Advanced Algorithms for Finance
Master of Financial Engineering - M2
Academic Year 2014 - 2015
Jung Jonathan

Second Practical : Matlab and Linear Systems

Matlab makes it possible to execute a code stored as a text file or to directly run instructions from the command line. The default folder to store files is *Documents/Matlab/*. If some texte file, say *code.m*, containing a Matlab code is found in this folder, the command *code* runs it. It is possible to change the folder using command *ls*. Matlab is designed to work with matrices. For instance, a matrix A of n rows m columns can be defined using the following command :

$$A = [a_{11}\ \ a_{12}\ \ ...\ \ a_{1m}; a_{21}\ \ a_{22}\ \ ...\ \ a_{2n}; ...; a_{n1}\ \ a_{n2}\ \ ...\ \ a_{nm}],$$

where a_{ij} is the entry in row i and line j. The entries of a given column are separated by whitespaces and consecutive rows by semicolumns. Now consider the following matrix A and column vector b :

$$A = \begin{pmatrix} 1 & 1 & 7 \\ 2 & -1 & 5 \\ -1 & -3 & -9 \end{pmatrix} \text{ and } b = \begin{pmatrix} -1 \\ -5 \\ -5 \end{pmatrix}.$$

1. In a file *essai.m*, create a code that defines matrix A and vector b and execute it. As a standard, Matlab displays the result of every command. In order for the result of a given command not to be displayed, this command has to end with a semicolumn. After your code has been run, variables A and b remain usable directly in the command line.
2. The entry of a matrix A in row i and column j can be extracted using command $A(i, j)$. It is also possible to extract a sub matrix from A using the following command :

$$B = A(i : j, k : l),$$

where i and j are the first and last row of the sub-matrix and k and l are its first and last column. Using command line, extract row vector $(-3, -9)$ from matrix A.

3. Command $A(i, j) = x$ modifies entry of row i and column j of matrix A. Similarly, one can modify a whole sub-matrix using the following command :

$$A(i : j, k : l) = X,$$

where i and j are the first and last row of the sub-matrix and k and l are its first and last column. Using command line, replace row vector $(2, -1)$ in matrix A by row vector $(3, 3)$.

4. In Matlab, functions are text files whose first line reads :

$$\text{function r} = \text{f}(a_1, a_2, ...).$$

Here f is the name of the function, a_1, a_2, ... are its arguments and r is the result returned by the function. The file containing this function must be named as the function itself : here the name of the file should be *f.m*. Write a function *my_sum* whose arguments are two matrices of the same size and that returns their sum.

5. *if* conditions are written in the following way in Matlab :

if(a operator b)
...
else
...
end

Here, the *else* statement is optional and the (comparison) operator can be == (equal), < (less than), <= (less than or equal to), > (greater than), >= (greater than or equal to), ∼= (different). Modify the code written in question 4) so that, if the two matrices do not have the same size, the code returns 0 are display an error message with command $disp('\text{text}')$.

Remark : The number of rows of a matrix A is found using command $size(A, 1)$ and its number of columns using command $size(A, 2)$.

6. *For* loops are written as follows in Matlab :

   for i = first : step : last
   ...
   end

   Write a function fonction *my_transpose* that returns the transpose of a matrix A using *for* loops and assigning the entries of the transpose matrix one by one. You can check the result using the built-in *transpose* command.

7. Write a function *index_max* that returns the index k of the maximum of a column vector b.

8. Write a function *Gauss_line* that takes a matrix A and a column vector b and that performs Gaussian elimination on augmented matrix $A|b$. This function will return the augmented matrix in reduced row echelon form.

9. Use the function written in Question 8) to solve system $Ax = b$. Precise also the rank of the matrix A. Consider the following matrices :

$$i) \quad A_1 = \begin{pmatrix} 1 & 1 & 7 \\ 2 & -1 & 5 \\ -1 & -3 & -9 \end{pmatrix} \text{ and } b_1 = \begin{pmatrix} -1 \\ -5 \\ -5 \end{pmatrix}.$$

$$ii) \quad A_2 = \begin{pmatrix} 1 & 2 & 3 & 2 & -2 \\ 5 & 6 & 7 & -2 & -6 \\ 8 & 9 & 1 & 4 & 0 \end{pmatrix} \text{ and } b_2 = \begin{pmatrix} 14 \\ 38 \\ 29 \end{pmatrix}$$

$$iii) \quad A_3 = \begin{pmatrix} 1 & -1 & 3 & 2 & 4 \\ 5 & -5 & 7 & 10 & 12 \\ 8 & -8 & 1 & 16 & 9 \end{pmatrix} \text{ and } b_3 = \begin{pmatrix} 14 \\ 38 \\ 29 \end{pmatrix}$$

10. Using command *linsolve*, solve system $A_1x = b_1$ and $A_2x = b_2$. Compare the results with those obtained in Question 9).

11. We assume that the linear pricing hypothesis held. Write a function *Complete_StatePrices* that tells wether a market model over one period (given by a matrix S_1 and a column vector S_0) is complete and this case computes a vector of state prices if there exists one. You will use the function written in Question 8) to compute ranks and to solve linear systems.

12. We remark that if N is a matrix of generator of $Ker(A_2)$, for all vector c

$x_2 + N\,c$ is also a solution of $A_2 x = b_2$. Use the function *lsqlin* of *Matlab* to build a non negative solution of $A_2 x = b_2$. Use the same function to build a positive solution of $A_2 x = b_2$.

13. Use the function *lsqlin* to directly build a non negative solution of $A_2 x = b_2$. Use the same function to build a positive solution of $A_2 x = b_2$.

14. Assume you need to improve the function written in Question 9) to provide an arbitrage portfolio, in case there is one. What piece of code is required, that you do not already have at hand ?