MathWorks®

MATLAB® Basic Functions Reference

MATLAB Environment

`clc`	Clear command window
`help fun`	Display in-line help for `fun`
`doc fun`	Open documentation for `fun`
`load("filename","vars")`	Load variables from `.mat` file
`uiimport("filename")`	Open interactive import tool
`save("filename","vars")`	Save variables to file
`clear item`	Remove items from workspace
`examplescript`	Run the script file named `examplescript`
`format style`	Set output display format
`ver`	Get list of installed toolboxes
`tic, toc`	Start and stop timer
`Ctrl+C`	Abort the current calculation

Operators and Special Characters

`+, -, *, /`	Matrix math operations
`.*, ./`	Array multiplication and division (element-wise operations)
`^, .^`	Matrix and array power
`\`	Left division or linear optimization
`.', '`	Normal and complex conjugate transpose
`==, ~=, <, >, <=, >=`	Relational operators
`&&, ||, ~, xor`	Logical operations (AND, OR, NOT, XOR)
`;`	Suppress output display
`...`	Connect lines (with break)
`% Description`	Comment
`'Hello'`	Definition of a character vector
`"This is a string"`	Definition of a string
`str1 + str2`	Append strings

Special Variables and Constants

`ans`	Most recent answer
`pi`	π=3.141592654…
`i, j, 1i, 1j`	Imaginary unit
`NaN, nan`	Not a number (i.e., division by zero)
`Inf, inf`	Infinity
`eps`	Floating-point relative accuracy

Defining and Changing Array Variables

`a = 5`	Define variable a with value 5

`A = [1 2 3; 4 5 6]`
`A = [1 2 3`
`     4 5 6]`
	Define A as a 2x3 matrix
	"space" separates columns
	";" or new line separates rows

`[A,B]`	Concatenate arrays horizontally
`[A;B]`	Concatenate arrays vertically
`x(4) = 7`	Change 4th element of x to 7
`A(1,3) = 5`	Change A(1,3) to 5
`x(5:10)`	Get 5th to 10th elements of x
`x(1:2:end)`	Get every 2nd element of x (1st to last)
`x(x>6)`	List elements greater than 6
`x(x==10)=1`	Change elements using condition
`A(4,:)`	Get 4th row of A
`A(:,3)`	Get 3rd column of A
`A(6, 2:5)`	Get 2nd to 5th element in 6th row of A
`A(:,[1 7])=A(:,[7 1])`	Swap the 1st and 7th column
`a:b`	[a, a+1, a+2, …, a+n] with a+n≤b
`a:ds:b`	Create regularly spaced vector with spacing ds
`linspace(a,b,n)`	Create vector of n equally spaced values
`logspace(a,b,n)`	Create vector of n logarithmically spaced values
`zeros(m,n)`	Create m x n matrix of zeros
`ones(m,n)`	Create m x n matrix of ones
`eye(n)`	Create a n x n identity matrix
`A=diag(x)`	Create diagonal matrix from vector
`x=diag(A)`	Get diagonal elements of matrix
`meshgrid(x,y)`	Create 2D and 3D grids
`rand(m,n), randi`	Create uniformly distributed random numbers or integers
`randn(m,n)`	Create normally distributed random numbers

Complex Numbers

`i, j, 1i, 1j`	Imaginary unit
`real(z)`	Real part of complex number
`imag(z)`	Imaginary part of complex number
`angle(z)`	Phase angle in radians
`conj(z)`	Element-wise complex conjugate
`isreal(z)`	Determine whether array is real

MathWorks®

Elementary Functions

`sin(x), asin`	Sine and inverse (argument in radians)
`sind(x), asind`	Sine and inverse (argument in degrees)
`sinh(x), asinh`	Hyperbolic sine and inverse (arg. in radians)
Analogous for the other trigonometric functions: `cos`, `tan`, `csc`, `sec`, and `cot`	
`abs(x)`	Absolute value of x, complex magnitude
`exp(x)`	Exponential of x
`sqrt(x), nthroot(x,n)`	Square root, real nth root of real numbers
`log(x)`	Natural logarithm of x
`log2(x), log10`	Logarithm with base 2 and 10, respectively
`factorial(n)`	Factorial of n
`sign(x)`	Sign of x
`mod(x,d)`	Remainder after division (modulo)
`ceil(x), fix, floor`	Round toward +inf, 0, -inf
`round(x)`	Round to nearest decimal or integer

Tables

`table(var1,...,varN)`	Create table from data in variables var1, ..., varN
`readtable("file")`	Create table from file
`array2table(A)`	Convert numeric array to table
`T.var`	Extract data from variable var

`T(rows,columns),`
`T(rows,["col1","coln"])`
	Create a new table with specified rows and columns from T

`T.varname=data`	Assign data to (new) column in T
`T.Properties`	Access properties of T
`categorical(A)`	Create a categorical array
`summary(T), groupsummary`	Print summary of table
`join(T1, T2)`	Join tables with common variables

Tasks (Live Editor)

Live Editor tasks are apps that can be added to a live script to interactively perform a specific set of operations. Tasks represent a series of MATLAB commands. To see the commands that the task runs, show the generated code.

