Matlab

Introduction

- Matlab is very useful mathematic software for making operations with matrices.
- Matlab works with one type of a variable: **matrix**. A *number* is matrix of dimension 1×1. A *vector* is matrix of one row or column ($N \times 1$ or $1 \times N$).
- You can work directly in Command Window, or you can make scripts or functions in a text editor and use them repeatedly.
- User environment – windows: Command window - for writing commands or running m-files. Workspace/Current Directory. Command History. Editor window.

Miscellaneous

- Change keyboard to English! (Press key ALT+SHIFT)
- To change directory use button □...
- To go through command history use arrows (keys ↑ and ↓)
- Useful command `help` gives you info about any function or command, e.g. `help format`
- Matlab uses decimal point (not comma).
- If something is wrong, Matlab writes *error message* in command window.

Working with matrices

- To assign a value to a variable (matrix) use sign of equation =. The variable is on the left side, the value (or expression) is on the right side. Example: `a = 2;`
- The semicolon in the end of command line causes that the result is not displayed on screen.
- Matlab distinguishes small and capital letters: variable `a` is not `A`
- To make a matrix (vector), use square brackets. Example:

  ```
  A = [1 2 3; 4,5,6; 7 8,9];
  ```

  To separate individual elements, you can use space or comma. To separate rows, you can use semicolon or press key ENTER.
- Some special functions for creating special matrices. Function `ones` produces matrix of all elements 1, function `zeros` makes matrix of all elements 0 and function `eye` makes unit matrix (with 1 on main diagonal, otherwise 0). These functions have one or two paramteres that define number of rows and columns.

- The dimension of matrix is assigned in brackets, always in this order (row, column). Example: `B = eye(3,5);` Using only one parameter produces square matrix (with the same number or rows and columns) `C = ones(3);`

- To make a vector (matrix) of random numbers (with normal distribution), use function `randn`. Example: `b = randn(100,1);`

- You can make a vector by expansion from the first element to the last element with some step (the numbers are separated by colon). Example: `c = 10:0.5:20;` makes vector from 10 to 20 by step 0.5. If you use only two numbers: the first and the last element, the default setting of step is 1. Example: `t = 101:200;`

- Special variable in MATLAB: `NaN` or `nan` means Not-a-Number. (Very useful to fill in missing elements of a vector).

Working with elements of matrix

- Sometimes we want to work only with a part of matrix or with some elements. We use indexes of elements in the following order (row, column) assigned in the brackets behind the matrix `A(i,j)`. Example: We want to select the element on the second row and third column from matrix A and assign it to variable d: `d = A(2,3);`.

- To select the first two rows and first and third column: `e = A([1,2],[1,3]);` or `e = A(1:2,[1,3]);`

- To select the whole column (or row) you can use operator colon (`:`) and simply write: `f = A(:,2);` It means that you select all elements (all rows) in the column 2. Another possibility is `f = A(1:end,2);`

- To shorten vector you can write `g = b(41:end);`

Matrix operations

- For transposition of vector (matrix) use apostrophe (`’`). Example: Transpose vector x:
  ```
  x = [1, 2, 3, 4, 5];
  y = x’;
  ```
  or directly `y = [1, 2, 3, 4, 5]’;` Command `y = (1:5)’` makes the same result.

- Addition and subtraction of matrices is easy: `D = A + C;` however the matrices must have the same dimensions: $A_{m\times n}, C_{m\times n}$. The same holds for vectors, they must be the same length.

- Multiplication of matrices is made by operator `*`. We must hold the rules for multiplication of matrices: Number of columns of matrix A must equal to number of rows of matrix B. For matrices $A_{m\times n}$, $B_{n\times r}$ is possible to compute `E = A * B;` where $E_{m\times r}$.

- Operations with number and matrix or number and vector is not restricted; the dimension can be whatever. Example: `z = y/2` or `z = y*0.5` or `z = y*(1/2)`.

- Exponentiation is made by hat (`^`). However this command `F = A^3;` means `F = A * A * A;` and is valid only for square matrices. Operation `y^2;` which is equivalent to `y * y;` is not possible. The dimension doesn't hold the rule for multiplication.

- Sometimes we need to make operation element-by-element. It means that any operation is made not with the whole matrix but with individual element. For this type of operation add a dot (`.`) before operator. Example: `H = A.^3` means that every element of matrix A is to the 3^{rd} power. `G = A./4` means that every element of matrix A is divided by 4.

Working with variables

- Command `who` shows all variables you have defined. Command `clear all` erases all variables (usually used at the beginning of m-file).

- To find out the size of a matrix, use command `size(B)`. This command returns two items: number of rows and number of columns. For vectors you can use command `length(x)` that returns the length of row vector or column vector (always one number).

- You can switch between different formats of displaying of output. Example: `format bank` display number rounded to two decimal places.

- You can load data from external source (usually text file, "`.txt`"). Use command `load`. Example: `load('us_data.txt');`.

Displaying graphs

- Before drawing a graph create a new figure window by command `figure`. You can also specify the number of the figure e.g. `figure(1)`.

- Command `plot(b)` displays vector `b`; on the x-axis there is index of elements of vector `b`. If you want to specify x-axis, you should use two arguments: `plot(t,b)`. If you want to display more vectors into one figure window, you should use more arguments, but always in the order (x-axis, y-axis): `plot(t,b,t,w)`. Alternatively, `plot(t,[b,w])` produce the same result.

- **Remember!**: the vectors for plotting must be the same length.

- Commands `xlabel` and `ylabel` label x-axis and y-axis. Commands `legend` and `title` puts legend and title to the figure. Text must be in brackets and in apostrophes. Example: `xlabel('time')`, `legend('consumption')`. Command `grid` makes a grid in the figure window; the graph is then more transparent.

Scripts and functions

- You can create so called **m-file** which is ordinary text file with suffix "`.m`" that contains MATLAB code (list of commands). It is possible to run this file repeatedly, modify it and to correct errors (debug). For writing m-files is useful MATLAB editor with syntax highlighting. You can write comments behind `%`.

- To run m-mile: press key F5 or press button $\downarrow\boxed{\equiv}$ or write the name of m-file in Command Window.

- Special type of m-file is function. There are some input parameters and output values of every function. The output values are in square brackets, input parameters are in brackets behind the name of function. Example: `[gdp_trend, gdp_gap] = hp(gdp,1600);` or `[i,j] = size(B)`. You can create your own functions. Use function like command, don't run it as m-files.

- To quit MATLAB write `exit` or `quit` in the Command window or press button $\boxed{\times}$