Laboratory 1

MATLAB – a universal environment for scientific and technical calculations

MATLAB (MATrix LABoratory) is an interactive environment for performing scientific and engineering calculations and data visualization:

- a product of *The Math Works, Inc* (http://www.mathworks.com)
- the scope of applications covers various fields of science and technology, including biology, medicine, economics, metrology and many others
- its advantages include the ability to quickly obtain the results of complex calculations and present them in the form of two- or three-dimensional plots, as well as multi-color maps
- it is primarily a high-level programming language, and its environment is typical command language interpreter

Variables in MATLAB :

- real and complex arrays of dimensions [w x k] are the basic data type (where scalars are arrays [1 x 1])
- text (string) variables are the second type of variables
- constants do not occur in the concept of programming languages, they can be written in the form of variables

Variables are not declared in MATLAB; using a variable automatically assigns it to an appropriate area of the workspace. Variables are stored in the MATLAB workspace and are accessible only by name (there are no pointers, as is typical for, for example, C). The variable name can consist of uppercase and/or lowercase characters.

MATLAB language environment:

- openness and ease of expansion of the package - convenient access to commands, functions and libraries
- work in the interactive mode (calculations - plots - report - prints)
- possibility of choosing the hardware platform - programs and data can be transferred to other computers
- open architecture of the package, which consists of:
  - ***M-files***, allowing you to define your own commands and calculation algorithms,
  - ***MEX files*** (compiled subroutines in C or Fortran),
  - ***MAT files*** and ASCII files used to exchange data and calculation results between MATLAB and other programs,
  - ***Graphics*** are used to visualize data and calculation results (animation and sound effects),
  - ***GUI*** graphical interface enables interactive work using editing windows, buttons, sliders and menus,
  - ***DDE*** services perform static or dynamic exchange of text and graphic data between programs in the Windows environment,
- Toolboxes are over 20 specialized software packages for various uses; these are libraries of M-files that extend the scope of MATLAB applications with the latest and most effective achievements in various fields of science and technology
- SIMULINK is an interactive package for modeling and simulating dynamic systems. It allows you to create multi-level block diagrams. Objects are placed in windows as icons - they can be combined into block diagrams for simulation. You can create your own object libraries
- Notebook integrates the services of the MATLAB package and the Word text editor
- additional tools produced by independent companies

Basic documentation:

- MATLAB User's Guide - a manual with examples of practical applications and an overview of the principles of using the package
- MATLAB Reference Guide - contains an alphabetical list of almost all commands and functions, with descriptions and examples of their use

Text Help System

- is invoked with the mouse from the menu or as a command:

  >> *help*

  or

  >> *help command_name*

  or

  >> *help name_of_M-file*
- familiarization with basic operators, instructions and special characters used in MATLAB (> > *intro*)
- demo - a set of programs showing the use of MATLAB for solving selected numerical problems (> > *demo*)

Examples of MATLAB applications

- measurement
  - food quality testing - a database with analysis results of typical food products was created in the laboratory; the test results of a new sample are compared by the expert system with product samples from other manufacturers using image recognition methods
- medicine
  - analysis and visualization of EEG waveforms from 64 channels (electrodes)
- power engineering
  - optimization of the position of control rods in the core of a nuclear power plant
- technical and military applications
  - TOS system used to move a telecommunications satellite from a temporary orbit to a stationary one
  - modeling the aerodynamics of the JAS 39 Gripen fighter aircraft (Saab)
- transport
  - railways - study of the dynamics of a wagon on a magnetic cushion, track vibrations and the control system
  - identification of car parameters based on measurement data from a test drive
  - testing of simultaneous wheel steering control systems in Formula 1 cars

MATLAB Command Window Menu

The MATLAB command window has a header called **MATLAB Command Window**; it contains a main menu with the following options:

