Matlab lecture 2

Table of Contents

Clean up 1
Using the Matlab help 1
Test matrices 3
Identity matrix 5
Scripting 5
Publishing 6
Calling Functions 6
General information about variables and numeric types 8
Strings/Text 9
Examples for Scripts 10
Example: Calculate sine and plot it 10
Example: Sum up numbers 11
Example: How old am I 12

Version 0.1 by Markus Schellenberg, Winter Term 16/17

Clean up

```
close all, clear, clc
```

Using the Matlab help

Getting used to the `help` and `doc` functions is really important for the work with matlab. Run the follwing commands and try to find out something about some commands/functions you do not know by now:

```
help help
```

```
HELP Display help text in Command Window.
   HELP, by itself, lists all primary help topics. Each primary topic
   corresponds to a folder name on the MATLAB search path.

   HELP NAME displays the help for the functionality specified by NAME,
   such as a function, operator symbol, method, class, or toolbox.
   NAME can include a partial path.

   Some classes require that you specify the package name. Events,
   properties, and some methods require that you specify the class
   name. Separate the components of the name with periods, using one
   of the following forms:

       HELP CLASSNAME.NAME
       HELP PACKAGENAME.CLASSNAME
       HELP PACKAGENAME.CLASSNAME.NAME

   If NAME is the name of both a folder and a function, HELP displays
```

```
    help for both the folder and the function. The help for a folder
    is usually a list of the program files in that folder.

    If NAME appears in multiple folders on the MATLAB path, HELP display
    information about the first instance of NAME found on the path.

    NOTE:

    In the help, some function names are capitalized to make them
    stand out. In practice, type function names in lowercase. For
    functions that are shown with mixed case (such as javaObject),
    type the mixed case as shown.

    EXAMPLES:

    help close           % help for the CLOSE function
    help database/close  % help for CLOSE in the Database Toolbox
    help database        % list of functions in the Database Toolbox
                         % and help for the DATABASE function
    help containers.Map.isKey   % help for isKey method

    See also DOC, DOCSEARCH, LOOKFOR, MATLABPATH, WHICH.

 Overloaded methods:
    cvtest/help
    cvdata/help
    fdesign.help

 Reference page in Help browser
    doc help
```

gives you informations about the `help` command itself. It will be displayed as a text in the workspace.

```
help doc
```

```
            DOC Reference page in Help browser.

    DOC opens the Help browser, if it is not already running, and
    otherwise brings the Help browser to the top.

    DOC FUNCTIONNAME displays the reference page for FUNCTIONNAME in
    the Help browser. FUNCTIONNAME can be a function or block in an
    installed MathWorks product.

    DOC METHODNAME displays the reference page for the method
    METHODNAME. You may need to run DOC CLASSNAME and use links on the
    CLASSNAME reference page to view the METHODNAME reference page.

    DOC CLASSNAME displays the reference page for the class CLASSNAME.
    You may need to qualify CLASSNAME by including its package: DOC
    PACKAGENAME.CLASSNAME.

    DOC CLASSNAME.METHODNAME displays the reference page for the method
```

```
    METHODNAME in the class CLASSNAME. You may need to qualify
    CLASSNAME by including its package: DOC PACKAGENAME.CLASSNAME.

    DOC PRODUCTTOOLBOXNAME displays the documentation roadmap page for
    PRODUCTTOOLBOXNAME in the Help browser. PRODUCTTOOLBOXNAME is the
    folder name for a product in matlabroot/toolbox. To get
    PRODUCTTOOLBOXNAME for a product, run WHICH FUNCTIONNAME, where
    FUNCTIONNAME is the name of a function in that product; MATLAB
    returns the full path to FUNCTIONNAME, and PRODUCTTOOLBOXNAME is
    the folder following matlabroot/toolbox/.

    DOC FOLDERNAME/FUNCTIONNAME displays the reference page for the
    FUNCTIONNAME that exists in FOLDERNAME. Use this syntax to display t
    reference page for an overloaded function.

    DOC USERCREATEDCLASSNAME displays the help comments from the
    user-created class definition file, UserCreatedClassName.m, in an
    HTML format in the Help browser. UserCreatedClassName.m must have a
    help comment following the classdef UserCreatedClassName statement
    or following the constructor method for UserCreatedClassName. To
    directly view the help for any method, property, or event of
    UserCreatedClassName, use dot notation, as in DOC
    USERCREATEDCLASSNAME.METHODNAME.

    Examples:
       doc abs
       doc signal/abs      % ABS function in the Signal Processing Toolb
       doc handle.findobj  % FINDOBJ method in the HANDLE class
       doc handle          % HANDLE class
       doc containers.Map  % Map class in the containers method
       doc sads            % User-created class, sads
       doc sads.steer      % steer method in the user-created class, sad

Reference page in Help browser
   doc doc
```

gives you informations about the `doc` command. The `doc` command provides the documentation in an easier to read hmtl format inside a MATLAB html browser.

```
doc help
```

provides information about the `help` command inside the documetation.

```
docsearch help OR doc

% doc docsearch
```

