How to Use MATLAB

What is MATLAB

MATLAB is an interactive package for numerical analysis, matrix computation, control system design and linear system analysis and design. On the server "bass", MATLAB version 4.0 is available. In addition to the standard functions provided by MATLAB, there are seven "toolboxes", or collections of functions and procedures, available as part of the MATLAB package:

- Control System Toolbox.
- System Identification Toolbox.
- Neural Network Toolbox.
- Optimization Toolbox.
- Robust Control Toolbox.
- Signal Processing Toolbox.
- Spline Toolbox.
- SIMULINK

Getting Started

To start MATLAB on any Unix workstation or X-terminal, type **matlab**. A >> prompt will appear to indicate that you are in MATLAB. To terminate MATLAB, type **quit**. Notice that MATLAB is case sensitive, so all commands must be entered in lower case.

Online Help

During any MATLAB session, online help for a variety of topics is available. To see a list of help topics, type **help**. For help on a specific topic, type **help** followed by the directory/topic name (e.g. **help matlab/general** for General purpose commands), which will give you all command name in the topic. For help on a specific command, type **help command_name** (e.g. **help who**).

General Purpose Commands

diary – Save text of MATLAB session.

Once a **diary** command is issued, all subsequent commands and output are written to a transcript file. **diary** <*fname*> open a transcript file named fname. **diary off** suspends it. **diary on** turns it back on. Close the transcript file using the **diary off** command at the end of the session.

save – Save workspace variables on disk.

save <*fname*> saves all workspace variables to the binary "MAT-file" named fname.mat. The data may be retrieved with command **load** <*fname*>. **save** <*fname* > **X Y Z** save only variables X, Y, and Z. Notice that if you only want to save several variables, make sure to put variable names after command **save** <*fname*>.

who – Lists the variables in the current workspace.

clear – Clear variables from the workspace.

clear removes all variables from the workspace. **clear X** removes variable or function X from the workspace.

size – Check size of matrix.

size(X), for M-by-N matrix X, returns (M,N) containing the number of rows and columns in the matrix.

format – Set output format.

All computations in MATLAB are done in double precision. The format of the displayed output can be controlled by the following commands.

format short	fixed point with 4 decimal places (the default)
format long	fixed point with 14 decimal places
format short e	scientific notation with 4 decimal places
format long e	scientific notation with 15 decimal places

! –Execute operating system command.

During MATLAB session, Unix commands can be executed by using **!***Unix- command*

Defining Variables

The basic variable types in MATLAB are scalar, vector, and matrix. Variables are defined by typing the name of the variable followed by an equal sign and the value of the variable. The values of a vector or matrix are enclosed by brackets, and rows of matrices are separated by semicolons. For example:

A=1	let a equal a scalar
B=[1 2 3]	let b equal a vector
C=[1 2 3;4 5 6;7 8 9]	let c equal a matrix

Matrix entry shortcuts

MATLAB provides several commands to create commonly used matrices.

A=eye(3)	create a 3×3 identity matrix
B=ones(3)	create a 3×3 matrix of ones
C=zeros(3)	create a 3×3 zero matrix
D=diag([1 2 3])	create a matrix whose diagonal is 1 2 3

Matrices with regularly increasing or decreasing values can be created as follows:

E=[1:5]	create a vector with entries 1 2 3 4 5
F=[1:0.1:2]	create a vector with entries from 1 to 2 incremented by 0.1

Displaying Data

The values of a variable can be displayed by simply typing the variable name. In matrix case, a specific element, row or column of a matrix can be examined separately. A single element of a matrix can be addressed by typing the name of the matrix followed by the row and column indices of the element in parentheses. To address an entire column or row, a colon can be substituted for either index. For example:

A(1,1)	display the element in the 1st row, 1st column of A
A(:,1)	display the 1st column of A
A(1,:)	display the 1st row of A

Matrix Arithmetic and Functions

The following matrix operations are available in MATLAB:

$+$	addition,	e.g. A+B
$-$	subtraction,	e.g. A-B
$*$	multiplication,	e.g. A*B
^	power,	e.g. A^2
$'$	transpose,	e.g. A'
$\backslash$	left division,	e.g. A\b
/	right division,	e.g. b/A

The "matrix division" operators have following meanings:
$x = A \setminus b$ is the solution of $Ax = b$ and,
$x = b/A$ is the solution of $xA = b$

MATLAB is able to calculate simple functions. Some examples are given below:

det(A)	calculate the determinant of A
inv(A)	calculate the inverse of A
rank(A)	calculate the rank of A

Conditional and looping commands

The format of the **for, while, and if** commands is similar to the format of most computer languages. For example, the statement

```
for i =1:n
      x(i)=i^2;
end
```

will produce a n-dimension vector $x = [1^2, 2^2, ..., n^2]$.

The general form of a **while** loop is

> **while** relation
>   statements
> **end**

The statements will be repeatedly executed as long as the relation remains true. For example, for a given number a, the following will compute the smallest nonnegative integer n such that $2^n \geq a$:

```
n= 0;
while 2^n < a
    n=n+1;
end
```

The general form of an **if** statement is illustrated by

```
if n < 0
  x(n)=0;
elseif rem(n,2)==0
  x(n)=2;
else
  x(n)=1;
end
```

The relational operators in MATLAB are

<	less than
>	greater than
<=	less than or equal
>=	greater than or equal
==	equal
~=	not equal

Note that "=" is used in an assignment statement while "==" is used in a relation. Relations may be connected or quantified by the logical operators

& and; — or; ~ not;

Graphics

plot

The **plot** command creates linear x-y plots;if x and y are vectors of the same length, the command **plot(x,y)** open a graph window and draws an x-y plot of the elements of x versus the elements of y. You can, for example, draw the graph of the sine function over the interval -4 to 4 with the following commands:

```
x= -4:.01:4;
y=sin(x);
plot(x,y)
```

print

A hardcopy of the graph window can be obtained with the MATLAB command **print**. The command **print filename** saves the current graph window in PostScript form to the designated filename, overwriting it if it already exists. The filename extension ".ps" is appended to the filename if an extension is not supplied.

mesh

Three dimensional mesh surface plots are drawn with the function **mesh**. The command **mesh(z)** creates a three-dimensional plot of the elements of the matrix z. The mesh surface is defined by the z-coordinates of points above a rectangular grid in the x-y plane.

M-files

MATLAB can execute a sequence of statement stored on diskfiles. Such files are called "M-files" because they must have ".m" as extension name (e.g. filename.m). A M-files consists of a sequence of normal MATLAB statements. If the file has the filename, say, test.m, then the MATLAB command **test** will cause the statements in the file to be executed. Variables in a M-file are global and will change the value of the environment.

In an M-file the user can be prompted to interactively enter input data with the function **input**. When, for example, the statement

```
iter=input('Enter the number of iterations: ')
```

is encountered, the prompt message is displayed and execution pauses while the user keys in the input data. Upon pressing the return key, the data is assigned to the variable iter and execution resumes.

Text strings can be displayed with the function **disp**. For example:

```
disp('this message is hereby displayed')
```

All editing tools on Unix machine, such as **vi** and **emac**, can be used to edit M-file.