value1',...)` creates a submenu to menu `p` with the specified properties. If `p` is omitted, the menu is created for the current figure. Using the handle `h`, a series of menus can be created as needed.

**Properties**
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checked</td>
<td>True if menu is checked.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Children</td>
<td>Read only. Handles of the child of the object.</td>
<td>handles vector</td>
</tr>
<tr>
<td>CreateFcn</td>
<td>Function to call when the object is created.</td>
<td>callback string</td>
</tr>
<tr>
<td>DeleteFcn</td>
<td>Function to call when the object is deleted.</td>
<td>callback string</td>
</tr>
<tr>
<td>Kill</td>
<td>Kills the object.</td>
<td></td>
</tr>
<tr>
<td>Label</td>
<td>Menu string.</td>
<td>string</td>
</tr>
<tr>
<td>Parent</td>
<td>The object parent. Determines where the object will appear.</td>
<td>handle</td>
</tr>
<tr>
<td>Separator</td>
<td>Is menu item is a separator.</td>
<td>'on' or 'off'</td>
</tr>
<tr>
<td>Tag</td>
<td>Stores a tag to identify the object. Useful for findobj commands.</td>
<td>matrix</td>
</tr>
<tr>
<td>Type</td>
<td>Type of object.</td>
<td>string</td>
</tr>
<tr>
<td>UserData</td>
<td>Store user specified data in the object.</td>
<td>any data type</td>
</tr>
</tbody>
</table>

**See Also**

menu, uicontextmenu, uicontrol

- **uiresume** ............................ Continue after uiwait

  **Description**

  uiresume resumes the program execution after it was stopped by uiwait.

  **See Also**

  pause, uiwait, waitfor, waitforbuttonpress

- **uiwait** ............................. Pause until uiresume

  **Description**

  uiwait pauses program execution until uiresume is called.
See Also
pause, uiresume, waitfor, waitforbuttonpress

- umtoggle

  Toggles state of the menu item

  **Description**

  umtoggle(h) toggles the checked state of the menu item h.

  **See Also**

- vbaxes

  **Description**

  vbaxes(HWND hndl) creates axes on the VB form with handle hndl in VB programs. All further plots will appear on the specified window.

  **Example:**

  vbaxes Form1.hwnd

  **See Also**

  vbposition, winaxes

- vbrefresh

  **Description**

  vbrefresh(h) performs all queued windows events for the figure h.
See Also
refresh

• view ................................................. Change viewpoint

Description
view(az,el) changes the current 3-d viewpoint to azimuth az and elevation el.

See Also
contour,givens,mesh,plot3

• waitfor .......................... Wait for an object to change

Description
waitfor(h) waits until the object h is destroyed, or until the containing figure is closed.

See Also
uiresume,uwait,waitforbuttonpress

• waitforbuttonpress ... Wait for keyboard or mouse click

Description
a=waitforbuttonpress waits until a key was pressed, where a=1, or the mouse was clicked, where a=0. The 'CurrentCharacter' property determines which character was pressed or the 'CurrentPoint' property to determine where the mouse was clicked. In case of a mouse click, cmd’CurrentCharacter’ is the button number which was pressed.
See Also
uiresume, uwait, waitfor

- **watchoff**  ................Set mouse pointer to arrow

Description

watchoff(fig) sets the mouse pointer to arrow in the figure fig. If fig is omitted, the current figure is assumed.

See Also
watchon

- **watchon**  .................Set mouse pointer to hourglass

Description

watchon(fig) sets the mouse pointer to hourglass in the figure fig. If fig is omitted, the current figure is assumed.

See Also
watchoff