

Matlab Guide
Lab Guidelines for Econ 304 Students

1. How to Logon and start MATLAB:

Step 1: *Starting MATLAB:*

MATLAB is available on all of the computers in the Econ Computer lab.

Click the “Start” button (on the bottom left of the screen). From the list, select the “All Programs” icon. Then click on the the “MATLAB” icon.

2. How to Run the MATLAB Program.

Step 1: *Entering the “editor”*

When MATLAB starts up, 3 different windows will open up. Most of your work will be done in the “Command Window” and the “Debugger” (or, more correctly, MATLAB’s “M-file editor”). To move between the windows use the tabs in the task bar.

Step 2: *Creating a new MATLAB “program” or opening an already existing one.*

Select the “Command window,” goto the “file” menu and choose “open” to open a file already saved on your disk. If you feel adventurous, you can write a new MATLAB file using the “Debugger” by choosing “new” in the “file” menu item of the “Command window” (Be sure to save it!!!). When you are editing a program, maximize the window to give yourself more room (by clicking on the “window” icon, top-right-center, of the “Debugger” window).

The Debugger, just like SAS’s editor window, highlights the specific syntax that MATLAB uses. This is useful for noticing mistakes.

MATLAB will only run programs that are in specific directories (or on specific “paths”). To avoid updating the paths, just save the program that you are working on in the “work” directory within the “MATLAB6p5” directory.

Step 3: *To Run A Program.*

Note: MATLAB (like SAS) is mainly a code driven program. That is, to input data, and perform "Procedures" you write a “program” in text and MATLAB implements all of the features in that program.

3. An overview of MATLAB.

Step 1: *Introduction.*

The first window that you see is the command window. You can operate in this window (for example if you type 2+2 – or any calculation at the “>>” prompt and hit the “enter” key on the keyboard you will get the desired result) but it is not very useful for the type of projects that you will be concerned with. To be truly effective in MATLAB you write “programs”, just like in SAS. The *MATLAB Language* is like any standard language in that it has its own *vocabulary*. Again just like

SAS, a lot of the work is done for you by this language - that is, the standard procedures that econometricians use are already coded into the program so you just need to know what the right commands are. You can write these programs within MATLAB or you can use any text-editor to write MATLAB programs (as long as you save the programs as “text only”).

Step 2: *Editing Programs.*

As with SAS, the easiest way to learn how to use MATLAB is to use a pre-written program. I have supplied such a program on my web-page (called Econ304_eg1.m) – it also at the end of this document. If you want to alter this program in any way type “edit a:\ Econ304_eg1.m at the “>>” prompt – if you have the program on a disk (i.e. the “a – drive”). The program will then open up in the “Matlab debugger” (You can also edit the program in a text editor).

Step 3: *Debugging a Program.*

Once a program is opened, the syntax will be highlighted. The main use of this part of the MATLAB program is to run segments of the program to ensure that it is error free. You do this by moving the cursor to a place in the program that you think may pose a problem (say line 42). From the breakpoints menu you select “Set/Clear breakpoints”. A large red dot will appear alongside the line that you selected. This means that MATLAB will stop running this program at this point (you run MATLAB in the debugging window by hitting F5). You can tell from the “Command Window” if there are any errors up to this point. You can also tell from the debugging window, if there are no problems a green arrow will appear in front of the line where the breakpoint is. If there are errors, fix them. If not, you can “clear all breakpoints” (under the Breakpoints menu). Then you can set a new “breakpoint” or simply run the complete program (F5).

A Sample Program

```
% *****
% *****
%           - Econ_304.M - MYLES CALLAN - 21st of March 2003
%
%           Get into the habit of naming and dating your programs!!!
%
%           The Matlab version of the SAS program on the course webpage
%
%           This program "loads" the mon1.dat data set and performs some
%           simple analysis of this data.
% *****
% *****
% *****
% Loading the data to be used by this program
%
% Load the data (from the "a" drive), which has 426 rows and 5 columns
% This dataset is "Mon1" (case sensitive) for use within this program
% *****

load a:Mon1.dat -ascii;

% *****
% Transform variables into logs and log differences, creating new variables
%
% Note: you didn't have to name the variables when you loaded the data
% as you would in SAS (although you can). When referring to a variable you use
% the column numbers associated with each variable. So that M1 will be:
% Mon1[:,1] in the example below, i.e. the first column in the dataset
%
% ":" in each of the following transformations implies "all of the rows"
% or all of the columns, depending on the position in relation to the comma
% *****

lm1=log(Mon1(:,1));
% The natural log of M1, 1st column of Mon1 (NOTE: log to the base 10 is
% log10(Mon1(:,1));

dlm1=lm1(1:rows(lm1),:)-lag(lm1);
% The first difference of log M1

lm2=log(Mon1(:,2));
dlm2=lm2(1:rows(lm2),:)-lag(lm2);
% The same as the previous transformation for M2

% You can do this for the matrix as a whole
ldata=log(Mon1);
dlldata=ldata-lag(ldata);
% Note: dlm1 is the same as dlldata(:,1);

lip=log(Mon1(:,4));
dlip=lip(1:rows(lip),:)-lag(lip);

d0_m1=Mon1(1:rows(Mon1),1)-lag(Mon1(:,1));
d1_m1=lag(d0_m1);

d0_m2=Mon1(1:rows(Mon1),2)-lag(Mon1(:,2));
d0_ip=Mon1(1:rows(Mon1),4)-lag(Mon1(:,4));
d0_pr=Mon1(1:rows(Mon1),5)-lag(Mon1(:,5));
```

```

% concatenate some of the new variables with the original matrix

Mon1=[ Mon1 lm1 lm2 lip dlip];

% *****
% compute (x) means and (y) variance of all the variables
% *****
x=mean(Mon1);
y=var(Mon1);

% *****
% compute correlation coefficients (R) for all variables and the associated
% probabilities
% *****
[R, p]=corrcoef(Mon1);

% *****
% run a regression, note that intercept is automatically included
% the output of this regression will include t-statistics
% coefficient estimates, etc.
% NOTE: The prt command prints all of the results and the plt command plots
% the results.
% *****

result = ols(dlm1,dlm2);
prt(result);
plt(result);

beta=result.beta;

% *****
% The next commands tells Matlab WHAT TO display to the "output file"
% you do this by simply typing the variable names without a semicolon ";"
% *****
disp('means of the variables');
x
disp('variance covariance matrix of the variables');
y

disp('correlation matrix of the variables');
R
disp('probability of correlation matrix of the variables');
p

% *****
% The next commands tells Matlab WHAT TO write to the "output file"
% *****
save a:\temp.dat x y R beta -ascii

```