MATLAB Cheat Sheet: B(E)0B17MTB & A8B17CAS Courses

Useful Commands

`clc`	clear command window
`clear`	clear workspace
`clear x`	clear x from workspace
`close all`	close all figures
`help function`	help for function
`doc function`	documentation for function
`quit, exit`	terminates MATLAB
`edit functionName`	opens MATLAB editor
`format short/long/compact`	sets command window output format

Built in Functions/Constants

`pi`	constant π
`exp()`	exponential function
`eps()`	floating point accuracy
`1e6`	1×10^6
`Inf`	infinity value
`NaN`	not-a-number value
`1i, 1j`	imaginary unit

Vectors/Matrices Creation

`x = [1 2 3]`	row vector 1×3
`x = [1; 2; 3]`	column vector 3×1
`x = [1 2; 3 4]`	matrix 2×2
`a:b`	row vector $[a, a+1, \ldots, b]$
`a:x:b`	row vector $[a, a+x, \ldots, b]$
`linspace(a, b, n)`	n evenly spaced points between a and b
`logspace(a, b, n)`	n logarithmically spaced points between 10^a and 10^b
`ones(n, m)`	matrix of values 1, $[n \times m]$
`zeros(n, m)`	matrix of values 0, $[n \times m]$
`eye(n)`	identity matrix, $[n \times n]$
`NaN(n, m)`	matrix of values `NaN`, $[n \times m]$
`Inf(n, m)`	matrix of values `true`, $[n \times m]$
`true(n, m)`	matrix of logical ones, $[n \times m]$
`false(n, m)`	matrix of logical zeros, $[n \times m]$
`rand(n, m)`	matrix of random uniformly distributed values, $[n \times m]$ (see also `randi` and `randn`
`diag()`	diagonal elements of matrix, vector -> square matrix

Vectors/Matrices Operations

`a + b`	element-wise addition
`a - b`	element-wise subtraction
`a * b`	matrix multiplication
`a .* b`	element-wise multiplication
`a / b`	matrix division, mrdivide
`a \ b`	mldivide
`a ./ b`	element-wise division
`a ^ b`	matrix power
`a .^ b`	element-wise power
`a.'`	transpose
`a'`	transpose + complex conjugate (Hermitian transpose)
`[A B]`	concatenates 2 matrices horizontally
`[A; B]`	concatenates 2 matrices vertically
`cat(dim, A, B)`	concatenates 2 matrices
`length(A)`	size of longest dimension of A
`size(A)`	size of A
`ndims(A)`	number of dimensions of A
`numel(A)`	number of elements of A
`tril(A)`	lower triangular part of matrix
`triu(A)`	upper triangular part of matrix
`repmat(A, x, y)`	repeats copy of matrix
`reshape(A, x, y)`	changes dimensions of matrix
`squeeze(A)`	removes dimensions of size 1
`flip(A, dim)`	flips array (see also `fliplr` and `flipud`)
`circshift(A, K, dim)`	circularly shifts array
`det(A)`	matrix determinant
`inv(A)`	matrix inversion
`roots(vec)`	polynomial roots
`round(A)`	round to nearest decimal integer (see also `ceil`, `floor`, and `fix`)
`mod(A, n)`	modulo operation
`rem(A, n)`	reminder after division

Logical Operators

`&`	logical AND
`&&`	short-circuit logical AND
`|`	logical OR
`||`	short-circuit logical OR
`~`	not
`all`	function all
`any`	function any
`is*`	several function with logical output

Relation Operators

`>`	greater then
`>=`	greater then or equal to
`<`	less then
`<=`	less then or equal to
`==`	equal to
`~=`	not equal to

Vector Indexing

`a(1)`	the first element in vector
`a([1, 2, 5])`	the first, second and fifth element in vector
`a(1:3)`	the first three elements of an vector
`a([1:3 5:6])`	selected elements of a vector
`a(end)`	the last element of the vector
`a(5:end)`	from the fifth to the last element in vector
`a(5:end-1)`	from the fifth to the penultimate element in vector
`a(1) = 10`	replace the first element in vector by 10
`a([1 3 5]) = [2 5 7]`	replace elements in vector by values 2, 5, and 7
`a([1 3 5]) = []`	discard elements on position 1, 3, and 5

