MATLAB Reference Sheet

Variables

If you want to...	MATLAB Command	Comment
Create a variable called `a` and set it equal to 1	`a = 1`	Anytime you use `a` from now on, unless you change its value or clear it, MATLAB knows you mean 1. Note that MATLAB is case sensitive, so the variables `a` and `A` are not the same.
Find out what variables have been declared	`who` or `whos`	`who` returns a simple list of variable names. `whos` also lists the sizes and types of each variable.
Clear the variable `a`	`clear a`	MATLAB will now no longer know what `a` means.
Clear all variables	`clear`	

Basic Commands

If you want to...	MATLAB Command	Comment
Get the previous result	`ans`	So if the previous command was `12/5`, `ans` will contain 2.4000.
Get the previous command	Up arrow	Using the up arrow allows you to cycle through all previous commands. There is also a list of previous commands in the Command History window at the bottom right of the screen.
Show more or fewer decimal places	`format long` or `format short`	`format long` switches to 15 decimal places; `format short` switches back to the default of 4. More formatting options are here.
Hide the line output	Semicolon at the end	So `a=12/5` shows the result `a = 2.4000` but `a=12/5;` does not. This also allows you to perform multiple commands in a single line (e.g. `a=2;b=3;c=a*b` returns `c = 6`).
Clear the command window	`clc`	This does not clear the variables; it just clears the previous entries and output.
Create a row vector	`a=[0 1 5]` or `a=[0,1,5]`	
Create a column vector	`a=[0;1;5]`	
Create a vector with uniform spacing	`v1=0:0.25:1`	This returns the vector `v1 = [0 0.25 0.5 0.75 1]`. If you omit the spacing, MATLAB uses 1 (so `d=1:4` returns `d = [1 2 3 4]`).
Sum the elements of a vector	`sum(v1)`	
Access the 3rd component of `v1`	`v1(3)`	This returns 0.5 for `v1` defined above. This syntax works no matter whether `a` is a row or column vector.
Get the size (dimensions) of `v1`	`size(v1)`	For the vector `v1` above, which has one row and five columns, `size(v1)` is `[1 5]`.

Constants

Constant	MATLAB Syntax	Comment
i	`i` or `j`	i here is the imaginary unit $\sqrt{-1}$. So entering `(1+i)*(1+i)` in MATLAB yields the answer `0 + 2.0000i`
π	`pi`	
e	`exp(1)`	e here is the exponential (2.7183...)
∞	`Inf`	This is the result that you get if you enter `1/0` in MATLAB.
Not a Number	`NaN`	This is the result that you get if you enter `0/0` in MATLAB.

Basic Operators and Functions

To Calculate	MATLAB Syntax	Comment
$a+b$ or $a-b$	`a+b` or `a-b`	These work no matter whether `a` and `b` are scalars or vectors.
ab (a multiplied by b)	`a*b` or `a.*b`	`a*b` works if `a` and `b` are scalars (single numbers). Use `a.*b` if `a` and `b` are vectors and you want to multiply corresponding elements. For example, if `a=[1 2 3]` and `b=[0 1 2]`, `a*b` returns an error but `a.*b = [0 2 6]`.
a/b	`a/b` or `a./b`	`a/b` works if `a` and `b` are scalars (single numbers). Use `a./b` if `a` and `b` are vectors and you want to divide corresponding elements (similar to multiplication, above).
a^2	`a^2` or `a.^2`	`a^2` works if `a` is a scalar (single number) but if you want to take powers of all the elements of a vector, you must use `a.^2`. For example, if `a=[1 2 3]`, `a^2` returns an error but `a.^2 = [1 4 9]`.
$\sqrt{a}$	`sqrt(a)`	Works for both scalars and vectors. Use powers to calculate more complex roots. So for the cube root of a use `a.^(1/3)`.
$\lvert a\rvert$ (absolute value of a)	`abs(a)`	Works for both scalars and vectors.
$\sin a$	`sin(a)`	Works for both scalars and vectors. All other trig functions (cos, tan, sec, csc, cot) use analogous Matlab syntax.
$\sin^{-1} a$ (or $\arcsin a$)	`asin(a)`	Works for both scalars and vectors. All other inverse trig functions use analogous Matlab syntax (`acos`, `atan`, etc).
e^a	`exp(a)`	e here is the exponential (2.7183...). Works for both scalars and vectors.
$\ln a$	`log(a)`	Works for both scalars and vectors. Note that `log` in MATLAB means the natural (base e) logarithm. To take a base 10 logarithm (commonly written $\log a$) use the command `log10`.

Vector Manipulation

To Calculate	MATLAB Syntax	Comment
$\|\|\mathbf{v}\|\|$	`norm(v)`	This is the norm/length/magnitude of the vector.
$\mathbf{a} \cdot \mathbf{b}$	`dot(a,b)`	
$\mathbf{a} \times \mathbf{b}$	`cross(a,b)`	

