created by MIT 18.06 & improved by VinC

A Matlab Cheat-sheet

Basics:

`save 'file.mat'` save variables to *file.mat*
`load 'file.mat'` load variables from *file.mat*
`diary on` record input/output to file *diary*
`diary off` stop recording
`whos` list all variables currenly defined
`clear` delete/undefine all variables
`help command` quick help on a given *command*
`doc command` extensive help on a given *command*

Defining/changing variables:

`x = 3` define variable x to be 3
`x = [1 2 3]` set x to the 1×3 row-vector (1,2,3)
`x = [1 2 3];` same, but don't echo x to output
`x = [1;2;3]` set x to the 3×1 column-vector (1,2,3)
`A = [1 2 3 4;5 6 7 8;9 10 11 12];`
set A to the 3×4 matrix with rows 1,2,3,4 etc.
`x(2) = 7` change x from (1,2,3) to (1,7,3)
`A(2,1) = 0` change $A_{2,1}$ from 5 to 0

Arithmetic and functions of numbers:

`3*4, 7+4, 2-6 8/3` multiply, add, subtract, and divide numbers
`3^7, 3^(8+2i)` compute 3 to the 7th power, or 3 to the 8+2i power
`sqrt(-5)` compute the square root of –5
`exp(12)` compute e^{12}
`log(3), log10(100)` compute the natural log (ln) and base-10 log ($\log_{10}$)
`abs(-5)` compute the absolute value |–5|
`sin(5*pi/3)` compute the sine of 5π/3
`besselj(2,6)` compute the Bessel function $J_2(6)$

Arithmetic and functions of vectors and matrices:

`x * 3` multiply every element of x by 3
`x + 2` add 2 to every element of x
`x + y` element-wise addition of two vectors x and y
`A * y` product of a matrix A and a vector y
`A * B` product of two matrices A and B
`x * y` not allowed if x and y are two column vectors!
`x .* y` element-wise product of vectors x and y
`A^3` the square matrix A to the 3rd power
`x^3` not allowed if x is not a square matrix!
`x.^3` every element of x is taken to the 3rd power
`1:6` a row vector steps from 1 to 6 ([1,2,3,4,5,6])
`1:0.3:2` increases and steps of 0.3 until get up to 2 ([1, 1.3, 1.6, 1.9])
`cos(x)` the cosine of every element of x
`abs(A)` the absolute value of every element of A `exp(A)` e to the power of every element of A
`sqrt(A)` the square root of every element of A
`expm(A)` the matrix exponential e^A
`sqrtm(A)` the matrix whose square is A

Transposes and dot products:

`x.', A.'` the transposes of x and A
`x', A'` the complex-conjugate of the transposes of x and A
`x' * y` the dot (inner) product of two *column* vectors x and y
`dot(x,y), sum(x.*y)` …two other ways to write the dot product
`x * y'` the *outer* product of two *column* vectors x and y

Constructing a few simple matrices:

`rand(12,4)` a 12×4 matrix with uniform random numbers in [0,1)
`randn(12,4)` a 12×4 matrix with Gaussian random (center 0, variance 1)
`zeros(12,4)` a 12×4 matrix of zeros
`ones(12,4)` a 12×4 matrix of ones
`eye(5)` a 5×5 identity matrix I ("eye")
`eye(12,4)` a 12×4 matrix whose first 4 rows are the 4×4 identity
`linspace(1.2,4.7,100)` row vector of 100 equally-spaced numbers from 1.2 to 4.7
`7:15` row vector of 7,8,9,…,14,15
`1:0.3:2` increases and steps of 0.3 until get up to 2 ([1, 1.3, 1.6, 1.9])
`diag(x)` matrix whose diagonal is the entries of x (and other elements = 0)

Portions of matrices and vectors:

`x(2:12)` the 2nd to the 12th elements of x
`x(2:end)` the 2nd to the last elements of x
`x(1:3:end)` every third element of x, from 1st to the last
`x(:)` all the elements of x
`A(5,:)` the row vector of every element in the 5th row of A
`A(5,1:3)` the row vector of the first 3 elements in the 5th row of A
`A(:,2)` the column vector of every element in the 2nd column of A
`diag(A)` column vector of the diagonal elements of A

Solving linear equations:

`A \ b` for A a matrix and b a column vector, the solution x to $Ax=b$
`inv(A)` the inverse matrix A^{-1}
`[L,U,P] = lu(A)` the LU factorization PA=LU
`eig(A)` the eigenvalues of A
`[V,D] = eig(A)` the columns of V are the eigenvectors of A, and the diagonals `diag(D)` are the eigenvalues of A

Plotting:

`plot(y)` plot y as the y axis, with 1,2,3,… as the x axis
`plot(x,y)` plot y versus x (must have same length)
`plot(x,A)` plot columns of A versus x (must have same # rows)
`loglog(x,y)` plot y versus x on a log-log scale
`semilogx(x,y)` plot y versus x with x on a log scale
`semilogy(x,y)` plot y versus x with y on a log scale
`fplot(@(x) …expression…,[a,b])` plot some expression in x from x=a to x=b
`axis equal` force the x and y axes of the current plot to be scaled equally
`title('A Title')` add a title A Title at the top of the plot
`xlabel('blah')` label the x axis as blah
`ylabel('blah')` label the y axis as blah
`legend('foo','bar')` label 2 curves in the plot foo and bar
`grid` include a grid in the plot
`figure` open up a new figure window