Matrix Indexing

`A(1, 1)`	an element in the first row and first column of matrix
`A([1 5], 2:4)`	elements of matrix in the first and fifth rows and in columns two, three and four
`A(n, :)`	the nth row in array
`A(:, n)`	the nth column in array
`A(n, :) = []`	discard nth row in array
`A(:, n) = []`	discard nth column in array
`A(1)`	the first element in array (linear indexing)
`A(1:3)`	the first three elements of an array (linear indexing)

Logical Indexing

`A(A == 3)`	extract elements equal to 3
`A(A > 5)`	extract elements greater than 5
`A(A > 5 & A < 30)`	extract elements greater than 5 and at the same time less than 30

Debugging and Time Measurement

`keyboard`	pauses execution
`dbcont`	resume execution (end debug mode)
`dbclear all`	removes all breakpoints
`tic, toc`	start/stop time measurement
`profile on/off/clear/viewer`	profile code execution

Loops and Branching

`if expression ... elseif expression ... else ... end`	program branching using if-else
`switch variable case value1 ... case {value2, value3} ... otherwise ... end`	program branching using switch
`for n = 1:10 ... end`	for cycle
`while expression ... end`	while cycle
`break`	terminates execution of loop
`continue`	pass control to next iteration
`return`	return to invoking function

Functions

`function [out1, out2] = foo(in1, in2)`	function definition
`@sin`	handle function definition
`@(x) x^2 + sin(x)`	anonymous function definition
`nargin`	returns number of inputs
`nargout`	returns number of outputs
`varargin`	input variable allowing multiple inputs
`varargout`	output variable allowing multiple outputs

Set operations

`intersect(A, B)`	set intersection of two arrays
`union(A, B)`	set union of two arrays
`setdiff(A, B)`	set difference of two arrays
`setxor(A, B)`	set exclusive or of two arrays
`unique(A)`	unique values in array
`sort(X)`	sort array elements
`sortrows(X)`	sorts rows of matrix
`issorted(X)`	determines if the array is sorted
`ismember(A, B)`	array elements that are members of set array

Basic Visualization

`figure`	opens empty figure
`axes`	creates Cartesian axes
`plot(x, y)`	2D line plot
`hold on/off`	retain current plot with new line
`grid on/off/minor`	display/hide grid lines
`title(txt)`	adds title to figure
`xlabel(txt)`	adds x-axis label (same for y and z)
`xlim([min, max])`	sets limits of x-axis (same for y and z)
`colormap`	view colormap
`stem(x, y)`	discrete stem plot
`pcolor(A)`	displays array data as colored cells
`surf(X, Y, Z)`	displays surface plot
`semilogx(X)`	semi-logarithmic plot (same for y)
`image(X)`	Show image from array
`doc LineSpec`	plot parameters to customize curves
`set(ref, Name, Value)`	sets graphics object property
`get(ref, Name)`	query graphics object property value
`ref.Name = Value`	dot notation, sets graphics object property
`Value = ref.Name`	dot notation, query graphics object property value
`view(az, el)`	camera line of sight

Character arrays and Strings

`'Hello world!'`	character vector is created using single quotation marks
`A = char(B)`	convert another data type to character array
`strfind(str, pattern)`	find string within other string
`strcmp(str1, str2)`	compare strings
`strjoin(str, delimiter)`	join strings in array
`strtok(str, delimiter)`	split string into parts
`regexp(str, reg)`	match regular expression
`"Hello world!"`	string is created using double quotation marks
`A = string(B)`	convert another data type to string