`docsearch` helps you to search through MATLAB documentation. In this case it will display information if either one of the keywords `help` or `doc` is found an any doc pages.

Test matrices

Matlab has a lot of build in matrices for testing and convinience. Here are some examples:

```
A = zeros(5,5)
```

```
A =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

creates a 5 by 5 matrix filled up only with zeros.

```
A = ones(5,5)
```

```
A =

     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
```

creates a 5 by 5 matrix filled up only with ones.

```
A = rand(5,1)
```

```
A =

    0.2548
    0.2240
    0.6678
    0.8444
    0.3445
```

creates a 5 by 1 matrix with random numbers from 0-1.

```
A = rand(1,5)
```

```
A =

    0.7805    0.6753    0.0067    0.6022    0.3868
```

creates a 1 by 5 matrix with random numbers from 0-1.

```
A = rand(5,5)
```

```
A =
```

```
    0.9160    0.7702    0.1759    0.6074    0.2691
    0.0012    0.3225    0.7218    0.1917    0.7655
    0.4624    0.7847    0.4735    0.7384    0.1887
    0.4243    0.4714    0.1527    0.2428    0.2875
    0.4609    0.0358    0.3411    0.9174    0.0911
```

creates a 5 by 5 matrix with random numbers from 0-1.

```
A = rand(3,3) * 100
```

```
A =

   57.6209   42.5729   67.9017
   68.3363   64.4443   63.5787
   54.6593   64.7618   94.5174
```

creates a 3 by 3 matrix with random numbers from 0-100.

Identity matrix

Generally accepted mathematical notation uses the capital letter `I` to denote identity matrices, matrices of various sizes with ones on the main diagonal and zeros elsewhere. The function `eye(m,n)` returns an m-by-n rectangular identity matrix and `eye(n)` returns an n-by-n square identity matrix.

```
I = eye(5)
I = eye(2,3)
```

```
I =

     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
     0     0     0     1     0
     0     0     0     0     1


I =

     1     0     0
     0     1     0
```

For all kinds of test matrices just run: `doc gallery`.

Scripting

The console is brilliant to try out commands. You will see immediately if the sytax is right/orientation of the result or other things, you want to achieve. But if you have more than a couple of commands it is often easier to write the commands down in a script.

--> Right click in current folder --> new file --> script --> name it, open it

OR:

click on New --> Script, etc

Your first step should be to give your script a header. This could be the double percentage symbol and a title, the function of this script, your name, a date. You do not need to do this, but it will make it much easier for you (or others) to understand your script at some later time. The header could be very short in the beginning.

Then you could possibly start withpossible:

```
clear       % delete all variables and constants in memory
close all   % close all figures
clc         % delete the screen
```

Parts/Sections to the code can be let run by highlighting it and press F9

Publishing

A very good way to bring a certain structure to your program is to use the publish function. This script is also created by the use of the publish function. To get more information you might browse to: https://www0.maths.ox.ac.uk/system/files/coursematerial/2015/2881/9/publish_guide.pdf On this webpage you will find 'A guide to Matlab publish' ('publish_guide.pdf'). Please get also familiar with the Publish-Tab at the top of your MATLAB editor.

Calling Functions

MATLAB provides a large number of functions that perform computational tasks. Functions are equivalent to subroutines or methods in other programming languages.

