Introduction to MATLAB

Econ 8305 Fall 2015

Hang Zhou

The George Washington University

Overview

Warning

You will never handle a programming language by just reading notes or books. The best way to learn is to get more practice. So, write your own code!

Why do Macroeconomists Need to Learn Programming Languages?

- Estimate and simulate DSGE models (e.g. Dynare)
- Some newly developed econometrics methods which are not available in the build in statistic software (STATA, Eviews, OxMetrics...)
- You should at least be familiar with one of the following languages: Matlab, R, Guass, SAS, Python, etc.

Study Recourses

- MATLAB Tutorials and Learning Resources on Mathwork website: link
- MIT online course: link
- On-line help facility: use “help” command

Assignment and Operators

- Assignment (=) a=3
- Addiction (+) a+b
- Subtraction (-) a-b
- Multiplication (*) a*b
- Division (/) a/b
- Power ($\wedge$) a$\wedge$b

The order in which calculations are performed: ( ) , $\wedge$, * or /, + or -
You can use ";' ' at the end of each command to suppress the output on interface.

Define a Matrix

Entering matrix (vector) variables:

$A = [1, 2; 3, 4]$ 4 by 4 matrix

$B = [1, 2, 3, 4]$ 1 by 4 matrix

$C = [1; 2; 3; 4]$ 4 by 1 matrix

“,” or “ ” is used to separate columns, “;” is used to separate rows.

Some Useful functions

zeros(m,n): creates an mxn matrix whose elements are equal to zero

ones(m,n): creates an mxn matrix whose elements are equal to one.

eye(m,n): creates an mxn identity matrix

rand(m,n): creates an mxn matrix whose elements are all random number between

Submatrix

Submatrix is frequently used in Matlab programming, make sure you are familiar with the following commends.

Define a Submatrix

A(i,j): returns the element of entry (i,j) in matrix A.
A(:,j): returns the jth column of A.
A(i,:): returns the ith row of A.
A(:,j:k): returns the submatrix of A consisting of th columns j, j+1, ...k.
A(i:k,:): returns the submatrix of A consisting of th rows i, i+1, ...k.
A(:,:): returns A unchanged

Tips: The colon operator (":") can be used as a way of generating row vector A:
A=a:i:b a is the starting number, b is the ending number, i is the increment.

Matrix Operations

MATLAB allows you to do operations on variables as single entities like in matrix-matrix multiplication as well as operations on the individual elements of the matrix.
The arithmetic operators for matrix are the same as the ones for scalar (“+”, “-”, “*”, “\”)

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \; B = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$$

$$A+B = \begin{pmatrix} 6 & 8 \\ 10 & 12 \end{pmatrix} \; A-B = \begin{pmatrix} -4 & -4 \\ -4 & -4 \end{pmatrix} \; A*B = \begin{pmatrix} 19 & 22 \\ 43 & 50 \end{pmatrix}$$

Elementwise Arithmetic Operations

Arithmetic operators can also be performed element-by-element (".*", ".\", ".^")

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \quad B = \begin{pmatrix} 5 & 6 \\ 7 & 8 \end{pmatrix}$$

$$A.*B = \begin{pmatrix} 1*5 & 2*6 \\ 3*7 & 4*8 \end{pmatrix} \quad A.\backslash B = \begin{pmatrix} 1\backslash 5 & 2\backslash 6 \\ 3\backslash 7 & 4\backslash 8 \end{pmatrix} \quad A.\wedge 2 = \begin{pmatrix} 1^2 & 2^2 \\ 3^2 & 4^2 \end{pmatrix}$$

Some Useful Functions on Matrix

A': gives the transpose of the matrix A.

det(A): gives the determinant of the square matrix A.

rank(A): givens the rank of A.

inv(A): gives the inverse of the square matrix A.

diag(A): gives a column vector containing the elements on the main diagonal of A.

chol(A): gives an upper triangular matrix which is the Cholesky factor of A.

Save and Load Data

Matlab can save and read the data in files suffix with ".mat".

Commands

save *filename* saves all variables on the file filename.mat.
save *filename* v1 v2 ... saves the variable v1, v2,... on file filename.mat.
load filename loads all variable form filename.mat into the work space.

You can also load other types of data: Microsoft Excel workbook, text file, etc.

Import Data from Excel Files

- xlsread(filename,xlRange) load the data from Excel file filename. xlRange specifies the data range (e.g. A1:C63).
- Or, you can import the data interactively

Graphics

Function plot(x,y) plots the vector y and vector x with x on the horizontal axis and y on the vertical axis.

example:
x= -1:0.05:1;
plot(x, sin(x))
You can obtain the plot of sine function of which the domain is [-1,1].

