Basic Skills to Matlab

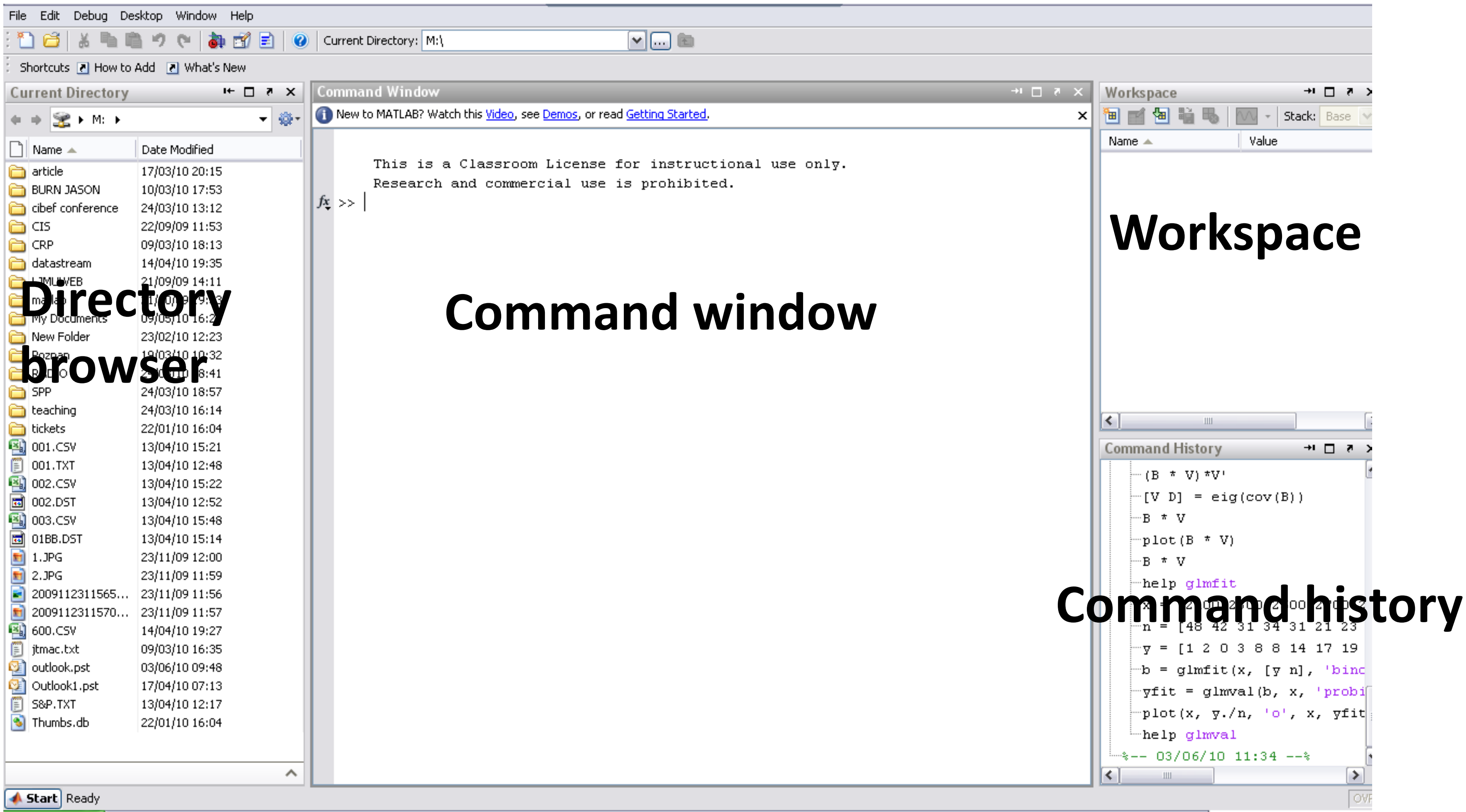
October 2009

By Binam Ghimire
Email: b.r.ghimire@ljmu.ac.uk

Contents

- Description of MATLAB parts
- Variables, functions
- Plotting
- Controlling program flow
- Importing data from excel
- Some useful econometric functions

MATLAB layout



Description of MATLAB (I)

- It can be thought of as a powerful calculator that can be programmed

- In addition it contains toolboxes (groups of functions) for Statistics, NN, GA, …

- Basic command: *help/doc command*

Description of MATLAB (II)

- Commands can be executed from a command window.

