San Diego State University
Matlab Reference Card
2/10/2003 RN

Introduction

MATLAB (short for "MATrix LABoratory") is a high performance software package for numerical computation and visualization. MATLAB is a complete programming environment that is easier to use and more powerful for numerical applications than traditional programming languages such as FORTRAN or C.

MATLAB Toolboxes are collections of application-specific functions. The following toolboxes are available: Compiler, Control, Communications, Fixed Point Blocks, Fuzzy Logic, Fixed-Point Blocks, Identification, Image, MATLAB Compiler, Neural Network, Optimization, PDE, Real Time, Signal, SIMULINK, Symbolic Math, Statistics.

Setup

MATLAB is available on Rohan and many PC labs on campus. To fully explore MATLAB's interactive graphic ability, you must be using an environment that supports graphical user interface, such as Xwindows. For help with Xwindows see:

http://www-rohan.sdsu.edu/help/xwindows.html

Starting and Quitting MATLAB

Start MATLAB by typing the following command at the UNIX command prompt:

matlab

Once running, MATLAB will present a pair of greater-than symbols, >>, as its command-line prompt. To end your MATLAB session, type *quit* at the prompt.

File Conventions

MATLAB expects certain file extensions when some commands are executed:

File Contents	*Extension*
MATLAB script file	.m
MATLAB binary file	.mat

MATLAB can read and execute commands from standard input, as well as from ASCII script files with a .m extension. These script files are called M-files, and can be created with an editor such as emacs, pico, vi, or MATLAB's built-in editor. Execute the commands in a file named myproject.m, by typing the file name without the .m extension at the MATLAB >> prompt:

myproject

The execute the file from the UNIX prompt by typing:

matlab < myproject.m > myproject.out

MATLAB executes each command from myproject, then returns to a prompt. As MATLAB treats any text following a % sign as a comment, you can use comments to document your work.

Help in MATLAB

An extensive help facility is available in MATLAB by typing *help*. MATLAB responds by displaying a list of help topics, along with a brief description of each topic. Typing *help* followed by a *topic* displays specific help on that topic. Example, typing *help plot* provides help on MATLAB's 2-D plotting command plot.

View a tour of MATLAB's features by typing *demo*.

The MATLAB Manuals are available at the Reserve Book Room in Love Library. A list of MATLAB Manuals is at:

http://www-rohan.sdsu.edu/matlab.html

Command Syntax

MATLAB is case-sensitive. All built-in commands are in lower case. A command normally terminates with a carriage return. Including a semicolon (**;**) at the end of a command suppresses MATLAB's echoing to the terminal (this is useful when dealing with large sets of numbers.)

Matrices

MATLAB works with essentially one kind of object: a matrix of numbers (which could include complex elements). Scalars are 1-by-1 matrices, while vectors are 1-by-n or n-by-1 matrices.

When entering a matrix, separate columns by spaces or commas; separate rows by semicolons. For example, typing:

A = [1 2; 3 4]

results in:

```
A =
     1     2
     3     4
```

MATLAB stores the above 2-by-2 matrix into the variable A for later use. To retrieve a variable, simply type its name (e.g., A).

Matrix and Array Operations

Matrix operations are fundamental to MATLAB, and are based on principles of linear algebra. Matrix multiplication, division, power, and transpose are denoted by the symbols *, /, ^, and.

Array operations refer to element-by-element arithmetic operations, rather than the usual linear algebraic operations. Array operations are denoted by preceding an operator with a period (.) such as .*, ./, and .^. Array and matrix operations are the same for both addition and subtraction. See the *Examples* section for a comparison of matrix and array operations.

Saving MATLAB Variables

Variables created in a MATLAB session can be saved to a file using the save command:

save myfile

saves all variables into a file called myfile.mat. Files with a .mat extension are in binary format, and are not readable as text. Saved variables can be retrieved by using the load command:

load myfile

Colon Notation

The colon (:) is an important notation. It can be used to generate vectors or access submatrices.

Examples

Example	*Description*
diary('try')	Save the text of a MATLAB session into file try (graphics excluded).
clear	Clear all variables from memory.
a = [1 2; 3 4]	Create a simple 2x2 matrix.
b = inv(a)	Invert matrix a and store it as variable b.
c = a * b	Multiply matrices a and b and store the result as variable c.
c = a .* b	Element by element multiply.
c(:, 2)	The 2nd column of matrix c.
c(2, :)	The 2nd row of matrix c.
q = [a b]	Combine matrices a and b and store the result as variable q.
who	List currently active variables.
x = 0 : pi/30 : 4*pi	Use colon notation to create a row vector whose elements range from 0 to 4 pi, in increments of pi/30.
x = x'	Take the transpose of x.
y = exp(-0.3 *x).*sin(x)	Create column vector y as a function of column vector x. Note the .* operator!
hold on	hold plot so additional plots will overlay this plot.
plot(x,y)	Plot the column vectors x and y.
title('Decaying Sinusoid')	Create a plot title.
xlabel('Time (sec)')	Label the x-axis.
ylabel('Position (in)')	Label the y-axis.
print -dps decay	Create PostScript plot file decay.ps.
clf	Clear the displayed figure.
[x, y] = meshgrid(-1: 0.05:1 , -1: 0.1 : 1)	Generate matrices x and y to support 3D surface plotting over the provided intervals.
size(x)	Display the dimensions of matrix x.
z = sin(5 * x .* y);	Create matrix z from matrices x and y.
surf(x,y,z)	Plot the surface z = sin(5xy).
view([20 60])	Change the view and replot (azimuth & elevation).
diary off	Turn off diary.

MATLAB's command language is expressive, extensible, and easy to use. Using MATLAB interactively by entering commands at its >> prompt is an effective way to learn.

MATLAB Toolboxes

Functions in MATLAB toolboxes can be used in the same way as regular MATLAB functions. To display all available toolboxes, type the following from your MATLAB prompt:

dir $TOOLBOX

Find all the functions in a given toolbox by typing:

help toolbox-name

Where *toolbox-name* is replaced by the name of the toolbox you wanted help with. For example:

help signal

displays help on the signal toolbox.

Documentation

MATLAB comes with an extensive set of online documentation that can be viewed by typing *helpdesk* from the MATLAB prompt. The UNIX man page description of MATLAB can be seen by typing *man matlab* at the UNIX prompt.

Additional Help

More information on MATLAB, including online documentation, examples and tutorials, is at:

http://www-rohan.sdsu.edu/help/matlab.html