MAS 622j MATLAB® help

Originally written by Tom Minka, and modified by Yuan Qi and Ashish Kapoor Sept. 2002

Getting started

Open an editor window next to your MATLAB® window. You'll often find yourself mousing text in the editor window and pasting it into the matlab window.

If you're a complete MATLAB® novice, type "intro" or, if you're real fond of splashy colors, type "demo". You can also read help from matlab help menu.

You can also type "helpwin" or "helpdesk" to get a help window. The MATLAB® help command is "help". Try "help general" or just plain "help". The apropos command is "lookfor". Try, e.g. "lookfor tangent" or "lookfor random".

Syntax

The MATLAB® continuation code is "..." i.e. if you have a long formula that is several lines long, end each line with ... and continue the formula on the next line.

The MATLAB® comment character is "%". The semicolon ";" is useful too; it makes the command on that line operate silently. For example, "A = B;" copies matrix B into matrix A. "A = B" does the same thing, but prints out the whole contents of matrix B while copying.

Data structures

The most useful data strucutre in MATLAB® is matrix. Scalars is considered 1 by 1 matrices, and strings (delimited by 'single quotes') are considered vectors (which in turn are just skinny matrices). Check out

```
  sprintf  sscanf  num2str  int2str
```

MATLAB®5 and 6 added the very useful "cell array" and "struct" data types. A cell array is just like a matrix except each entry can be any data type, not just a number. For example:

```
>> c = {'joe' 5 [1 2 3]}

c =

    'joe'    [5]    [1x3 double]

>> c{3}

ans =

     1     2     3
```

A struct is similar except its contents are addressable by name only.

```
>> s = struct('name', 'joe', 'age', 30)

s =

    name: 'joe'
     age: 30

>> s.name

ans =

joe

>> s.age

ans =

    30
```

Plotting and I/O

MATLAB® is very good at plotting your data. See our plotting examples and/or get MATLAB®'s help on:

```
  plot  grid  hold  drawnow  axis  axes  orient
  subplot  mesh  meshgrid  plot3  rotate3d
```

Then you'll want to print your plot to a file. To print the current plot to a postscript file, type

```
  orient tall   %% this line is optional
  print -deps myfilename.ps
```

You can use C style file I/O. Get help on commands

```
  fopen  fclose  fscanf  fprintf  printf
```

The MATLAB® parser is not too clever, so don't put "-" in a filename; poor MATLAB® will think you want to subtract!

MATLAB® has a native data file format, plus it can save and load data from ASCII files. See the stock answers and check out

```
  save filename.dat -ascii
  load filename
```

Subroutines and objects

Yes, you can write subroutines in MATLAB®. Each subroutine lives in its own file, with a name ending with .m, and you call it by filename. MATLAB® will sometimes not notice that you have created a new subroutine. Use `path(path)` to make it rescan the directories.

In MATLAB®, subroutines can be associated with specific classes to simulate "methods" for object-orientation. To make a method for class `myclass`, all you have to do is put the subroutine in a subdirectory called `@myclass`.

You must always have a subroutine in `@myclass` which is just called `myclass`. This subroutine is called to create instances of the class. Get MATLAB®'s help on `class`. If you redefine a class, e.g.

by editing `@myclass/myclass`, MATLAB® will complain unless you clear the old definition with `clear myclass`. Don't ask me why.

`clear x` can also be used to remove the variable *x* from the workspace, e.g. to reclaim memory. Careful: `clear` alone removes all variables!

Thomas P. Minka