

MATLAB Primer for MLSC

Sethu Vijayakumar

Tuesday, January 06, 2004

In general, if you are unfamiliar with MATLAB, it may be useful to work with a friend initially to get used to the MATLAB software. MATLAB is fairly easy to use, and in our course we will only need a very small subset of the available commands. The library and the bookstore have literature on Matlab. Note that Matlab is REALLY a great tool, not just for the course, but also for any kind of numerical analysis of data. Also note that MATLAB has nothing to do with Mathematica: Mathematica is great for symbolic manipulations and pretty slow for numerics, while MATLAB is great (and fast) at numerics, but it has no powerful symbol manipulation. MATLAB means MATRIX-Laboratory, not MATH-Laboratory!

1 Starting Matlab

On DICE machines, type `>matlab`. On PCs or Macs double click the MATLAB application.

2 Demos

Matlab comes with a variety of demos that help to get started. In general, matlab provides help for every function by just typing `"help <function_name>"`. Try `"help demos"` as a first attempt. This lists a variety of demos that introduce matlab. Try `"intro"` to learn about some of the very basics of matlab. Try `"graf2d"` to see some of the plotting abilities of matlab.

3 Scripts and Functions

Matlab allows you to write functions and scripts. We will only need scripts in the course. Scripts are just matlab commands written to a file. The file should have the suffix `".m"` to make matlab recognize it is a matlab file. The file must also reside in your current working directory. Use the `"cd"` command (like in DOS or UNIX) to change the directory. Use `"pwd"` to find out in which directory you currently are.

In general, matlab commands are terminated by a semicolon after each command.

4 Reading Data Files

Use the `"load"` command to read the data files provided for your homework. For example:

```
load X.data -ascii
```

will load the data in the file `X.data` and interpret the data as ASCII data during loading. You will find the data under the variable `X` afterwards. Try the `"whos"` command to see all variables. All variables in matlab are treated as matrices, vectors, or scalars. There are also structures like in C, but this is not relevant for our course.

You can display variables by avoiding the semicolon after a command. For e.g., just typing

```
X
```

will write the entire matrix to the standard output (you may not want to do this since the matrix X is quite long).

5 Manipulating Matrices

The “intro” demo gives a good idea of what you can do with matrices. Here are some more helpful hints.

```
X(3,4)
```

accesses element (3,4) (i.e., 3rd row, 4th column).

```
X(:,4)
```

accesses the fourth column entirely (i.e., this is a vector).

```
X(3,:)
```

accesses the third row entirely (i.e., this is a row vector).

```
X(3,4:8)
```

accesses the third row, columns 4 to 8 etc. I assume you get the pictures how to get elements or blocks of matrices.

```
X=[]
```

clears a variable and sets its content to “empty”. You can also clear individual rows and columns from a matrix or vector, but note that the indices will shift appropriately, and what used to be X(4,3) may now be somewhere else.

6 Generating Vectors

A vector can be generated, for example, by:

```
x=(0.4:0.05:0.8)';
```

This creates a vector whose first element is 0.4, the second is 0.45, etc., until the last element 0.8. To make it a column vector, the transpose operator ' was added. This way of creating vectors is very useful to create the centers of a histogram, and then the vector can be used in the “hist” command to make matlab calculate the histogram for you. Note that the histogram is NOT normalized; you need to do this manually.

7 Linear Algebra

Matlab allows you to just multiply matrices and vectors in the most straightforward way by using *,+,-,/ operators as in other programming languages. Note that you must make sure that you use the algebra operators appropriately. This mean, you can multiply a matrix by a scalar, but only by a vector if the vector has the right length and “orientation” (i.e., row or columns format). Matlab automatically knows whether something is a scalar or a matrix or a vector -- there are no naming conventions. But a good method is to use capital letters for matrices, and small letters for vectors, and some special small letter variables for scalars (e.g., “s”).

8 Programming a Loop

In a script file, a loop would be programmed like:

```
n=length(X); % the percent sign is a comment. The function
              % nlength returns the length of a variable; more
              % general is actually the nsize command, which
              % returns number of rows and number of columns
              % of a matrix.

for i=1:n,
    sprintf('This is the %d iteration\n',i)
end
```

The above statement would loop from 1 to n and print the indicated sentence. Note that matlab has many commands very similar to C. The most important difference is that character strings are in **** single**** quotes instead of double quotes.

9 Generating Special Matrices

Note that a vector is an impoverished matrix, and everything stated about matrices is also true for vectors. Examples of special matrices are:

```
M=ones(3,4)
```

creates a 3x4 matrix with only 1's in the matrix coefficients.

```
M=zeros(6,7)
```

creates a 6x7 matrix with only 0's in the matrix coefficients.

```
M=rand(6,7)
```

creates a 6x7 matrix with uniformly distributed values between 0 and 1.

```
M=randn(6,7)
```

creates a 6x7 matrix with Gaussian distributed values with $N(0,1)$.

10 Graphs

Check the "graf2d" demo. Check the commands "plot", "plot3", "bar", "bar3", "hist", "surf", and "surfl" by executing "help <command>". Create some example matrices and play with the commands. A very useful command is the "hold on" and "hold off" command: it allows you to keep the content of a graph and plot another graph on top. Also nice is the "zoom on" and "zoom off" command: if zoom is enabled, you can drag rectangles in a graph to automatically enlarge the details in this rectangle. Double click resets the graph to original size.

11 Printing

Matlab can print every graphics window by executing "print". By default, the active graphics window is sent to the default printer. Check "help print" for available options. The "print" command can also save graphics to files, in pretty much all standard formats (e.g., postscript, etc.). The "figure" command can be used to make a window active or to create new windows (e.g., figure(6) creates window with number "6" if it does not exist yet, or make this window active if it does exist).

12 Saving Variables

Check the "save" command to learn how to save your entire matlab workspace or selected variables. Notice the options "-ascii" etc., to determine the format in which you save.