- **File** - contains options that allow operations on M-files:
  - *New/M-file* - opens the edit window to create a new M-file. The remaining new options perform:
    - *New/ Figure* – opens a new graphic window,
    - *New/Model* - opens the Untitled window to create a new Simulink model
  - *Open M-file ...* - modifies an existing M-file,
  - *Run M-file ...* - runs M-file,
  - *Print* - prints selected M-files, with the print parameters set via Printer Setup,
  - *Save Workspace As ...* - saves all variables from the workspace to the MAT-file with the given name,
  - *Look for Selected* - searches for information by keywords,
  - *Exit* - ends the MATLAB session
- **Edit** - transfers information to the buffer via functions *Cut, Copy, Paste, Clear*
- **Options** - selects the editor for files and parameters related to, for example, the format of numeric data, color, size and font of characters, etc.
- **Windows** - gives a list of open MATLAB windows
- **Help** - interactive help

MATLAB programming style

- intensive use of functions and arithmetic operators for table and array operations
- a significant reduction in the use of *for* loops, especially in the case of operations on vectors and arrays
- creating your own script and function M-files so that they work correctly for both scalars and arrays
- creating conditions for multiple use of created programs and their fragments
- analysis of M-files provided with MATLAB

Basics of MATLAB

File and directory support

Function name	How the function works
who	lists current variables
whos	lists current variables and gives their dimensions
which *function*	indicates the directory where the function is
size *array*	outputs the array dimensions to the screen
what	outputs a list of m-files to the screen
type *file*	outputs to the screen a listing of the file named file.m
exit	ends the MATLAB session
save	remembers the entire contents of the workspace
load	loads the contents of the matlab.mat file into the workspace
dir,ls	displays the contents of the directory
cd	changes current directory
pwd	shows the name of the current directory
matlabpath	shows a list of directories seen by MATLAB
format	sets the format of the entered data
diary *file_name*	saves the MATLAB session in a file

Special characters

- are used to enter data and comments, write expressions and commands

Character symbol	Special character description	Example
=	value assignment	>> x = 3
[ ]	used when creating vectors, arrays and output argument lists of functions	>> a = [2 3 7]
( )	selecting expressions that are evaluated first and the list of function input arguments	>> y = 2 * (sqrt(225) + 1)
.	decimal point; element of arithmetic operators	>> x = 2.4315
..	parent directory	>> pwd; cd ..; pwd
...	continuation of command on next line	
,	separation of indexes, function arguments, commands	
;	end of array row; stop printing the response	>> d = [5 1 3];
%	the beginning of the comment	% Comment
:	vector generation, array indexing	>> m = 0 : .2 : 5
'	array transposition or conjugate operator	>> A=[1 2 ; 3 4]; B = A'
!	executing an operating system command	>> ! nc

Special variables

Name	Description of a variable or constant
ans	*work variable*
computer	*name of the computer on which MATLAB is running*
eps	*floating point precision (accuracy of calculations)*
flops	*floating point counter*
i,j	*imaginary unit*
Inf	*infinity*
NaN	*undefined value*
nargin	*number of function input arguments*
nargout	*number of function output arguments*
pi	*3.1415926535897...*
realmax	*largest available real number*
realmin	*smallest available real number*

Name	Special variables and time functions
clock	*current date and time*
cputime	*elapsed computer operating time*
date	*current date*
etime	*gives the value of the selected time interval*
tic,toc	*functions for measuring computer time*

Arrays and strings

1) In MATLAB, arrays can be defined in several ways:

- entering a list of array elements from the keyboard:

```
>> A = [1 4 2; 3 7 8; 3 2 5]
>> A = [1 4 2
        3 7 8
        3 2 5]
>> B = [1 2; 3 4] + i * [5 6; 7 8]
```

- loading an array from an external disk file - ***load***
- constructing an array using a function:

Function name	Description of the array constructing functions	Examples
eye	Identity array	>> EYE(N), EYE(N,M)
linspace	a vector with uniformly distributed values	>> LINSPACE(x1, x2, N)
logspace	a vector with logarithmically distributed values	>> LOGSPACE(d1, d2, N)
ones	array with elements equal to 1	>> ONES(M,N)
rand	uniformly distributed random array	>> RAND(M,N)
randn	normally distributed random array	>> RANDN(M,N)
zeros	array with elements equal to 0	>> ZEROS(M,N)
magic	magic square	>> MAGIC(N)

- constructing an array using a colon

2) Colon - vector and array generation operator

- vector generation

  >> **x = (j : k)** - such a record generates a vector [j, j+1, ..., k]
  >> **y = (j : and : k)** - such a record generates a vector [j, j+i , j+2i, ..., k]