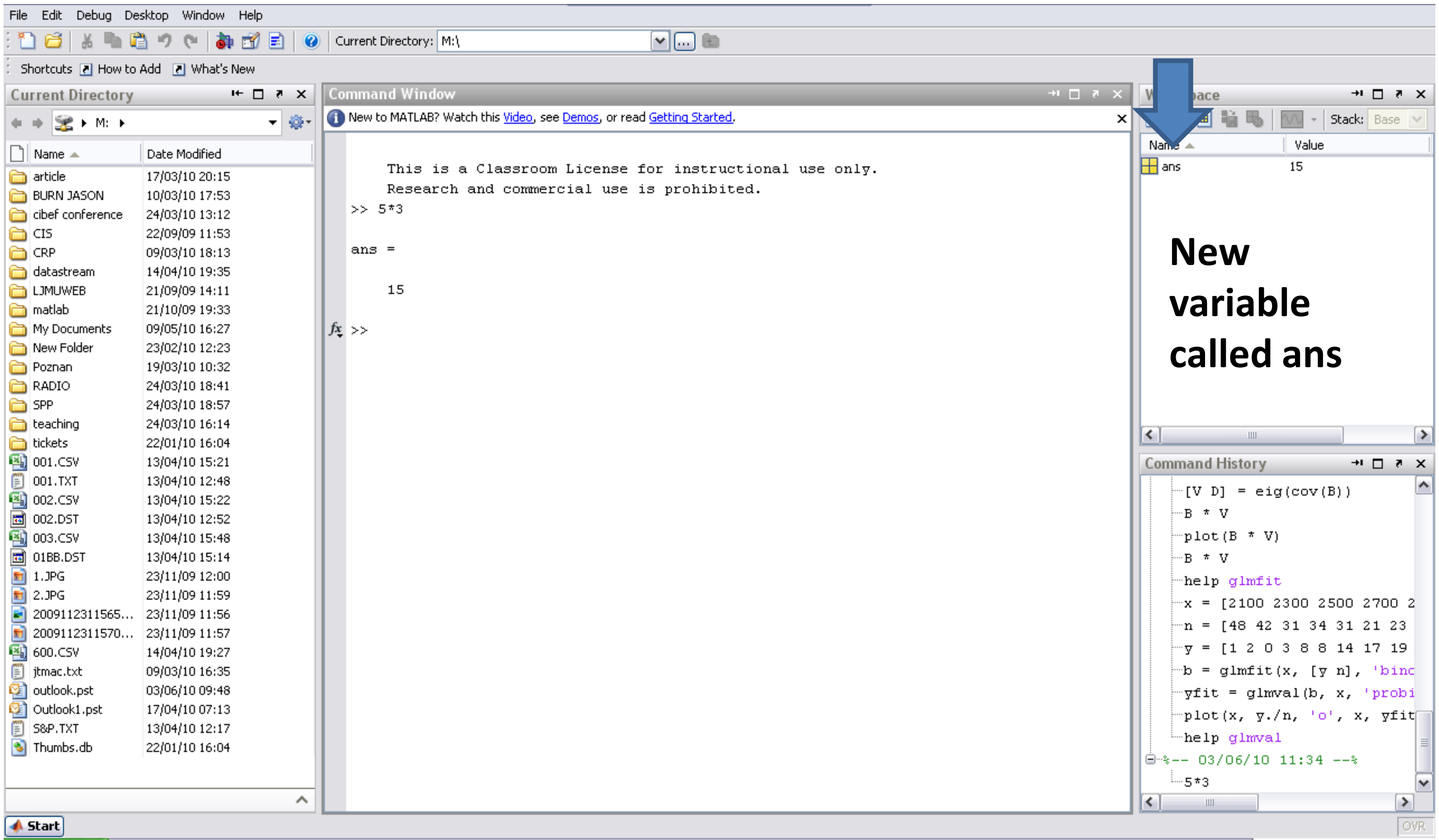
- However, it is more convenient to create new file: *file-new-script*. Inside the file write as many commands as you want. And execute.

- Execute script by **F5**. Execute only the selected line in the script by **F9**.

