An Introduction to MATLAB

AIV M1 Bootcamp

September 2015

valentina.peschetola@gmail.com

chiara.fracassi@inria.fr

Attendance

Please arrive on time. Attendance to the class is mandatory for all students. If you already know MATLAB and find these exercises too easy, ask us, and we will happy to find more complicated exercises for you! If you have to skip a class we expect you to catch up before the following class.

Evaluation

We will not grade you: the goal of this class is to give you tools to survive this semester! However you should not underestimate the importance of this class: not being comfortable with MATLAB would seriously compromise your chances of succeeding your M1.

Prerequisite

You do not need any prior programming nor biological knowledge. A part of the exercises relies on real published dataset, but we choose simple enough problems so biological knowledge is never the bottleneck.

Documentation

The first thing we are going to teach you is to use MATLAB documentation. For all the following exercises, we thus provide the name of the MATLAB functions you should use so you can efficiently use this documentation. This will allow you to go at your own speed.

General Remind

1. Create a folder to keep your program organized.
2. Comment your code: anything following % is seen as comment. Comment thoroughly to avoid wasting time later!
3. Always test and verify your code.

About variables names in MATLAB

1. First character must be a letter (after the name can contain any combination of letters, numbers and underscore; punctuation and space are not allowed).
2. MATLAB is case-sensitive: var1 is different from Var1.
3. Give meaningful names to your variables/functions.
4. Avoid built-in names for variables: i, j, e, Inf, -Inf, pi, NaN, realmin, realmax …

Part I - Exercises

A. The very beginning

1. Start MATLAB
2. Calculate $3.15 + \frac{45.6 + 4.987}{32} * 8.4$ and store the results in the variable *x*.
3. Type *doc cos* in the command window.
4. Calculate $\cos\left(\frac{\pi}{2}\right)$, $\cos\left(\frac{3\pi}{2}\right)$, $\cos(\frac{3\pi}{5})$ and store the last result in the variable *y*.
5. Write a MATLAB script to swap the content of variables *x* and *y*.

B. Functions and algorithms

1. Documentation: *functions*.
2. Write a MATLAB function **m2sum** that takes two numbers and returns their sum.
3. Write a MATLAB function **pw** that asks two numbers as input and raises the first number to the second one.
4. Write a MATLAB function that takes an integer *n* and displays *n* time "hello". Documentation: *for*, *disp*.
5. Write a MATLAB function that takes a vector of length *n* (with $n > 50$) and plots the square root of all elements.
6. Write a MATLAB function **sequence0** that takes an integer *n* and returns the n^{th} element of the sequence (u_n) defined by:
   - $u_0 = 0$
   - $u_{n+1} = 2*\cos(u_n) + 1$
7. Write a MATLAB function **sequence1** that takes an integer *n* and returns the n^{th} element of the sequence (u_n) defined by:
   - $u_0 = 1$
   - $u_{n+1} = u_n + n^2 - 4$
8. Write a MATLAB function **sequence2** that does the same thing for the sequence defined by:
   - $u_0 = 8$
   - $u_{n+1} = u_n/4$ if u_n is even, $2*u_n+1$ otherwise. Documentation : *if*, *mod*.
9. Write a MATLAB function **pra0** that takes five numbers and returns their sum if one of them is zero, their product otherwise.

10. Write a MATLAB function **pra1** that takes three numbers and returns the "middle" one (the one that is nor the highest nor the lowest).

11. Write a MATLAB function **pra2** that takes three numbers and returns the product of the two lowest numbers.

12. Write a MATLAB function **pra3** that takes five numbers and returns 0 if at least two of the numbers are equals, 1 otherwise.

C. More advanced algorithms

1. Write a MATLAB function **mgcd** that takes two integers and returns their greater common divider. Documentation: while.

2. Write a MATLAB function **misprime** that takes a positive integer and tells whether it is a prime number. Documentation: sqrt, floor, break.

3. Calculate the sum of the first 500 prime integers.

D. Algorithmic on vectors

1. Write a matlab function mini that takes a vector and returns its lowest element. Documentation: Inf.

2. Write a matlab function mini2 that takes a vector and returns its lowest element and the index of this element.

3. Write a matlab function reverse that takes a vector and reverses it (the first element becomes the last one and so on). Documentation: size, length, zeros.

4. Write a matlab function incsort that takes a vector and sorts it in ascending order.

E. Loading and writing data

1. Load the variable **m** contained in the file load1.mat. Sum the elements of **m** and put the result in a variable s that you will store in the file save1.mat. Documentation: load, sum, save.

2. Load the file data1.xls and sum the third line of numerical values. Documentation: xlsread.

F. Working with matrices

1. Load the matrix A from the file a.mat.

2. Multiply element by element the third and fifth column of A, ie create a colum vector z such that $\forall i$, z(i) = A(i, 3) $*$ A(i, 5). Documentation: arith.

3. Create a column vector that contains the sum of each line of A. Sum all the elements of A that

belong to an even column.

4. Count the number of elements of A that are greater than 2. Documentation: find, numel.

G. Matrix again

1. Write a general function "subtractme", applicable to any two-dimensional matrix, that subtracts from each row its minimum value and saves it in a new matrix.
2. Write a general function "stackme", applicable to any two-dimensional matrix, to generate a stack in which each slice is the result of the element-wise product of itself and one of its diagonal elements.