- selection of desired rows, columns and array elements

  >> A(: , j) - writing out the j-th column of the array A
  >> A(: , j:k) - writing out columns A(j), A(j+1), ..., A(k)
  >> A(i , :) - writing out the i-th row of the array A
  >> A(:) - writing out all array elements in one column
  >> A(j : k) - writing out, in one line, the elements of array A starting from the element with index *j* up to index *k*
  >> A(j , k) - writing out the element from j-th row and k-th column

3) Selected array functions (a =[1 2 0; 2 5 -1; 4 10 -1])

Function name	Description of the array constructing functions	Examples
‘	array transpose	>> b = a’
*****	array multiplication	>> c = a * b
det	array determinant	>> det(a)
inv	array inverse	>> inv(a), I = inv(a) * a
eig	eigenvalues and eigenvectors	>> [D V] = eig(a)
poly	coefficients of characteristic equation	>> p = round(poly(a))
roots	zeros of polynomial	>> roots(p)

conv	polynomial multiplication	>> conv(p,p)
diag	diagonal array (elements on array diagonal)	>> diag(a)
trio	triangular array with elements above the main diagonal	>> trio(a)
trill	triangular array with elements below the main diagonal	>> trill (a)
svd	singular value decomposition	>> svd (a)

4) Strings

String is a text in the form of a sequence of characters delimited by apostrophes.

```
>> s = 'This is a string of characters'
>> size (s) - size of the vector storing the string
>> s = [ s , ', limited by apostrophes’]
>> n = 4; disp ( ['polynomial ',int2str(n-1),'-th order'])
```

Function name	Description of functions for converting strings and numbers
int2str	integer to string conversion
num2str	number to string conversion
sprintf	number to string conversion with selected format
sscanf	string to number conversion with selected format
str2num	string to number conversion
mat2str	array to string conversion

Arithmetic and trigonometric functions

Name	Function description	Name	Function description
abs	absolute value	**round**	rounding to the lowest integer
ceil	rounding towards + inf.	**sign**	sign of function
exp	exponential function	**sqrt**	square root
fix	rounding towards zero	**floor**	rounding towards – inf.
gcd	greatest common divisor	**sin, sinh, asin**	sine, s. hyperb., arcsine
log	natural logarithm	**cos, cosh, acos**	
log10	base 10 logarithm	**tan, tanh, atan**	
rem	division remainder	**cot, coth, acot**	

Complex numbers

Function name	Command description	Example
	introduction of a complex expression	>> z1 = 3 + 4j
conj	conjugate of z1	>> conj (z1)
abs	modulus of z1	>> abs (z1)
angle	phase angle of z1	>> angle (z1)
real	real part of z1	>> real (z1)
image	imaginary part of z1	>> imag (z1)

Graphics in MATLAB

- the ability to easily present calculation results in graphical form
- the result of graphic functions appears in the MATLAB graphics window
- the graphics are object-oriented and very effective
- simple image animation options are available
- GUI - graphical user interface

Two-dimensional plots

A two-dimensional plot with a linear scale on both axes can be made using the *plot* function. Variants of using this function:

plot(y)
plot(x,y)
plot(x,y , *' line_type '* **)**
plot(x1,y1, *'line_type1'* **,x2,y2,** *'line_type2'***)**

where:

x, y – N-element vectors or an array of size N x M
line_type - color and/or type of the chart line
x2,y2,'line_type2', ... - parameters of subsequent graphs.

Symbol	Color	Symbol	Line type
y	yellow	**.**	point
m	purple	**o**	circle
c	blue	**x**	x sign
r	red	**+**	plus
g	green	*****	asterisk
b	blue	**-**	continuous
w	white	**:**	dot
k	black	**-.**	dash-dot
		- -	dash

Function name	Description of the function for drawing plots
plot	linear scale of both axes
loglog	logarithmic scale of both axes
bar	bar chart
hist	histogram
polar	pie chart
stem	discrete chart
stairs	step chart
grid	overlaying a grid on the chart
legend	chart legend
text	placing a caption in a selected place on the chart
title	text describing the chart
xlabel	x-axis description
ylabel	y-axis description