- To add new graphics onto the current plot, use the command "hold on" before the new plot function.
- To release the graphics, so the next plot will replace the current graphic, type the command "hold off".

Relation and Logical Operators

There are 6 relation operators to compare between variables:

$<$ smaller than	$<=$ smaller than or equal to
$>$ greater than	$>=$ greater than or equal to
$==$ equal to	$\sim=$ not equal to

The relational operators generate binary variables: 1 stands for that the comparison is **true**, 0 means comparison is **false**.

There are 3 logical operators in Matlab:

&	and	|	or	$\sim$	not

The logical operators have the lowest precedence of the operators. Both relational and arithmetic operations are performed prior to logical operations.

Conditional Statement

The most commonly used conditional statement is if statement:

Basic form

if *logical expression*
  statements
end

example:

```
if A(1,1)~=0
   A(1,:)=B(1,:);
end
```

According to the logical expression, if the first element in matrix A is not 0, then, elements in the first row of matrix A will be replaced by the first row of matrix B.

Conditional Statement

Extensions of if statement: else

if *logical expression*
  statements 1
else
  statements 2
end

The commands of statements 1 are executed if the logical expression is true, otherwise, the statements 2 will be executed.

example:

```
if A(1,1)~=0
   A(1,:)=B(1,:);
else
   A(:,1)=B(:,1);
end
```

Conditional Statement

Extensions of if statement: elseif

if *logical expression 1*
  statements1
elseif *logical expression 2*
  statements2
end

The statements 1 are executed if logical expression 1 is true, while the statements 2 are executed if logical expression 1 is false and logical expression 2 is true.

example:

```
if A(1,1)~=0
   A(1,:)=B(1,:);
elseif A(2,2)~=0
   A(:,1)=B(:,1);
end
```

Loops

Two loops commands are used in Matlab: for and while
Command "for" repeats a statement or group of statements predefined number of times. The statements are terminated by "end".

"for" loop

for *variable* = *expression*
  statements
end

example:

```
for i=1:2
   A(i,1)=100;
end
```

As a result, the first 2 elements in the first column of matrix A is replaced by 100.

Loops

Command "while" repeats a statement as long as a logical expression is true. The construction is terminated by "end".

"while" loop

while *logical expression*
  statements
end

example:

```
i=1;
while i<3
   A(i,1)=100;
   i=i+1;
end
```

Once again, the first 2 elements in the first column of matrix A is replaced by 100.

Script Files

M-file is a file with suffix ".m". It includes scripts files, function files, etc.
Life will be much easier if you store all the commands into one file. Such MATLAB command files are called script.

Steps

- In the menu, choose "new" and then "script"
- Write you program code
- Save the file by choosing the "save" from the menu
- Click "run" or press F5

Function Files

Matlab has a large amount of built-in functions. However, they still can't satisfy what you need in most of the cases. Therefore, you need to write you own function.
Function files are M-files that start with word "function". They can accept inputs arguments and return outputs.

Steps

- In the menu, choose "new" and then "function"
- Change the function name, input arguments, output arguments
- Write you program code
- Save the file by choosing the "save" from the menu
- Call the function in script file, command window or another function file

Example: Monte Carlo Simulation in Econometrics

Function File

```
function [ beta ] = ols( X,Y )
% returns to the OLS estimator of Y=beta*X
beta=inv(X'*X)*X'*Y;
end
```

Script File

```
x=rand(200,1); % Generate a series of x from uniform distribution
BETA=zeros(10000,1); % Define a 10000 by 1 matrix BETA
for i=1:10000
   y=0.5*x+randn(1); %Simulate y 10000 times. The error term is
drawn from standard normal distribution.
   BETA(i)=ols(x,y);
end
mean(BETA) % Calculate the mean of those 10000 estimators
```

Miscellaneous

- Define a matrix before the loop, otherwise, computational speed will be slowed down.
- Always remember to write your comments when you are programming.
- If you don't remember a build-in function, google it.
- Learn details of DSGE model/econometrics technique from reading code.

References

- Part-Enander, Eva (1996) "The MATLAB handbook", Prentice Hall
- Brian R. Hunt, Ronald L. Lipsman and Jonathan M. Rosenberg (2006) "A Guide to MATLAB - For Beginners and Experienced Users", Cambridge University Press
- Steven C. Chapra (2008)"Applied numerical methods with MATLAB for engineers and scientists", McGraw-Hill Higher Education