Matrix Manipulation

To Calculate	MATLAB Syntax	Comment
Define a matrix	`A=[1 2; 3 4]`	Same as vectors in that semicolons separate rows; spaces or commas separate entries in a row.
Access entry (i, j) of A	`A(i,j)`	So for `A` above, `A(2,1)` would return 3.
$A + B$ or $A - B$	`A+B` or `A-B`	
AB (matrix multiplication)	`A*B`	Note that this is different than element-by-element multiplication `A.*B`.
Add 1 to each element of A	`A+1`	Note that this is simply a MATLAB shorthand - in normal notation, you cannot add a scalar to a matrix.
A^2	`A^2`	Different from element-by-element powers `A.^2`; this is the same as the matrix multiplication `A*A`.
$n \times n$ identity matrix	`eye(n)`	So `x=eye(3)` is the same as `x=[1 0 0;0 1 0;0 0 1]`.
$n \times m$ matrix of zeros or ones	`zeros([n m])` or `ones([n m])`	So `x=ones([2 3])` is the same as `x=[1 1 1;1 1 1]`.
A^T	`A.'`	`A'` (without the period) takes the Hermitian transpose (transpose with complex conjugate)
A^{-1}	`inv(A)`	
$\|A\|$ (determinant of A)	`det(A)`	

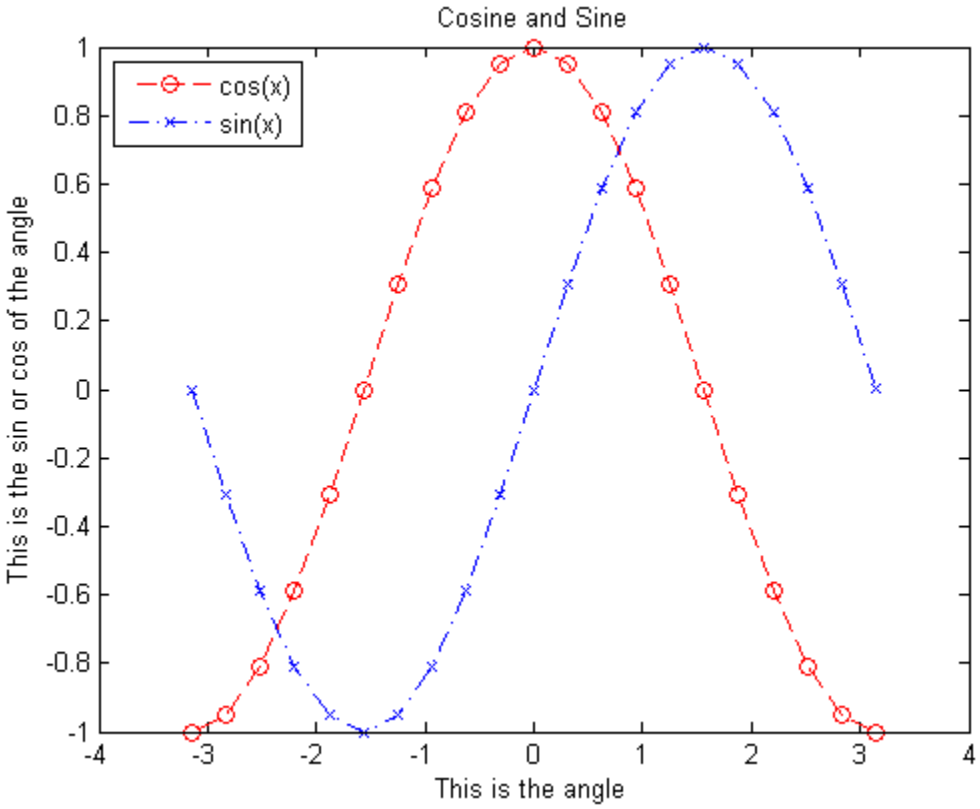
Basic Graphing

MATLAB Command	Use
`plot(x,y)`	Plots the values of vector `x` against the values of vector `y`.
`hold`	Freezes the current graph so that subsequent graphs will be plotted on top of it. Use `hold off` to turn this off.
`plot(x1,y1,x2,y2...)`	Plots vector `y1` versus vector `x1`, vector `y2` versus vector `x2`, etc on the same axes.
`plot(X1,Y1,LS1,X2,Y2,LS2...)`	Plots each vector `Yn` versus vector `Xn` on the same axes using the LineSpec LSn to format each curve. LineSpecs are short strings (e.g. `'-.or'`) that define the line style, marker symbol, and color for a given curve. More information on how to use LineSpec strings is available at the MathWorks website.
`axis([xmin xmax ymin ymax])`	Scales the graph so that the x-axis goes from `xmin` to `xmax` and the y-axis goes from `ymin` to `ymax`. Use `axis equal` to set the aspect ratio so that the data units are the same in every direction. More axis scaling options are described here.
`title('figure title')`	Places the title "figure title" above the graph.
`xlabel('axis text')` or `ylabel('axis text')`	Labels the x- or y-axis on the plot with "axis text".
`legend('curve1','curve2'...)`	Creates a legend using the labels `curve1`, `curve2`, etc. `...,'Location','Southeast')` can be inserted at the end to place the legend in the bottom-right corner, for example.
`clf`	Clears the figure window.
`close`	Closes the figure window.

A simple graphing example is provided on the next page.

A simple graphing example:

```
>> x=-pi:pi/10:pi; y1=cos(x); y2=sin(x); plot(x,y1,'--or',x,y2,'-.xb')
>> title('Cosine and Sine')
>> xlabel('This is the angle'); ylabel('This is the sin or cos of the angle')
>> legend('cos(x)','sin(x)','Location','Northwest')
```



Tutorials

MathWorks provides several good MATLAB tutorials, which can be found here. Their Getting Started with MATLAB video provides an excellent introduction to using MATLAB in just over five minutes. The first 3:30 should be sufficient for the beginning of the semester, as the last 90 seconds or so gets into concepts that we won't cover until later in the course.