Matlab for Finance Course: Session 3

Dr. Peter A. Bebbington

Brainpool AI

peter@brainpool.ai

bebbington @peterbebbington

March 26, 2023

REVIEW OF SESSION 2

- Scripts and Editor
- Executing Scripts
- Debugging
- Arrays Inversion
- Boolean Logic and Control Flow
- Conditional Indexing
- Vectorise

OBJECTIVE

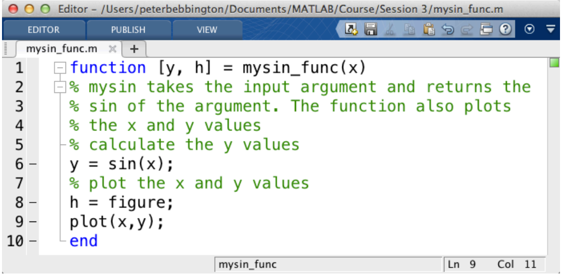
- File Functions
- Import Tool
- `hist_stock_data` Function
- Workspace
- Displaying Financial Data
- Stylized Facts

```
function[out1, out2, ...] = filename(arg1, arg2, ...)
% help comment
    statements
```

variables inside a function are local and cannot be accessed from the command prompt or another function

Example:

- `>> edit mysin_func.m`



- `>> [y, h] = mysin_func(0:pi/50:2*pi)`

DATA TYPES

Most financial data that will be imported into Matlab will come in three main forms

- .csv: Comma Separated Values
- .tsv: Tab Separated Values
- .txt: Text data in some formate

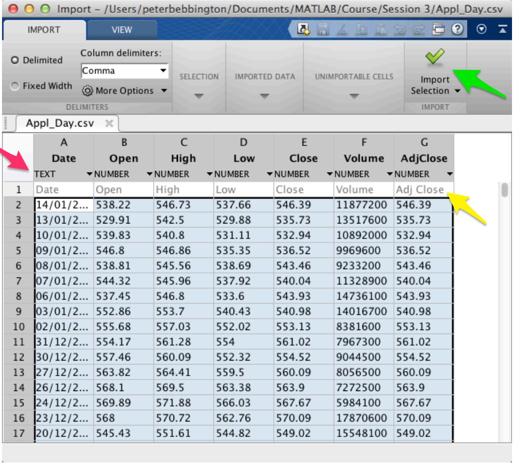
Other types of data that will be imported include .xls, .xml, .mat (can be loaded using `load data.mat` command) It is important to understand the organisation of different data types in order to understand the memory requirements for data

FILE FUNCTIONS

Command	Meaning
`fopen(filename)`	Open a file
`fclose(fid)`	Close a file
`fread(fid)`	Read binary data
`fwrite(fid,A,precision)`	Write binary data
`fprintf(fid,A,precision)`	Write formatted data
`fscanf(fid,format)`	Read formatted data
`sprintf(format,A)`	Write to a string
`sscanf(s,format)`	Read string
`ferror(fid)`	Query about errors
`foef(fid)`	Test for end of file
`fseek(fid,offset,origin)`	Set the file position indicator

IMPORT TOOL

- Simply drag and drop a ".csv" file to the command window of Matlab to import data