Description of MATLAB (III)
Command window – like calculator

- Write into command window any operation: e.g. 5*3.
- New variable *ans* is created.
- Basic operations include: +, -, *, /, log, exp, ^n,

Description of MATLAB (IV)



Description of MATLAB (V)
Script – like programming language

- Do the same as in the previous slide, but now create a script and type inside 5*3. Save it.
- Execute it using **F5**.
- Scripts are a convenient way of executing multiple operations by one button.
- TIP: Most of the time you will be creating scripts. However, when you are not sure whether the command will work, execute it in the command window first. Then add it to the script.

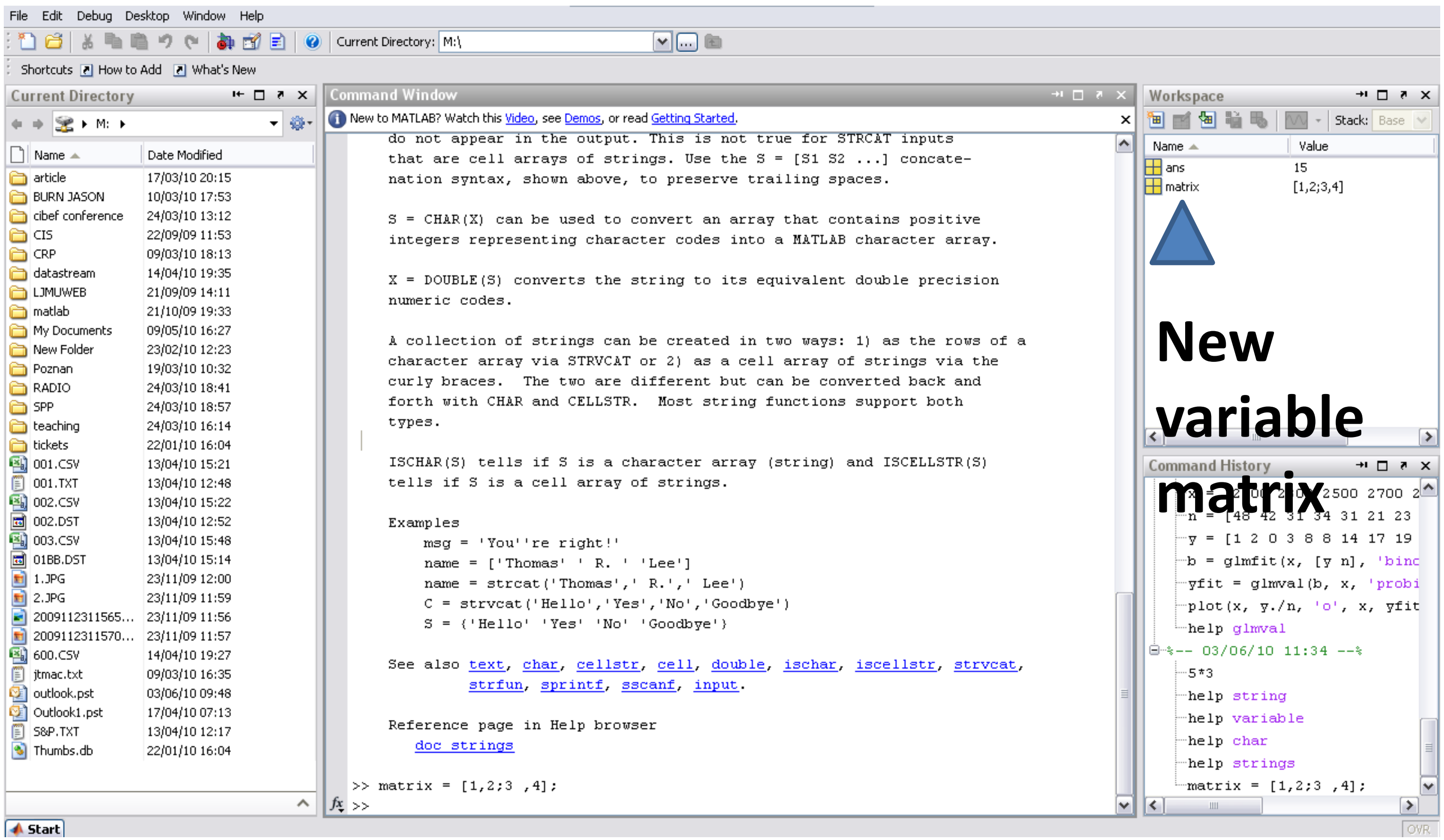
Variables (I)

- When you write *number* = 3, number is a variable.