File Handling

`dir folderName`	lists folder content
`pwd`	identifies current folder
`exist name`	check existence of variable, function, folder, ...
`cd(path)`	changes current folder
`mkdir(path, folderName)`	makes new folder
`rmdir(path)`	remove folder
`writematrix(A, filename)`	writes matrix to file (see also `readmatrix`)
`imwrite(A, filename)`	writes image to graphics file (see also `imread`)
`fid = fopen(filename)`	open file
`fclose(fid)`	close opened file
`fgetl(fid)`	read line from file
`fread(fid)`	read from binary file
`fwrite(fid, A)`	write to binary file

Other Data Types

`A = cell(m, n)`	create cell array
`A = {x, y, z}`	create cell array
`A(1, 1)`	smooth parentheses indexing, creates subset cell array
`A{1, 1}`	curly brackets indexing, access to data in cell array
`B = struct(field, value)`	create structure array
`B.field = value`	create structure array
`double(A)`	converts array to double precision (see also `single`)
`int8(A)`	converts array to 8-bit signed integer (see also `int16`, `int32`, `uint8`, `int18`, ...)
`datetime()`	array representing points in time (see also `years`, `days`, `hours`, ...)
`sparse(A)`	creates sparse matrix

Other useful functions

`find(X)`	find indices of nonzero elements
`fprintf(fileID, format, A1)`	write data to text file
`sprintf(format, A1)`	format data into character vector
`eval(txt)`	executes MATLAB expression in text
`feval(fun, x1)`	evaluate function
`str2num(X)`	convert character array to numeric array
`str2double(X)`	convert characters to double precision numbers
`num2str(X)`	convert numeric array to character array
`realmax`	largest positive floating-point number (see also `realmin`, `intmax`, and `realmin`)
`interp1`	1-D interpolation (see also `interp2`, `interp3`, and `interpn`)
`fminbnd`	find minimum of single-variable function
`fft`	Fast Fourier transform (see also `fft2`, `fftn`, `ifft`, `fftshift`, ...)
`timer`	creates object of timer to schedule execution of MATLAB commands (see documentation)

Shortcuts Command Window

ENTER	line execution
ESC	delete whole line
DEL	delete one character (right to the cursor)
BACKSPACE	delete one character (left to the cursor)
HOME	moves cursor to the beginning of line
END	moves cursor to the end of line
CTRL + ↑	moves cursor to the beginning of next word
CTRL + ↓	moves cursor to the beginning of previous word
SHIFT + ENTER	sends cursor to the next line (without execution)
CTRL + K	deletes all to the right of cursor
CTRL + C	forces interruption of MATLAB (*e.g.*long/erroneous calculation)
CTRL + TAB	switching between windows of MATLAB environment
↓ and ↑	command history listing (searching in available CTRL + F)
F1	context help related to the word where the cursor is placed
TAB	function or variable name hint

Shortcuts MATLAB Editor

CTRL + PG.UP/PG.DOWN	moves between tabs
CTRL + R	comment lines
CTRL + T	uncomment lines
CTRL + D	open highlighted m-file (function/script)
F5	execute current script/function
CTRL + S	save current file (done automatically after pressing F5
CTRL + HOME	jump to the beginning of file
CTRL + END	jump to the end of file
CTRL + →/←	jump word-by-word or expression-by-expression to the right/left
CTRL + W	close current file
CTRL + O	activates open file dialog box
CTRL + F	find/replace dialog box
CTRL + G	"go to", jumps to the indicated line number
CTRL + I	indent of block of lines corresponding to key words
F1	open context help related to the function at position of the cursor
F9	run highlighted code

MATLAB File Suffix (not all)

.m	script/function/class
.mlx	MATLAB live script
.p	protected MATLAB code
.mat	binary data file
.fig	MATLAB figure
.mdl, .slx	Simulink model
.mdlp, .slxp	Simulink protected model
.mexa64, .mexmaci64, .mexw32, .mexw64	mex libraries
.mlappinstall	APP file – installer
.mlpkginstall	support package – installer
.mltbx	toolbox file – installer
.mn	MuPAD notebook
.mu	MuPAD code