Common tasks available from the Live Editor tab on the desktop toolstrip:

- Clean Missing Data
- Find Change Points
- Remove Trends
- Clean Outlier
- Find Local Extrema
- Smooth Data

Plotting

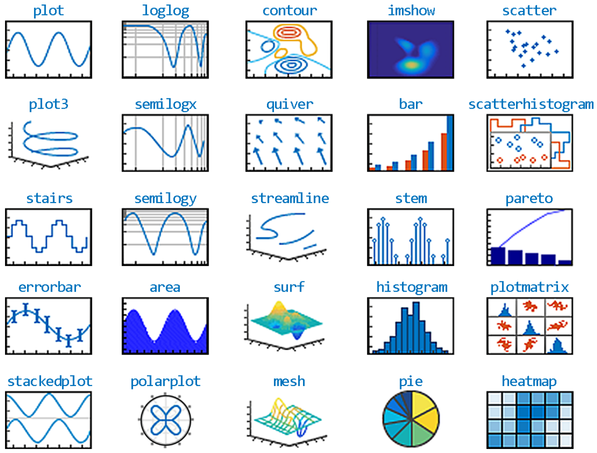
`plot(x,y,LineSpec)`
Line styles:
`-, --, :, -.`
Markers:
`+, o, *, ., x, s, d`
Colors:
`r, g, b, c, m, y, k, w`
	Plot y vs. x
	(`LineSpec` is optional)
	`LineSpec` is a combination of `linestyle`, `marker`, and `color` as a string.
	Example: `"-r"`
	= red solid line without markers

`title("Title")`	Add plot title
`legend("1st", "2nd")`	Add legend to axes
`x/y/zlabel("label")`	Add x/y/z axis label
`x/y/zticks(ticksvec)`	Get or set x/y/z axis ticks
`x/y/zticklabels(labels)`	Get or set x/y/z axis tick labels
`x/y/ztickangle(angle)`	Rotate x/y/z axis tick labels
`x/y/zlim`	Get or set x/y/z axis range
`axis(lim), axis style`	Set axis limits and style
`text(x,y,"txt")`	Add text
`grid on/off`	Show axis grid
`hold on/off`	Retain the current plot when adding new plots

`subplot(m,n,p),`
`tiledlayout(m,n)`
	Create axes in tiled positions

`yyaxis left/right`	Create second y-axis
`figure`	Create figure window
`gcf, gca`	Get current figure, get current axis
`clf`	Clear current figure
`close all`	Close open figures

Common Plot Types



Plot Gallery: *mathworks.com/products/matlab/plot-gallery*

Programming Methods

Functions

```
% Save your function in a function file or at the end
% of a script file. Function files must have the
% same name as the 1st function
function cavg = cumavg(x) %multiple args. possible
    cavg=cumsum(vec)./(1:length(vec));
end
```

Anonymous Functions

```
% defined via function handles
fun = @(x) cos(x.^2)./abs(3*x);
```

Control Structures

`if, elseif` Conditions

```
if n<10
    disp("n smaller 10")
elseif n<=20
    disp("n between 10 and 20")
else
    disp("n larger than 20")
```

Switch Case

```
n = input("Enter an integer: ");
switch n
    case -1
        disp("negative one")
    case {0,1,2,3} % check four cases together
        disp("integer between 0 and 3")
    otherwise
        disp("integer value outside interval [-1,3]")
end % control structures terminate with end
```

For-Loop

```
% loop a specific number of times, and keep
% track of each iteration with an incrementing
% index variable
for i = 1:3
    disp("cool");
end % control structures terminate with end
```

While-Loop

```
% loops as long as a condition remains true
n = 1;
nFactorial = 1;
while nFactorial < 1e100
    n = n + 1;
    nFactorial = nFactorial * n;
end % control structures terminate with end
```

Further programming/control commands

`break`	Terminate execution of for- or while-loop
`continue`	Pass control to the next iteration of a loop
`try, catch`	Execute statements and catch errors

Numerical Methods

`fzero(fun,x0)`	Root of nonlinear function
`fminsearch(fun,x0)`	Find minimum of function
`fminbnd(fun,x1,x2)`	Find minimum of fun in [x1, x2]
`fft(x), ifft(x)`	Fast Fourier transform and its inverse