```
A = [1 3 5]     % create matrix A
max(A)          % calculate the maximum of A

B = [10 6 4]    % create matrix A
max(A,B)        % calculate the maximum of A and B


maxA = max(A);
```

```
A =

     1     3     5


ans =

     5
```

```
B =

    10     6     4


ans =

    10     6     5
```

gives you the output from the function `max|and stores it into variable |maxA`.

When there are multiple output arguments, enclose them in square brackets:

```
[maxA,location] = max(A);
```

To display them again on the screen just type in:

```
maxA
location
```

```
maxA =

     5


location =

     3
```

Try:

```
C = [1 6 3; 4 5 6; 11 2 1] % defines the matrix C
```

```
C =

     1     6     3
     4     5     6
    11     2     1
```

Calculate the maximum of each column:

```
max(C)
```

```
ans =

    11     6     6
```

You can have the same calculation by using:

```
max(C, [], 1)
```

```
ans =

    11     6     6
```

Calculate the maximum of each row:

```
max(C, [], 2)
```

```
ans =

     6
     6
    11
```

Calculate the maximum of the whole matrix

```
max(max(C))
```

```
ans =

    11
```

Replaces all numbers that are smaller or equal than 5 with 5

```
max(C, 5)
```

```
ans =

     5     6     5
     5     5     6
    11     5     5
```

General information about variables and numeric types

The `who` and `whos` command are powerful functions to give you information about any variable or function inside your current workspace.

`who` displays a list of current variables in the workspace

```
who
```

```
Your variables are:

A         B         C         ans       location  maxA
```

`whos` displays a more detailed list of current variables in the workspace

```
whos
```

```
  Name          Size              Bytes  Class     Attributes

  A             1x3                  24  double
  B             1x3                  24  double
  C             3x3                  72  double
  ans           3x3                  72  double
  location      1x1                   8  double
  maxA          1x1                   8  double
```

`whos(VARIABLE NAME)` displays information about one specific variable:

```
whos A
```

```
  Name      Size              Bytes  Class     Attributes

  A         1x3                  24  double
```

Some numeric types has been shown on the Power Point slides during the lecture. For a complete list please visit: https://www.mathworks.com/help/matlab/... numeric-types.html?requestedDomain=www.mathworks.com&nocookie=true

Strings/Text

In MATLAB Text or Strings are displayed using the single quotes `''`. Try the follwing:

```
disp('Hello world!')    % Displays the characters
text = 'Hello world'    % Stores the text inside the variable |text|
```

```
Hello world!

text =

Hello world
```

If you want to restore the information in a variable name just type in:

```
text
```

```
text =

Hello world
```

or

```
disp(text);
```

```
        Hello world
```

You will see, that the `disp` command gives you only the content and not the variable name (text).

You can join Strings and other numeric types in vectors or matrices. Example:

```
name = 'Don';
age = 12;
X = [name,' will be ',num2str(age),' this year.'];
disp(X)
```

```
        Don will be 12 this year.
```

Examples for Scripts

In the follwing lines you will find three different examples for single scripts

Example: Calculate sine and plot it

Script to calculate the sine of each element of a given set of numbers and plot the graph

M. Schellenberg, 20.10.16

```
% clear all variables plots and screen
clear all
close all
clc
```

- **first try:** sin(0), sin(90), sin(360)
- **Note:** all the values are given in radiant. In order to get degrees use `sin(value * pi/180)` instead or use the command `sind(d)`

```
% Defintion of variables and constants
```

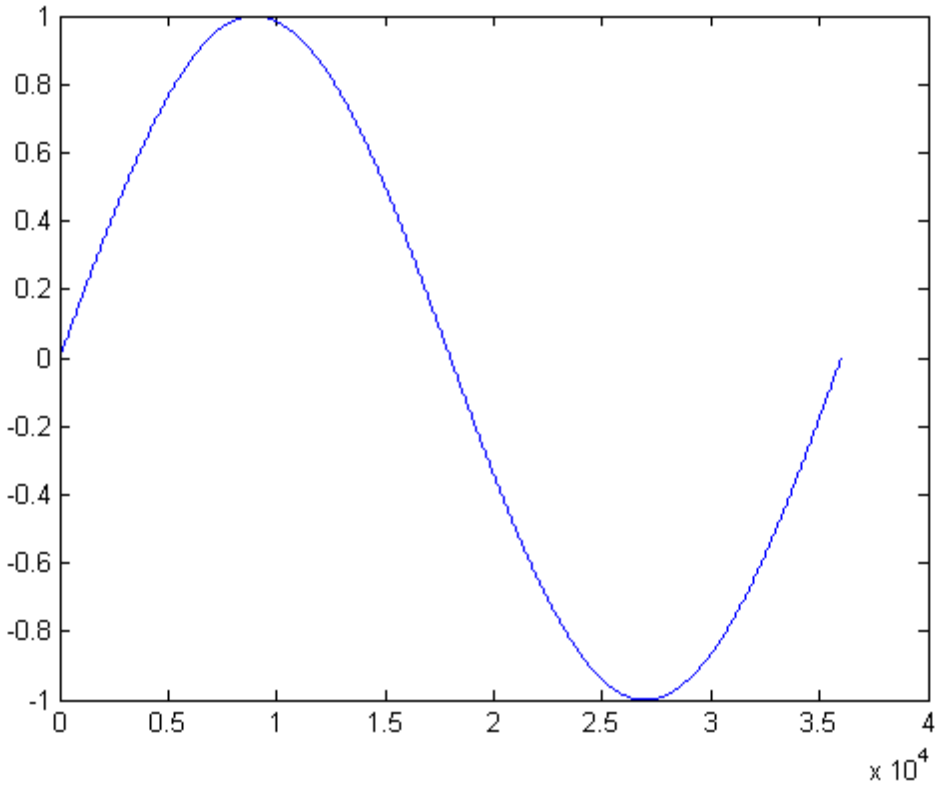
- **First try:** input = [0:360];
- **Then try:** input = [0:0.01:360]
- **Also possible:** plot(sind(input)) or plot(sind([0:0.01:360]))