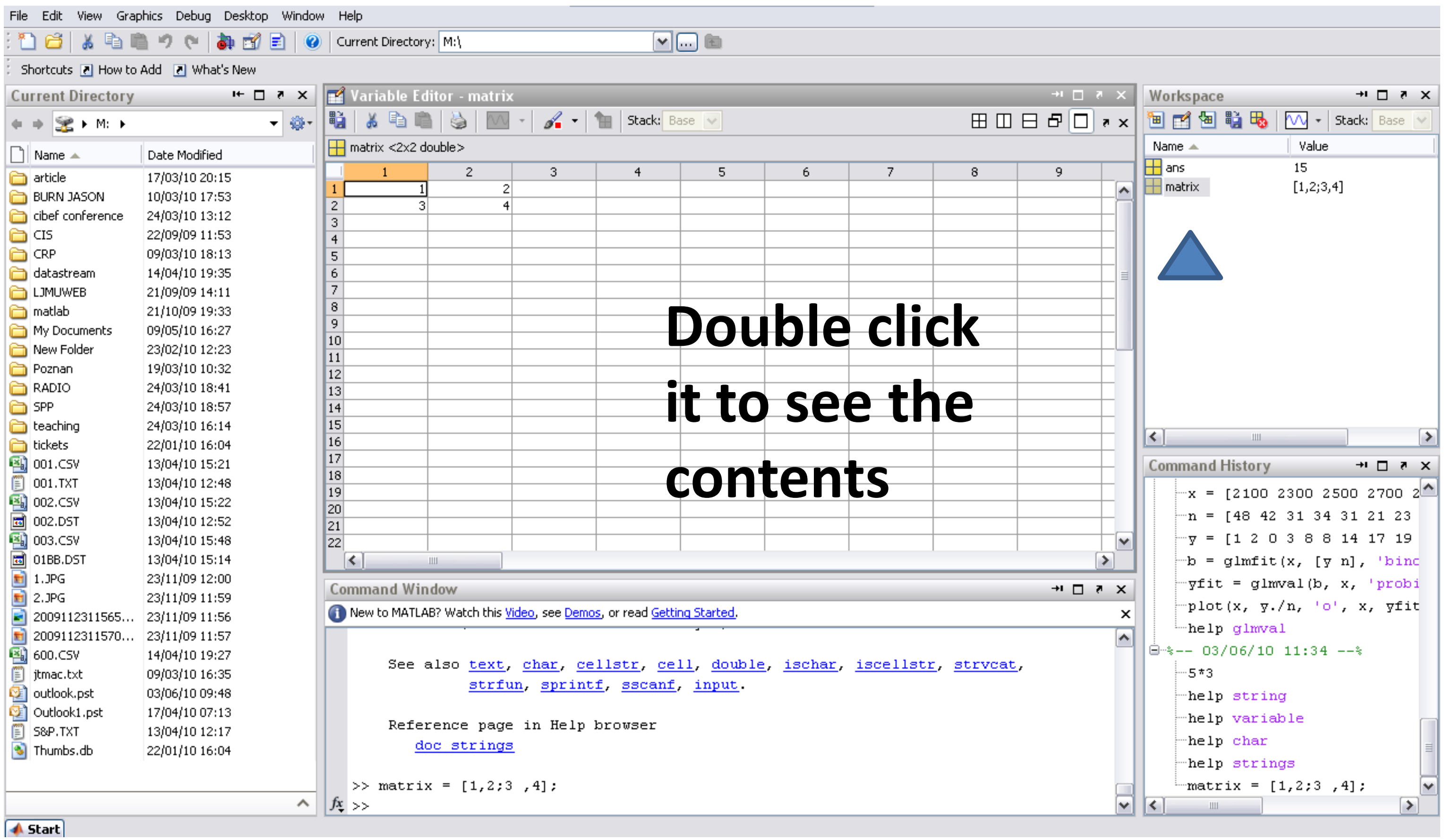
- There are different types of variables. Number in previous case is numeric. You can also write word = 'Liverpool'. This is called string. Just type help strings for more information on various types of variables.

- In Matlab it is really easy to work with vectors and matrices. Matrix is defined as *matrix* = [1,2;3 ,4]; This is a matrix with 2 rows and 2 columns.