H. Text printing and parsing

1. Write a matlab function nicetext that takes a matrix and prints the highest element of each column indicating the number of the column:The highest element of column 1 is 15The highest element of column 2 is 7. Documentation: fprintf.

2. The file samples.txt contains a measure for each sample identified by a letter and a digit (for example A1). We consider that samples identified with the same letter (but different digits) belong to the same group. Print the average value of the measure for each group. Documentation: fscanf.

I. Distances

1. The Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. Write a matlab function hamming that takes two vectors and returns their hamming distance. Documentation: assert.
2. The Levenshtein distance between two strings is defined as the minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character. Write a matlab function levenshtein that takes two vectors and returns their Levenshtein distance. Documentation: return.

J. Plotting 2D curves

1. Plot the graph of the function $x \rightarrow 3 * x^2 - 6 * x + 1$ over [0,10]. Documentation: plot.

2. Plot the graphs of the functions sin and $\cos^2$ over [0,5] with different colors. Add a legend to identify each function. Documentation: legend.

K. Randomness and plotting histograms

1. Create a 10000 by 5 matrix A filled with uniformly distributed random numbers between 0 and 10. Create a vector V containing the sums of each line of A. Plot the distribution of V. Documentation: rand, hist, bar.
2. Create a 100 by 1 matrix B filled with normally distributed random numbers with mean 2 and standard deviation 1.5. Plot the distribution of B. Documentation: randn.

L. Plotting points

1. Load the file points.mat. It contains a list l of 2D points. Plot these points using "x" as marker on a 2D graph. Zoom to plot only $x \in [0, 5]$ and $y \in [-1, 1]$. Documentation: xlim, ylim.Name the x-axis "time" and the y-axis "sales". Documentation: xlabel, ylabel.
2. The file points.mat also contains standard error for each point in variable err. Plot the same graph as in q. 1 with error bars. Documentation: errorbar.

Advanced plotting

1. Let's consider the function $f(x, y) = \sin(6 \times x) + y$. Plot f over $[0, 1] \times [0, 1]$ as a 3D graph, i.e. z = f(x,y). Documentation: surf.
2. Plot the same function as a color 2D graph, ie (x,y) is colored according to the value of f(x,y). Add a color scale. Documentation: pcolor, colorbar.
3. Put the last two graphs on the same figure. Documentation: subplot. Export this figure as a pdf file named f1.pdf. Documentation: print.

Basic statistics

1. We randomly picked 20 men and 15 women among the world population, and recorded their size in the file humansizes.mat. Perform a statistical test that will estimate whether men are statistically taller than women. Documentation: kstest, ttest2.
2. We did the same for Northern Gannet, with 30 females and 28 males, and saved the results in the file birdsizes.mat. Perform a statistical test that will estimate whether there is a size difference between males and females.

More practice in parsing, analyzing and plotting data

The file plants.txt contains a list of plants identified by their location on a field (two coordinates between 0 and 100) and their types (A, B or C). More precisely, each line contains in the order x coordinate (decimal), y coordinate (decimal), and type (1, 2 or 3 for A, B or C).

1. Load the data. Documentation: importdata.
2. Make a 2D representation of the field, plotting each plant with different markers and colors. Documentation: scatter,plot.
3. Divide the space in a 50x50 grid and make three color plots, each one indicating the density of one type of plant in each grid location. Documentation: colormap.
4. Find a way to represent both densities of plant A and plant B on the same color plot. Plot the distribution of the distances between two random plants of type A.Same question for plant B and plant C.
5. Compare the 2 previous distributions with what you would get for 10,000 plants randomly positioned on the field (each coordinate coming from a uniform distribution between 0 and 100).
6. Do you think there is an interference between plant A and plant B? Make graphs to support your hypothesis.

First order differential equations

Use MATLAB to get a numerical solution (ie a 2D curve) of the following differential equations, and use your math skills to get a formal solution when you can. We will always solve over [0,1]. Documentation: ode45.

1. $y' = 2$ with initial condition $y(0) = -1$
2. $y' - 3 \times y = 2$ with initial condition $y(0) = 1$
3. $y'(x) + 2y(x) = e^{2x}$ with initial condition $y(0) = 0$
4. $\frac{dy}{dx} - 5y = e^{5x}$ with initial condition $y(0) = 1$
5. $\dot{x} - x = sin(t)$ with initial condition $x(0) = 1$
6. $(2 + t)\frac{dy}{dt}(t) = 2 - y(t)$ with initial condition $y(0) = 1$

Second order differential equations

Use MATLAB to get a numerical solution (ie a 2D curve) of the following differential equations, and use your math skills to get a formal solution when you can. We will always solve over [0,1].

1. $y'' - 2y' + 2y = xe^x$ with initial conditions $y(0) = 1$ and $y'(0) = -1$
2. $\ddot{y} - 4\dot{y} + 4y = 2(t-2)e^t$ with initial conditions $y(0) = 1$ and $\dot{y}(0) = -1$
3. $\frac{d^2x}{dt^2} - 4\frac{dx}{dt} + 13x = 10\cos(2t) + 25\sin(2t)$ with initial conditions $x(0) = 1$ and $\frac{dx}{dt}(0) = -1$

System of differential equations

Use MATLAB to solve the following system over [0,100] with parameters r = 0.3 and m = 0.1.

- a' = r*sin(t)*a + m*a*b - m*a*c
- b' = r*cos(t)*b + m*b*c - m*b*a
- c' = r*sin(t/2)*c + m*c*a - m*c*b