Operators in MATLAB

- are used to build expressions in MATLAB
- are used to perform table and array operations, to determine transposed and conjugate arrays
- the priority (order of operation) of the operators is as follows:
  - arithmetic operators
  - relation operators
  - logical operators

Symbol	Operation name
+	addition
-	subtraction
*****	multiplication
^	exponentiation
/	right division (B*inv(A))
\	left division (inv(A)*B)
‘	array conjugate
.’	array transpose
kron	Kronecker tensor product

Symbol	Relation operator
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	equal to
~=	unequal to

Symbol	Logical operator
&	AND (conjunction)
|	OR (disjunction)
~	NOT (negation)
xor	EX OR

Examples:

```
>> x=[1 2 3]; y=[4 5 6];
>> x * y
>> y’
>> x * y’
>> x.* y
>> x. * y’
>> x’ * y
>> x1 = x * 2; x2 = x. * 2;
>> x \ y
>> x. \ y
>> x / y
>> x. / y
>> x ^ y
>> x. ^ y
```

MATLAB Instructions

- conditional instruction

```
if command
   expression
elseif command
   expression
else
   commands
end
```

- iterative instructions

  – perform an indefinite number of loop iterations:

```
while command
   expression
end
```

  – perform a precisely defined number of loop cycles:

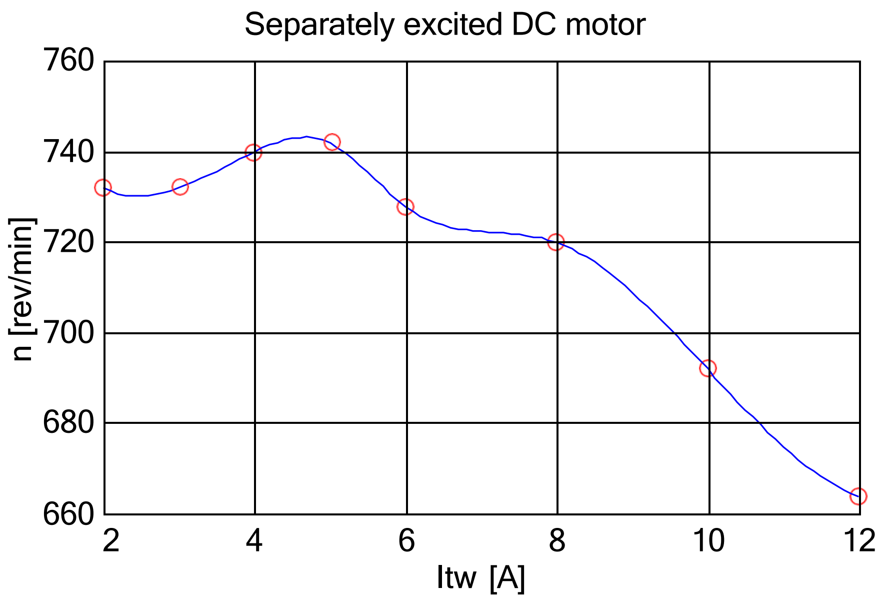
```
for variable=command
   expression
end
```

Name	Description of keywords used to create instructions
break	exit from iteration loop
else	used with if
elseif	used with if
end	terminates a sequence of commands
error	error diagnostic messages
for	repeat a sequence of commands, a specified number of times
if	conditionally execute a sequence of commands
return	return to a called function
while	repeat a sequence of commands, an unspecified number of times

Example of a program written in MATLAB

```
% Plot of the dependence of the motor speed on the armature current
% Interpolation using the spline function was used

        figure('Unit','Centim','Pos',[2,2,13,8.5]);
        x=[2 3 4 5 6 8 10 12];
        x=x';
        y=[732 732 740 742 728 720 692 664];

        xi = 2:.1:12;
        yi = spline(x,y,xi);

        axes('Units','Centim','Position',[2 1.5 10 6],'XLim',[2 12],'YLim',[660 740]);
        plot(x,y,'r o',xi,yi,'b');

        grid;
        ylabel('n [obr/min]');
        xlabel('Itw [A]');
        title('Separately excited DC motor');
```

As a result, we got the following plot:



References:

[1] MATLAB/Simulink documentation: http://www.mathworks.com/help/