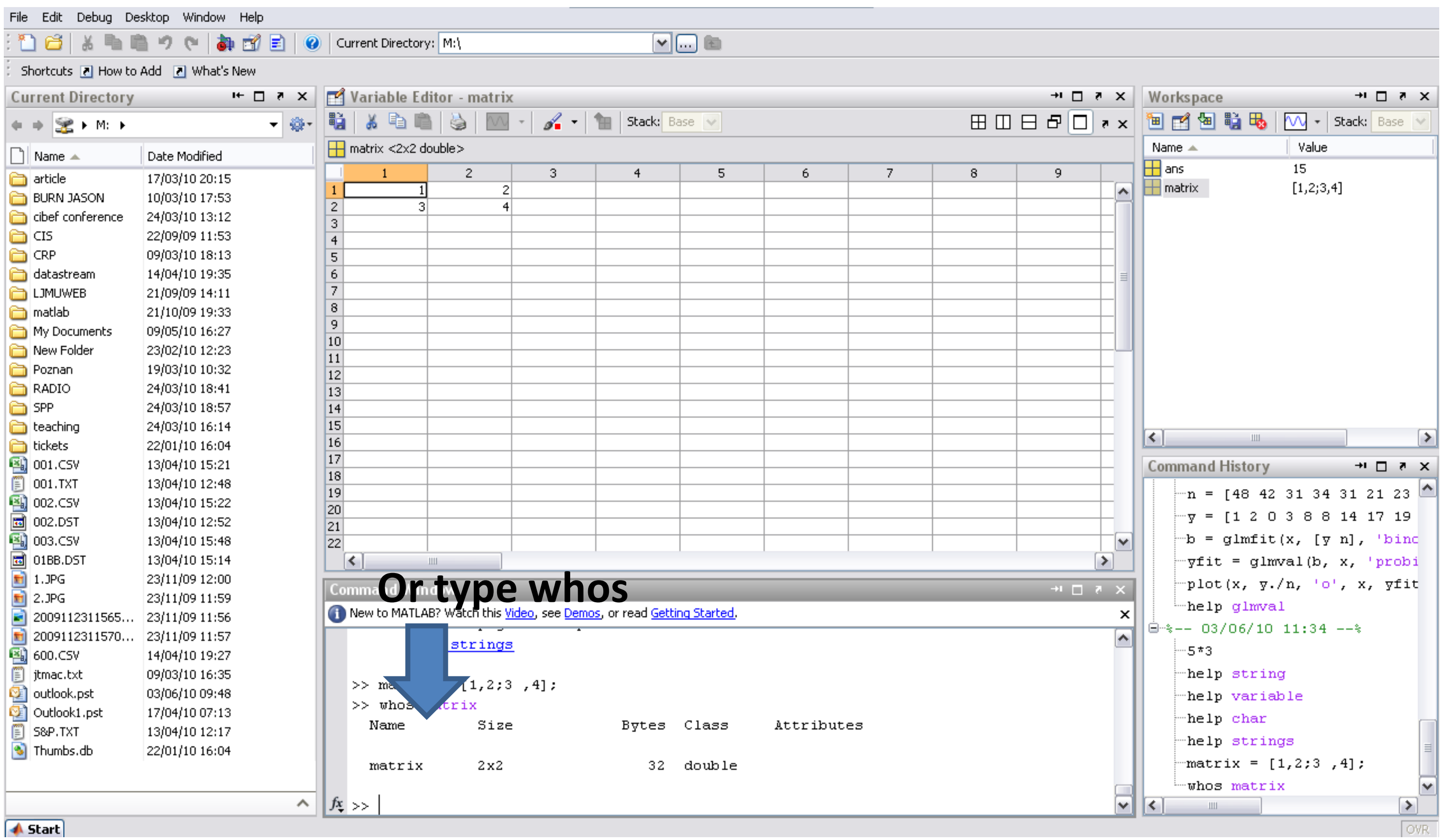
Variables (II)



Variables (III)



Variables (IV)



Variables (V)

- Transpose of a vector. A = a’.
- Vector with numbers from 1 to 8. A = [1:8];
- 1 to 8 with step 0.5. A = [1:0.5:8];
- In a vector you can access elements as e.g. A(1).
- In a vector, A(1,2) means element in 1st row and 2nd column
- In a vector A(:,1) means elements in all rows and 1st column
- Some predefined variables: *NaN, +/-Inf, pi, ans*.
- You can save/load variable: *save a, load a;*
- Use *clear* to clean the workspace.