Handle Graphics

`fig = figure`	get figure reference
`set(ref, Name, Value)`	sets graphics object property
`get(ref, Name)`	query graphics object property value
`ref.Name = Value`	dot notation, sets graphics object property
`Value = ref.Name`	dot notation, query graphics object property value
`groot, get(0)`	graphics root object (screen)
`axes`	creates Cartesian axes
`axis(lims)`	scales axes
`uimenu`	creates menu or menu item
`uicontextmenu`	creates context menu
`uitoolbar`	creates toolbar with icons
`uipanel`	creates panel; container for other objects
`uitabgroup`	creates tabgroup; container for other tabs
`uitab`	crates a tab; container for other objects
`uitable`	creates 2D table
`uibuttongroup`	creates superior object for group of buttons (*i.e.* radiobuttons)
`uicontrol`	creates fundamental element of GUI, behavior depends on style of the object
`uicontrol('Style', style)`	changes style of uicontrol
`uicontrol('Style', 'Text')`	creates text field
`uicontrol('Style', 'Edit')`	creates editable text field
`uicontrol('Style', 'Pushbutton')`	creates one-state button
`uicontrol('Style', 'Togglebutton')`	creates two-state button
`uicontrol('Style', 'Radiobutton')`	creates radio button
`uicontrol('Style', 'Checkbox')`	creates checkbox
`uicontrol('Style', 'Listbox')`	creates list of items
`uicontrol('Style', 'Popupmenu')`	creates popup menu with multiple choices
`uicontrol('Style', 'Slider')`	creates slider
`set(ref, 'Callback', @foo)`	set callback as function handle
`set(ref, 'Callback', @(src, event)foo(in1))`	set callback as anonymous function
`set(ref, 'Callback', {@foo, in1})`	set callback as function handle in cell array
`gcf`	returns reference of current `figure` (last active)
`gca`	returns reference of current `axes` (last active)
`gco`	returns reference of last "mouse-clicked" object
`findobj(property, value)`	finds object(s) with required property
`findall`	finds all graphics objects
`allchild(ref)`	finds all children objects of selected object
`copyobj(ref)`	copies an object
`delete(ref)`	deletes object
`setappdata(ref, field, data)`	defines stored data inside the application
`getappdata(ref, field)`	retrieves previously stored data
`msgbox(mess)`	creates message dialog window

App Building (new graphics)

`uifigure`	creates graphics window
`uiaxes`	creates axes in `uifigure`
`uipanel`	creates panel; container for other objects
`uitab`	creates tab; container for other objects
`uigridlayout`	layout manager; positioning of graphics components
`uibuttongroup`	creates superior object for group of buttons (*i.e.* radiobuttons)
`uimenu`	creates menu or menu item
`uitable`	creates 2D table
`uilabel`	creates text field
`uieditfield`	creates editable text field
`uitextarea`	creates multiple line editable text
`uibutton`	creates one-state or two-state button
`uiradiobutton`	creates radio button component
`uicheckbox`	creates checkbox component
`uilistbox`	creates list of items
`uidropdown`	creates drop-down component (popup menu)
`uislider`	creates slider component
`uispinner`	creates spinner component
`uiswitch(style)`	creates switch object (possible styles: 'slider', 'rocker', and 'toggle')
`uiknob(style)`	creates knob component (possible styles: 'continuous' and 'discrete')
`uigauge(style)`	creates gauge component (possible styles: 'circular', 'linear', 'ninetydegree', and 'semicircular')
`uilamp`	creates lamp component
`uitree`	creates tree component
`uidatepicker`	creates calendar component
`uiimage`	creates image component
`uihtml`	creats HTML component
`ref.ButtonPushedFcn = @foo`	set callback as function handle
`ref.ButtonPushedFcn = @(src, event)foo(in1))`	set callback as anonymous function
`ref.ButtonPushedFcn = {@foo, in1}`	set callback as function handle in cell array
`setappdata(ref, field, data)`	defines stored data inside the application
`getappdata(ref, field)`	retrieves previously stored data
`uialert`	predefined alert window
`uiconfirm`	predefined confirmation window