MATLAB Fundamentals - Cheat Sheet - Tools Course ETH Zürich

Basics

Workspace

`ans`	Most recent answer
`clc`	clear command window
`clear` var	clear variables Workspace
`clf`	Clear all plots
`close all`	Close all plots
`ctrl-c`	Kill the current calculation
`doc` fun	open documentation
`disp('text')`	Print text
`format` short—long	Set output display format
`help` fun	open in-line help
`load` filename {vars}	load variables from .mat file
`save` {-append} file {vars}	save var to file
`addpath` path	include path to ..
`iskeyword` arg	Check if arg is keyword
`% This is a comment`	Comments
`...`	connect lines (with break)
`";" (after command)`	suppresses output
`scriptname`	runs scriptname.m
`tic, toc`	start and stop timer
`ver`	List of installed toolboxes

MATLAB Documentation: mathworks.com/help/matlab/

Defining and Changing Variables

`a = 5`	Define variable a to be 5
`A = [1, 2, 3, 4; 5, 6, 7, 8; 9, 10, 11, 12]`	Set A to be a 3×4 matrix ”,” separates columns ”;” separates rows
`[A,B], horzcat(A,B)`	Concatenate arrays horizontally
`[A;B], vertcat(A,B)`	Concatenate arrays vertically
`x(2) = 7`	Change 2nd element of x to 7
`A(2,1) = 0`	Change $A_{2,1}$ to 0
`x(2:12)`	The 2nd to the 12th elem. of x
`x(1:3:end)`	Every 3rd elem. of x (1st to last)
`x(x>6)`	List elements > 6.
`x(x>8)=8`	change elements using condition
`A(4,:)`	Get the 4th row of A
`A(:,3)`	Get the 3rd column of A
`A(6, 1:3)`	Get 1st to 3rd elem in 6th row
`zeros(9, 5)`	Make a 9×5 matrix of zeros
`ones(9, 5)`	Make a 9×5 matrix of ones
`eye(7)`	Make a 7×7 identity matrix
`diag(x)`	Create diagonal matrix
`diag(A)`	Get diagonal elements of matrix
`meshgrid(x)`	2-D and 3-D grids
`7:15`	Row vector of $7, 8, \ldots, 14, 15$
`a:ds:b`	lin. spaced vector with spacing ds
`linspace(1,20,35)`	Lin. spaced vector (35 elements)
`logspace(1, 1e5, 50)`	Log. spaced vector (50 elements)

Arithmetics

`+, -`	Addition, Subtraction (elementwise)
`A*B`	Matrix multiplication
`A.*B`	elementwise multiplication
`A./B`	elementwise division
`B.\A`	Left array division
`/`	Solve $xA = B$ for x
`\`	Solve $Ax = B$ for x
`A^n`	normal/(square) matrix power
`A.^n`	Elementwise power of A
`sum(X)`	Sum of elements (along columns)
`prod(X)`	Product of elements (along columns)

Elementary Functions

`sin(A)`	Sine of argument in radians
`sind(A)`	Sine of argument in degrees
`asin(A)`	Inverse sine in radians
`sinh(A)`	Hyperbolic sine
there are analogous elementwise trigonometric functions for `cos`, `tan` and `cot`	
`abs(A)`	Compute $\|x\|$
`sqrt(x)`	Compute $\sqrt{x}$
`log(x)`	Compute $\ln(x)$
`log10(x)`	Compute $\log_{10}(x)$
`sign(x)`	sign of x
`exp(x)`	exponential of x

Complex Numbers

`abs(z)`	Absolute value and complex magnitude
`angle(z)`	Phase angle
`complex(a,b)`	Create complex numbers
`conj(z)`	Elementwise complex conjugate
`i or j`	Imaginary unit
`imag(z)`	Imaginary part of complex number
`isreal(z)`	Determine whether array is real
`real(z)`	Real part of complex number
`ctranspose(Z)`	Complex conjugate transpose

Constants

`pi`	$\pi = 3.141592653589793$
`NaN`	Not a number (i.e. 0/0)
`Inf`	Infinity
`eps`	Floating-point relative accuracy
`realmax`	Largest positive floating-point number
`realmin`	Smallest positive floating-point number

Numerics and Linear Algebra

Numerical Integration and Differentiation

`integral(f,a,b)`	Numerical integration
`integral2(f,a,b,c,d)`	2D num. integration
`integral3(f,a,b,..,r,s)`	3D num. integration
`trapz(x,y)`	Trapezoidal integration
`cumtrapz(x,y)`	Cumulative trapez integration
`diff(X)`	Differences (along columns)
`gradient(X)`	Numerical gradient