Variables – Exercise (I)

- Create the variable as a row vector with values 2 5 20 30 50
- Create the variable as column vector with values 2 5 20 30 50
- Save these two variables to a file
- Clear the workspace
- Load the two variables you just created

Variables – Exercise (II)

1. Enter the following matrices and vectors

```
a =   9 12 13  0
     10  3  6 15
      2  5 10  3

b =   1  4  2 11
      9  8 16 7
     12 5  0 3
```

2. use a and b to create these matrices
   - a) c is the element in the 3rd row and 3rd column of a
   - b) d is column 3 of a
   - c) e is rows 1 and 3 of b
   - d) f is a and b one above each other
   - e) g is column 1 of a next to column 4 of b

Variables – Exercise (III)

3. change some of the entries in this matrices
   - a) make element (2,2) of e be 20
   - b) make row 1 of a be all zeros
   - c) make column 3 of f be the numbers from one to 6
   - d) make column 1 of a be the number from column 2 of b

Functions (I)

- E.g. sqrt(x) is a function. It is the same as writing x^(1/2).

- Some other important functions: *log, log10, exp, round, floor, ceil, abs, sum, cumsum, diag, max, min, mean, std, corrcoef, eigs, fix, round, floor, ceil, mod, factorial, hist, bar, scatter.*

- When using functions – very important commands: *help* and *doc*. E.g. type doc log.

Functions (II)

- Length(x) – find out the length of the vector.
- Size(x,1) – find number of rows.
- Size(x,2) – find number of columns.
- Vector multiplication *a**b*. Element-wise multiplication *a.*b*.

Functions - Exercise

- Multiply 2 vectors: a = [1,2,3] and b = [1,2,3]’.

- Multiply element-wise. a=[1,2,3] and b=[1,2,3].

Functions (III)

- Create vector of ones - ones(1,10).

- Create vector of zeros – zeros(1,10).

- Create vector of random variables – rand(1,10)

Functions (IV)

- Find minimum value of a matrix. Min(a).
- Maximum - Max(a)
- Create newNewVector = oldVector(oldVector>3) with only elements higher than 3.

Functions (V)

- Even you can program your own functions

Function output = NAME(input)

end

Functions – Exercise (I)

1) Enter the following matrices and vectors:

A = 1 5 6

3 0 8

B = 7 3 5

2 8 1

C = 10

D = 2

2) Do these sums:

E = A – B

*F = D*B*

*G = A.*B*

H = A'

Functions – Exercise (II)

3) Do some maths on parts of matrices

   a) Put the first column of A into M

   b) Put the second column of G into N

   c) Add them together

   d) Multiple ONLY the third column of A by C and put the result back in the third column of A. You can use several steps if you want, but it is possible do use just one line.

   e) Find the Dth row of H (i.e. row 2) and sum all the elements in that row

   f) Create a new matrix K made up of A in the first 2 rows and B in the next 2 rows.

4) Look at difference between array multiplication and matrix multiplication

   a) Try A*B (you will get an error)

   b) Try A*B'(that is, A matrix multiplied by B transpose), and compare to A.*B (that is, array multiplication of A and B)

5) Find the maximum values of each column of A, and find the minimum value of each row of B

6) Write a function with input variables matrices A and B, and the output variable the sum of them

Plotting (I)

- Plot the value of sin: x = -5:0.05:5; y=sin(x);plot(x,y); The graph will automatically connect the lines.
- You can also plot line using different style plot(x,y,’g.-’);
- Once the graph is printed – you can save it. Or an easy way – *print dmeta;* and paste into Word.
- To add more graphs on 1 figure: *hold on;*
- To start drawing a new figure: *figure(1); figure(2); etc.*
- Once the figure is plotted you can click – *show plot tools* to edit axes, zoom, add arrows, etc.

Plotting – Exercise (I)

- Make a script that will generate 1000 uniformly distributed values. Use *rand*.
- Plot histogram. Use *hist*.
- Now generate 1000 values from a standard normal distribution: use *randn*. Again plot the histogram.
- Draw functions of log base 10, natural log on the same graph
- Draw sin and 2*sin on the same graph.
- Produce different m-files for all the exercises here.