Integration and Differentiation

`integral(f,a,b)`	Numerical integration (analogous functions for 2D and 3D)
`trapz(x,y)`	Trapezoidal numerical integration
`diff(X)`	Differences and approximate derivatives
`gradient(X)`	Numerical gradient
`curl(X,Y,Z,U,V,W)`	Curl and angular velocity
`divergence(X,..,W)`	Compute divergence of vector field
`ode45(ode,tspan,y0)`	Solve system of nonstiff ODEs
`ode15s(ode,tspan,y0)`	Solve system of stiff ODEs
`deval(sol,x)`	Evaluate solution of differential equation
`pdepe(m,pde,ic,... bc,xm,ts)`	Solve 1D partial differential equation
`pdeval(m,xmesh,... usol,xq)`	Interpolate numeric PDE solution

Interpolation and Polynomials

`interp1(x,v,xq)`	1D interpolation (analogous for 2D and 3D)
`pchip(x,v,xq)`	Piecewise cubic Hermite polynomial interpolation
`spline(x,v,xq)`	Cubic spline data interpolation
`ppval(pp,xq)`	Evaluate piecewise polynomial
`mkpp(breaks,coeffs)`	Make piecewise polynomial
`unmkpp(pp)`	Extract piecewise polynomial details
`poly(x)`	Polynomial with specified roots x
`polyeig(A0,A1,...,Ap)`	Eigenvalues for polynomial eigenvalue problem
`polyfit(x,y,d)`	Polynomial curve fitting
`residue(b,a)`	Partial fraction expansion/decomposition
`roots(p)`	Polynomial roots
`polyval(p,x)`	Evaluate poly p at points x
`polyint(p,k)`	Polynomial integration
`polyder(p)`	Polynomial differentiation

MathWorks®

Accelerating the pace of engineering and science

Matrices and Arrays

`length(A)`	Length of largest array dimension
`size(A)`	Array dimensions
`numel(A)`	Number of elements in array
`sort(A)`	Sort array elements
`sortrows(A)`	Sort rows of array or table
`flip(A)`	Flip order of elements in array
`squeeze(A)`	Remove dimensions of length 1
`reshape(A,sz)`	Reshape array
`repmat(A,n)`	Repeat copies of array
`any(A), all`	Check if any/all elements are nonzero
`nnz(A)`	Number of nonzero array elements
`find(A)`	Indices and values of nonzero elements

Linear Algebra

`rank(A)`	Rank of matrix
`trace(A)`	Sum of diagonal elements of matrix
`det(A)`	Determinant of matrix
`poly(A)`	Characteristic polynomial of matrix
`eig(A), eigs`	Eigenvalues and vectors of matrix (subset)
`inv(A), pinv`	Inverse and pseudo inverse of matrix
`norm(x)`	Norm of vector or matrix
`expm(A), logm)`	Matrix exponential and logarithm
`cross(A,B)`	Cross product
`dot(A,B)`	Dot product
`kron(A,B)`	Kronecker tensor product
`null(A)`	Null space of matrix
`orth(A)`	Orthonormal basis for matrix range
`tril(A), triu`	Lower and upper triangular part of matrix
`linsolve(A,B)`	Solve linear system of the form AX=B
`lsqminnorm(A,B)`	Least-squares solution to linear equation
`qr(A), lu, chol`	Matrix decompositions
`svd(A)`	Singular value decomposition
`gsvd(A,B)`	Generalized SVD
`rref(A)`	Reduced row echelon form of matrix

Descriptive Statistics

`sum(A), prod`	Sum or product (along columns)
`max(A), min, bounds`	Largest and smallest element
`mean(A), median, mode`	Statistical operations
`std(A), var`	Standard deviation and variance

`movsum(A,n), movprod,`
`movmax, movmin,`
`movmean, movmedian,`
`movstd, movvar`
	Moving statistical functions
	n = length of moving window

`cumsum(A), cumprod,`
`cummax, cummin`
	Cumulative statistical functions

`smoothdata(A)`	Smooth noisy data
`histcounts(X)`	Calculate histogram bin counts
`corrcoef(A), cov`	Correlation coefficients, covariance
`xcorr(x,y), xcov`	Cross-correlation, cross-covariance
`normalize(A)`	Normalize data
`detrend(x)`	Remove polynomial trend
`isoutlier(A)`	Find outliers in data

Symbolic Math*

`sym x, syms x y z`	Declare symbolic variable
`eqn = y == 2*a + b`	Define a symbolic equation
`solve(eqns,vars)`	Solve symbolic expression for variable
`subs(expr,var,val)`	Substitute variable in expression
`expand(expr)`	Expand symbolic expression
`factor(expr)`	Factorize symbolic expression
`simplify(expr)`	Simplify symbolic expression
`assume(var,assumption)`	Make assumption for variable
`assumptions(z)`	Show assumptions for symbolic object

`fplot(expr), fcontour,`
`fsurf, fmesh, fimplicit`
	Plotting functions for symbolic expressions

`diff(expr,var,n)`	Differentiate symbolic expression
`dsolve(deqn,cond)`	Solve differential equation symbolically
`int(expr,var,[a, b])`	Integrate symbolic expression
`taylor(fun,var,z0)`	Taylor expansion of function

*requires Symbolic Math Toolbox