Matrix Functions/ Linear Algebra

`A'`	Transpose of matrix or vector
`inv(A)`	inverse of A (use with care!)
`det(A)`	determinant of A
`eig(A),eigs(A)`	eigenvalues of A (subset)
`cross(A,B)`	Cross product
`dot(A,B)`	Dot product
`kron(A,B)`	Kronecker tensor product
`norm(x)`	Vector and matrix norms
`linsolve(A,B)`	Solve linear system of equations
`rank(A)`	Rank of matrix
`trace(A)`	Sum of diagonal elements
`curl(X,Y,Z,U,V,W)`	Curl and angular velocity
`divergence(X,..,W)`	Compute divergence of vector field
`null(A)`	Null space of matrix
`orth(A)`	Orthonormal basis for matrix range
`mldivide(A,B)`	Solve linear system $Ax = B$ for x
`mrdivide(B,A)`	Solve linear system $xA = B$ for x
`decomposition(A)`	Matrix decomposition
`lsqminnorm(A,B)`	Least-squares solution to linear eq.
`rref(A)`	Reduced row echelon form
`balance(A)`	Diagonal scaling (improve eig. vec.)
`svd(A)`	Singular value decomposition
`gsvd(A,B)`	Generalized svd
`chol(A)`	Cholesky factorization

Matrix manipulation

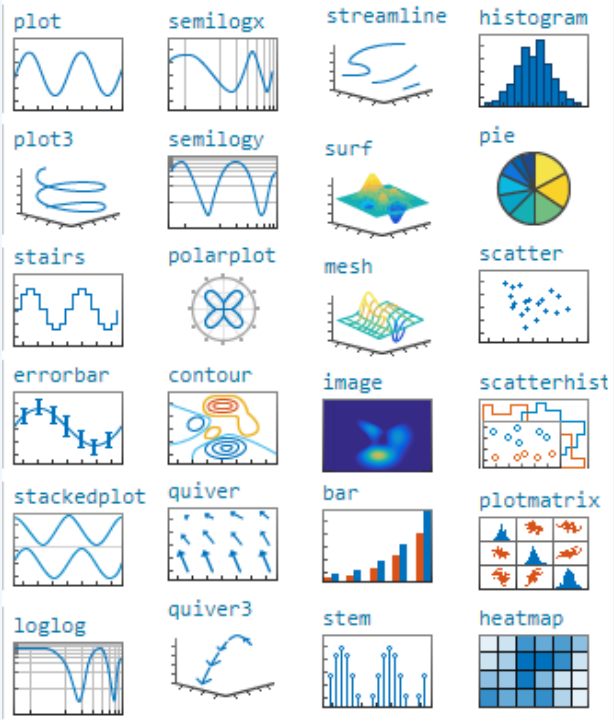
`cat(dim,A,B)`	Concatenate arrays
`ndims(A)`	Number of array dimensions
`flip(A)`	Flip order of elements
`fliplr(A)`	Flip array left to right
`flipud(A)`	Flip array up to down
`squeeze(A)`	Remove dimensions of length 1
`reshape(A,sz)`	Reshape array
`size(A)`	size of A
`sort(A)`	Sort array elements
`sortrows(A)`	Sort rows of matrix or table
`length(A)`	Length of largest array dimension

Graphics

Plotting

`plot(x,y)`	Plot y vs. x
`axis equal`	Scale axes equally
`title('A Title')`	Add title to the plot
`xlabel('x axis')`	Add label to the x axis
`ylabel('y axis')`	Add label to the y axis
`legend('foo', 'bar')`	Label 2 curves for the plot
`grid`	Add a grid to the plot
`hold on / off`	Multiple plots on single figure
`xlim /ylim / zlim`	get or set axes range
`figure`	Start a new plot

Plot types



Plot gallery: mathworks.com/products/matlab/plot-gallery

Programming methods

Functions

```
% defined in m-file
% File must have the same name as the function
function output = addNumbers(x, y)
    output = x + y; %multiple or var nr of args possible
end
```

Anonymous Functions

```
% defined via function handles
f = @(x) cos(x.^2)./(3*x);
```