Plotting – Exercise (II)

Try plotting these graphs

a) Generate a vector X with values from 1 to 10

b) Generate a vector Y containing X squared.

c) Generate a vector Z containing X*9

d) In Figure 2, plot Y against X, using a red line with stars at each data point. (type help plot if you need to)

e) Keep that plot, and on the same graph plot Z against X using a green line with squares at each point

f) Give your figure a title and legend

Program flow (I)

- Sometimes some of you will create programmes of 100 or more lines. (e.g. when trying to backtest an advanced trading strategy).

- Sometimes you want some parts of code to run more times, sometimes not to run at all.

Program flow (II)

- There are special keywords to control the program run.

- If statement

```
If condition
   any commands
Elseif condition
   any commands
Else
   any commands
end
```

Program flow (II)

- If you want to run some iteration, and you know how many times you will run it.

```
for  i=1:10
   any commands
end
```

Program flow (III)

- Setting up breakpoints
- Stopping automatically when the program fails

Program flow - Exercise

1. Try writing loops
   a) Write a for loop to find the mean of 3 random numbers 20 times and place the result in a vector A every time
   b) Write a second loop to find the mean of 30 random numbers 20 times and place the result in vector B
   c) find the standard deviations of A and B
   d) plot A and B in different subplots so you can compare their distributions (try a histogram!) - type help for *subplot*

2. Write a function which will find the standard error of a vector, and test that it works.

Importing data

- From excel use [NUMERIC,TXT,RAW] = *xlsread('path to file','name of the sheet');*

- Exercise – try to import any stock ticker from yahoo. First save it as xls and then load it into MATLAB.

Econometric functions (I) – descriptive statistics

- Calculate log returns from the data imported (using price2ret)
- Find mean, median, std, min, max
- Plot histogram (hist), with 10, 100, 1000 bars

Econometric functions (II) - beta

- Download S&P 500 index data and try to calculate beta of a share. Remember: ShareRet = alpha+beta(MarketRet-Rf). Use the *B = regress(y,x)* function. For simplification we can ignore Rf.

Econometric functions (III) - GARCH

- Autocorrelation in squared returns: *autocorr(returns.^2);*

- Fit GARCH(1,1) process to the returns of the S&P500:

[coeff, errors, LLF, innovations, sigmas] = garchfit(returns);

- Fit ARMA(1,1)-GJR-GARCH(1,1):

spec = garchset(‘VarianceModel’,'GJR’,'R’,1,’M',1,’P',1,’Q',1);
[coeff, errors, LLF, innovations, sigmas] = garchfit(spec, returns);

- Plot volatility: *plot(sigmas).*

- Download ***^VIX*** index from yahoo and plot it against volatility estimated from ARMA(1,1)-GJR-GARCH(1,1). NOTE: you first need to multiply volatility from GARCH by some constant so that the levels are comparable.

Econometric functions (IV) - PCA

- Download 5-, 10- and 30-year bond yields from yahoo – (^FVX, ^TNX,^TYX)
- Make a matrix of 3 columns and *n* rows.
- Standardize the data (either by subtracting *mean* and divide by *std*)
  B = (A - repmat(AMean,[size(A,1) 1])) ./ repmat(AStd,[size(A,1) 1]);
- Calculate *[V D] = eig(cov(B)).* The matrix V contains the coefficients for the principal components. The diagonal elements of D store the variance of the respective principal components.
- To calculate the principal components, multiply the standardized data by the principal component coefficients: *B*V*
- To reverse transformation, multiply by the transpose of the coefficent matrix: (B*V)*V';
- To get back to the original data, multiply each observation by the sample standard deviation vector and add the mean vector:
  *((B * COEFF) * COEFF') .* repmat(AStd,[n 1]) + repmat(AMean,[n 1])*
- Try entire procedure, but use - VReduced = V(:,3) – only the first principal component.

Neural networks

- All neural networks from blackboard have the same file structure.

