MATLAB CHEAT SHEET

Throughout this document x and y will be either row or column vectors and A will always be a matrix.

Basics

`clc`	Clear command window
`clear`	Clear all variables
`clf`	Clear all plots
`close all`	Close all plots
`doc function`	Open help page for function
`% This is a comment`	Comments
`ctrl-c`	Abort the current operation
`format short`	Display 4 decimal places
`format long`	Display 15 decimal places
`disp('text')`	Print text
`doc mean`	Open documentation of command `mean`
`help mean`	Open help of command `mean`
`edit myFile`	Creates/opens myFile

Defining and Changing Variables

`a = 3`	Define variable a to be 3
`x = [1, 2, 3]`	Set x to be the row vector $[1, 2, 3]$
`x = [1; 2; 3]`	Set x to be the column vector $[1, 2, 3]^T$

`A = [1, 2, 3, 4;`
`5, 6, 7, 8;`
`9, 10, 11, 12]`
	Set A to be a 3×4 matrix

`x(2) = 7`	Change x from $[1, 2, 3]$ to $[1, 7, 3]$
`A(2,1) = 0`	Change $A_{2,1}$ from 5 to 0

Basic Arithmetic and Functions

`3*4, 7+4, 2-6, 8/3`	multiply, add, subtract and divide
`3^7`	Compute 3^7
`sqrt(5)`	Compute $\sqrt{5}$
`log(3)`	Compute $\ln(3)$
`log10(100)`	Compute $\log_{10}(100)$
`abs(-5)`	Compute $\lvert -5 \rvert$
`sin(5*pi/3)`	Compute $\sin(5\pi/3)$
`floor(3.8)`	Compute $\lfloor 3.8 \rfloor$

Constructing Matrices and Vectors

`zeros(12, 5)`	Make a 12×5 matrix of zeros
`ones(12, 5)`	Make a 12×5 matrix of ones
`eye(5)`	Make a 5×5 identity matrix
`eye(12, 5)`	Make a 12×5 identity matrix
`linspace(1.4, 6.3, 1004)`	Make a vector with 1004 elements evenly spaced between 1.4 and 6.3
`[X,Y] = meshgrid(x,y)`	Creates grid
`logspace(1.4, 6.3, 1004)`	Make a vector with 1004 elements where the log of the spacing is evenly increasing between 1.4 and 6.3
`7:15`	Row vector of $7, 8, \ldots, 14, 15$

Saving and loading files

`save myfile.mat`	Saves workspace variables in `myfile.mat`
`load myfile.mat`	Loads variables in `myfile.mat` into the current workspace

Existance of names

`exist some_name`	Returns numeric code, depending on whether this name already exists and if so, also depending on it's category (built-in, user-defined, etc.)

Operations on Matrices and Vectors

`3 * x`	Multiply every element of x by 3
`x + 2`	Add 2 to every element of x
`x + y`	Element-wise addition of two vectors x and y
`A * y`	Product of a matrix and vector
`A * B`	Product of two matrices
`A .* B`	Element-wise product of two matrices
`A ^ 3`	Square matrix A to the third power
`A .^ 3`	Every element of A to the third power
`cos(A)`	Compute the cosine of every element of A
`abs(A)`	Compute the absolute values of every element of A
`A'`	Transpose of A
`inv(A)`	Compute the inverse of A
`det(A)`	Compute the determinant of A
`eig(A)`	Compute the eigenvalues of A
`size(A)`	Get the size of A

Matrix and array commands

`repelem`	Repeat copies of array elements
`repmat`	Repeat copies of array
`repmat`	Repeat copies of array
`length`	Length of largest array dimension
`sort`	sort array elements
`flip`	flip order of elements
`reshape`	Reshape array
`end`	Denotes last element
`size(A)`	Get the size of A

Entries of Matrices and Vectors

`x(2:12)`	The 2^{nd} to the 12^{th} elements of x
`x(2:end)`	The 2^{nd} to the last elements of x
`x(1:3:end)`	Every third element of x from the first to last
`A(5,:)`	Get the 5^{th} row of A
`A(:,5)`	Get the 5^{th} column of A
`A(5,1:3)`	Get the first to third elements in the 5^{th} row
`x(10:2:100)`	Contains every 2^{nd} element of x

Plotting

`plot(x,y)`	Plot y versus x (must be the same length)
`loglog(x,y)`	Plot y versus x on a log-log scale (both axes have a logarithmic scale)
`semilogx(x, y)`	Plot y versus x with x on a log scale
`semilogy(x, y)`	Plot y versus x with y on a log scale
`axis equal`	Force the x and y axes to be scaled equally
`title('A Title')`	Add a title to the plot
`xlabel('x label')`	Add a label to the x axis
`ylabel('y label')`	Add a label to the y axis
`legend('foo', 'bar')`	Label 2 curves for the plot
`grid`	Add a grid to the plot
`hold on`	Multiple plots on single figure
`figure`	Start a new plot

Constants

`pi`	$\pi = 3.141592653589793$
`NaN`	Not a number (i.e. $0/0$)
`Inf`	Infinity
`realmax`	Largest positive floating-point number $1.7977 \cdot 10^{308}$
`realmin`	Smallest positive floating-point number $2.2251 \cdot 10^{-308}$

Linear algebra

`x = A\b`	Solves `Ax = b`
`x = linsolve(A,B)`	Solves `A*X = B`
`[V,D] = eig(A,B)`	Diagonal matrix `D` of eigenvalues and matrix `V` whose columns are the corresponding right eigenvectors, so that `A*V = V*D`.
`inv`	Matrix inverse
`realmin`	Smallest positive floating-point number $2.2251 \cdot 10^{-308}$

MATLAB CHEAT SHEET

For loops

```
for k = 1:5
    disp(k);
end
```

While loops

```
k = 0;
while k < 7
    k = k + 1;
end
```

Logicals

```
a = 10;  % Assign a the value of 10
a == 5   % Test if a is equal to 5
    false
a == 10  % Test if a is equal to 10
    true
a >= 5   % Test if a is greater than or equal to 5
    true
a < 11   % Test if a is less than 11
    true
a ~= 4   % Test if a is not equal to 4
    true
a > 1 && a ~= 10 % Test if a is greater than 1 AND
    false        % not equal to 10
a > 1 || a ~= 10 % Test if a is greater than 1 OR
    true         % not equal to 10
```

Conditional Statements

```
if a > 10
    disp('Greater than 10');
elseif a == 5
    disp('a is 5');
else
    disp('Neither condition met');
end
```

Functions

```
function output = addNumbers(x, y)
    output = x + y;
end

addNumbers(10, -5)
    5
```

Multiline commands

```
s = 1 - 1/2 + 1/3 - 1/4 + 1/5 ...
      - 1/6 + 1/7 - 1/8 + 1/9;
```

Function Handles

```
f = @(x) sin(x.^2)./(5*x);

f(pi/2)
    0.0795
f([-pi/2, 0, pi/2])
    -0.0795 NaN 0.0795
```

Plotting

```
x = linspace(-3*pi, 3*pi, 1000);
y1 = sin(x);
y2 = cos(x);

plot(x, y1, 'k-');  % Plot sin(x) as a black line
hold on             % Now we can add another curve
plot(x, y2, 'r-');  % Plot cos(x) as a red line

% Set the axis limits
axis([-3*pi, 3*pi, -1.5, 1.5])

% Add axis labels
xlabel('x');
ylabel('y');

% Add a title
title('A plot of cos(x) and sin(x)');

% Add a legend
legend('sin(x)', 'cos(x)');
```