```
input = [0:0.01:360]; % The semicolon suppresses the output


% Calculation
result = sind(input); % sinus (in degrees) for every value

% Output as a plot
```

```
plot(result)
```



Example: Sum up numbers

This script shows how to effectively use matrix-operation to sum up numbers

M. Schellenberg, 24.10.16

```
% Clear all
clear all
close all
clc

% Declaration of variables and constants
input_number = 1000000;     % Elements of this number will be summed up
                            % Please change!

input = [1:input_number];   % Vector, will be used by different commands
                            % Do not change!
```

Tip: use the `tic ... toc` command to measure the time between the `tic` and the `toc` event

```
% Calculation by typing the numbers

tic
result_by_hand = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10;
```

```
toc
```

```
        Elapsed time is 0.000053 seconds.
```

Calculation using a loop:

```
 result_by_loop = 0;
 tic
 for i = 1:input_number
     result_by_loop = result_by_loop + i;
 end
 result_by_loop;
 toc
```

```
        Elapsed time is 0.600872 seconds.
```

Calculation by position inside vector

```
result_by_position = 0;
tic
for i = 1:input_number
    result_by_position = result_by_position + input(i);
end
toc
```

```
        Elapsed time is 0.692399 seconds.
```

Calculation by matrix operation: sum command

```
tic
result_by_matrixoperation = sum(input);
toc
```

```
        Elapsed time is 0.000699 seconds.
```

You can see, that for higher numbers the operation that uses the MATLAB matrix operations are much faster.

Example: How old am I

Calculates the age in years, months, and days

M. Schellenberg, 2016

```
% clean up
clear all
close all
clc

% Variables and Constants
name = 'YourName';
```

You can determine a vector that represents a given date by using the `datevec` command:

```
birthday = datevec('24-Dez-1990','dd-mmm-yyyy');  % fill in date of birth
```

Calculations

You can determine the actual date by using the `datevec` command:

```
today = datevec(date);      % todays date as a vector
age=today-birthday;         % calculate the age
```

To avovoid negative ages you can use the if command to check for that issue. Note: loops like `if`, `while`, `for` will be explained in one of the following lectures.

```
my_age=age(1);  % Copys the age in years into variable |my_age|
```

Your age changed at your birthday of each year. If your Birthday is still to come `age(2)` will be negative. That means that you have to subtract also a full year from your age calculated before.

```
if (age(2)<0)
    my_age=my_age-1;
elseif((age(2)==0)&& (age(3)<0));
    my_age=my_age-1;
end

% Output on the screen with the |disp| command:
output = [name, ' is ', num2str(age(1)) ,' years, ', num2str(age(2)),...
        ' months and ', num2str(age(3)), ' days old.'];
disp(output)

% Output on the screen with the |sprintf| command:
output = sprintf('%s is %d years, %d months and %d days old.',...
        name, age(1), age(2), age(3));
disp(output)

% Output on the screen with the |fprintf| command:
fprintf('%s is %d years, %d months and %d days old.\n',...
        name, age(1), age(2), age(3));
```

```
        YourName is 26 years, -1 months and -2 days old.
        YourName is 26 years, -1 months and -2 days old.
        YourName is 26 years, -1 months and -2 days old.
```

There might be negative numbers for the months or the days. Try to find out at home why this is the case and how to avoid it.

Published with MATLAB® R2013a