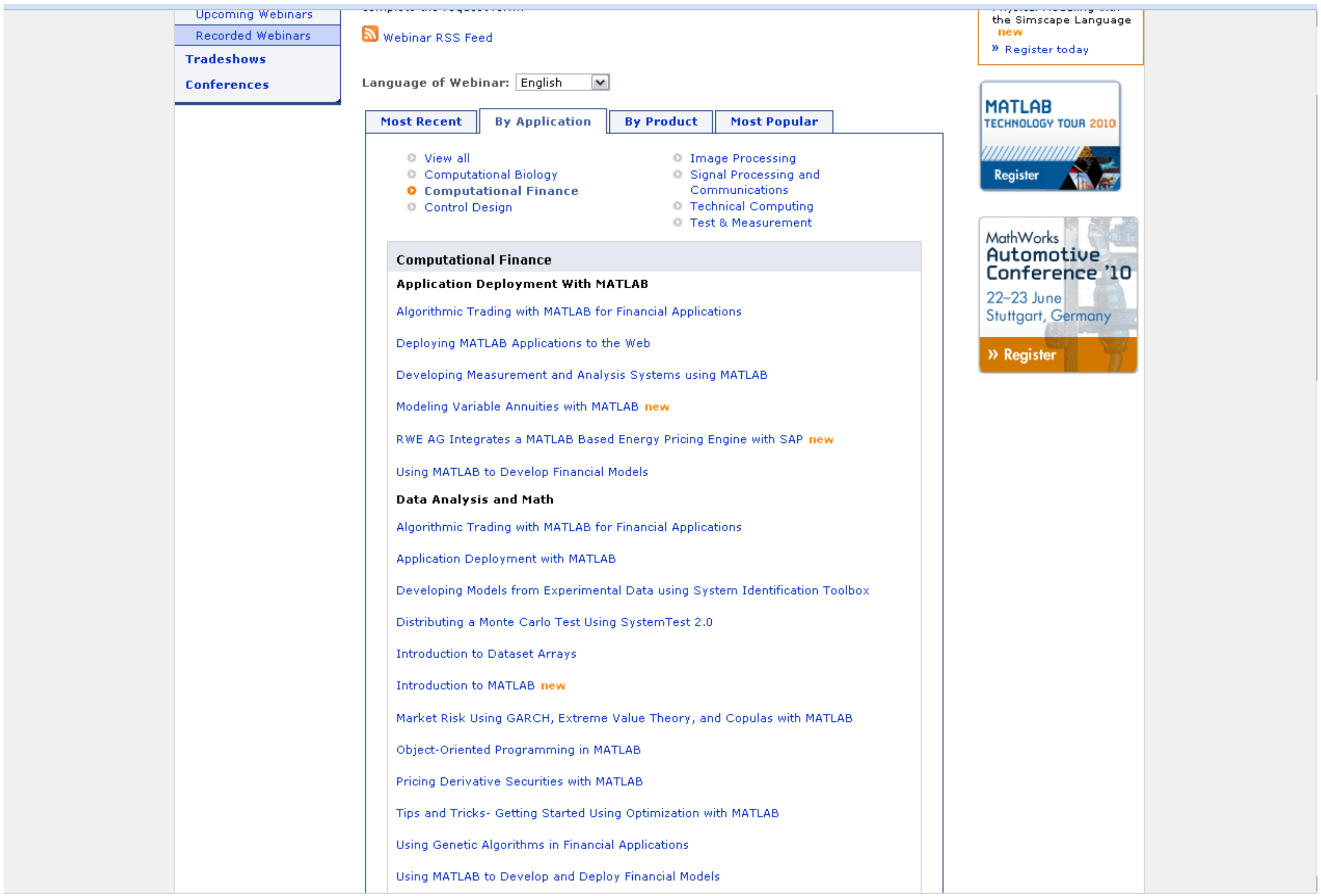
- Let’s focus on GM NN

- Main function is main_em_profit.

- Suggestion: Probably a more practical way of reading the inputs would be from excel - *[NUMERIC,TXT,RAW]= xlsread('FTSE250_GM.xls','Inputs');*

Where to go for other material

[www.mathworks.com/company/events/**webinars/**](http://www.mathworks.com/company/events/webinars/)



Resources used

- www.Mathworks.com
- http://matlabdatamining.blogspot.com
- http://www.math.utah.edu/lab/ms/matlab/matlab.html#starting
- http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-094-introduction-to-matlabae-january-iap-2009/lecture-notes/
- http://www.cs.ubc.ca/~murphyk/Teaching/Stat406-Spring07/lab1/matlab-exercises.pdf

[all accessed 2nd October 2009]