Relational and logical operations

==	Check equality	~=	Check inequality
>	greater than	>=	greater or equal to
<	less than	<=	less or equal to
&, &&	logical AND	~	logical NOT
|, ||	logical OR	xor	logical exclusive-OR

if, elseif Conditions

```
if n<10
    disp('n smaller 10')
elseif n<20
    disp('n between 10 and 20')
else
    disp('n larger than 20')
end % control structures terminate with end
```

Switch Case

```
n = input('Enter a number: ');
switch n
    case -1
        disp('negative one')
    case 0
        disp('zero')
    case {1,2,3} %check three cases together
        disp('positive one')
    otherwise
        disp('other value')
end % control structures terminate with end
```

For-Loop

```
%loop a specific number of times, and keep track of each ...
    iteration with an incrementing index variable
%parfor might be used to parallelize the execution
for i = 1:3
        disp('cool');    % comment with some LaTeXin it:  πx^2
end % control structures terminate with end
```

While-Loop

```
%loops as long as a condition remains true
n = 1;
nFactorial = 1;
while nFactorial < 1e100
    n = n + 1;
    nFactorial = nFactorial * n;
end % control structures terminate with end
```

Further programming commands

`break`	exit the current loop (combine with `if`)
`continue`	go to next iteration (combine with `if`)
`try, catch`	Execute statements and catch errors

Special Topics

Polynomials

`poly(x)`	Polynomial with roots x
`poly(A)`	Characteristic polynomial of matrix
`polyeig(x)`	Polynomial eigenvalue problem
`polyfit(x,y,d)`	Polynomial curve fitting
`residue(b,a)`	Partial fraction expansion/decomposition
`roots(x)`	Polynomial roots
`polyval(p,x)`	Evaluate poly p at points x
`conv(u,v)`	Convolution and polynomial multiplication
`deconv(u,v)`	Deconvolution and polynomial division
`polyint(p,k)`	Polynomial integration
`polyder(p)`	Polynomial differentiation

Interpolation and fitting

`interp1(x,v,xq)`	1-D data interpolation (table lookup)
`interp2(X,Y,V,Xq,Yq)`	2D interpolation for meshgrid data
`interp3(X,..V,..Zq)`	3D interpolation for meshgrid data
`pchip(x,v,xq)`	Piecew. cubic Hermite poly interpol
`spline(x,v,xq)`	Cubic spline data interpolation
`ppval(pp,xq)`	Evaluate piecewise polynomial
`mkpp(breaks,coeffs)`	Make piecewise polynomial
`unmkpp(pp)`	Extract piecewise polynomial details

Differential equations

`ode45(ode,tspan,y0)`	Solve system of nonstiff ODE
`ode15s(ode,tspan,y0)`	Solve system of stiff ODE
`pdepe(m,pde,ic,bc,xm,ts)`	Solve 1D PDEs
`pdeval(m,xmesh,usol,xq)`	Interpolate num. PDE solution

Optimization

`fminbnd(fun,x1,x2)`	Find minimum of fun(x) in $[x_1, x_2]$
`fminsearch(fun,x0)`	Find minimum of function
`lsqnonneg(C,d)`	Solve non-neg. lin. least-squares prob.
`fzero(fun,x0)`	Root of nonlinear function
`optimget(opt,'par')`	Optimization options values
`optimset('opt',val)`	Define optimization options

Descriptive Statistics

`bounds(A)`	Smallest and largest elements
`max(A)`	Maximum elements of an array
`min(A)`	Minimum elements of an array
`mode(A)`	Most frequent values in array
`mean(A)`	Average or mean value of array
`median(A)`	Median value of array
`std(A)`	Standard deviation
`var(A)`	Variance
`hist(X)`	calculate and plot histogram
`corrcoef(A)`	Correlation coefficients
`cov(A)`	Covariance
`xcorr(x,y)`	Cross-correlation
`xcov(x,y)`	Cross-covariance
`rand`	Uniformly distributed random numbers
`randn`	Normally distributed random numbers
`randi`	Uniformly distributed pseudorandom integers

further functions: `movmax, movmin, cummax, cummin, movprod, movsum, cumsum, cumprod, movmean, movmedian, movstd, movvar.`

Discrete Math

`factor(n)`	Prime factors
`factorial(n`	Factorial of input
`gcd(n,m)`	Greatest common divisor
`lcm(n,m)`	least common multiple
`mod(a,m)`	Remainder after division (modulo operation)
`ceil(X)`	Round toward positive infinity
`fix(X)`	Round toward zero
`floor(X)`	Round toward negative infinity
`round(X)`	Round to nearest decimal or integer