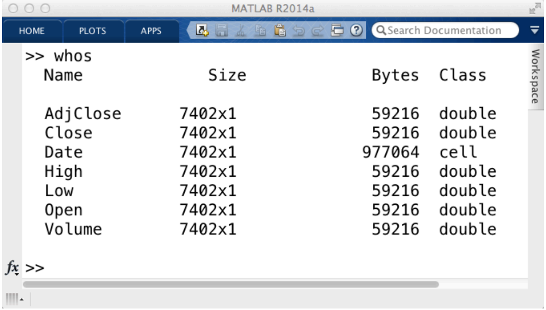


	A	B	C	D	E	F	G
	Date	Open	High	Low	Close	Volume	AdjClose
	TEXT	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER	NUMBER
1	Date	Open	High	Low	Close	Volume	Adj Close
2	14/01/2...	538.22	546.73	537.66	546.39	11877200	546.39
3	13/01/2...	529.91	542.5	529.88	535.73	13517600	535.73
4	10/01/2...	539.83	540.8	531.11	532.94	10892000	532.94
5	09/01/2...	546.8	546.86	535.35	536.52	9969600	536.52
6	08/01/2...	538.81	545.56	538.69	543.46	9233200	543.46
7	07/01/2...	544.32	545.96	537.92	540.04	11328900	540.04
8	06/01/2...	537.45	546.8	533.6	543.93	14736100	543.93
9	03/01/2...	552.86	553.7	540.43	540.98	14016700	540.98
10	02/01/2...	555.68	557.03	552.02	553.13	8381600	553.13
11	31/12/2...	554.17	561.28	554	561.02	7967300	561.02
12	30/12/2...	557.46	560.09	552.32	554.52	9044500	554.52
13	27/12/2...	563.82	564.41	559.5	560.09	8056500	560.09
14	26/12/2...	568.1	569.5	563.38	563.9	7272500	563.9
15	24/12/2...	569.89	571.88	566.03	567.67	5984100	567.67
16	23/12/2...	568	570.72	562.76	570.09	17870600	570.09
17	20/12/2...	545.43	551.61	544.82	549.02	15548100	549.02

- You can edit; data type (pink arrow), data field name (yellow arrow) and import data (green arrow)

WORKSPACE

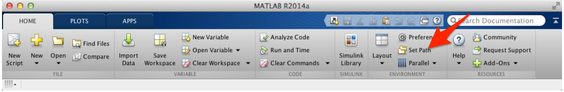
- Now that we have the data in Matlab we can create a workspace



- `>> clear`

HIST_STOCK_DATA

- Go to following url:
  `http://www.mathworks.co.uk/matlabcentral/fileexchange/18458-historical-stock-data-downloader/content/hist_stock_data.m`
- Download `hist_stock_data.m` and put in either the folder which is the current working folder or in the "MATLAB" folder (if the second you will need to add the path of "MATLAB" in the set path option or function `addpath('folderpath')`)



- One can see that using the hist_stock_data function data can be retrieved very easily by calling in the following form:
  `hist_stock_data('StartDate','EndDate','ticker1','ticker2',...)`

HIST_STOCK_DATA EXAMPLE

/Users/peterbebbington/Documents/MATLAB/Course/Session 3/importing_data.m

EDITOR PUBLISH VIEW

```
todaydatestr  = datestr(today,'mmddyyyy');
stock_data1   = hist_stock_data('01011990',todaydatestr,'KO','PEP');
price_series1 = [stock_data1(1).AdjClose(end:-1:1),stock_data1(2).AdjClose(end:-1:1)];
date1         = [stock_data1(1).Date(end:-1:1),stock_data1(2).Date(end:-1:1)];


stock_data2   = hist_stock_data('01011990','31121998','KO','PEP');
price_series2 = [stock_data2(1).AdjClose(end:-1:1),stock_data2(2).AdjClose(end:-1:1)];
date2         = [stock_data2(1).Date(end:-1:1),stock_data2(2).Date(end:-1:1)];
```

script Ln 4 Col 79

MATLAB R2014a

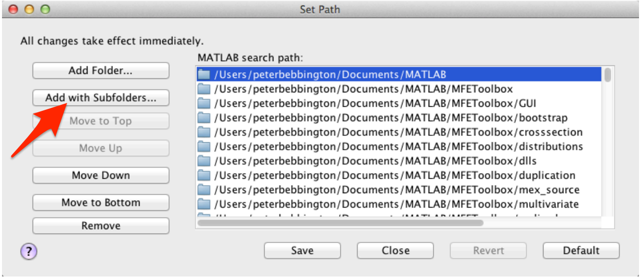
HOME PLOTS APPS Search Documentation

```
>> whos
  Name                  Size               Bytes  Class

  date1              6094x2              1608816  cell
  price_series1      6094x2                97504  double
  stock_data1           1x2              2196154  struct
  todaydatestr          1x8                   16  char

>> stock_data1
stock_data1 =
1x2 struct array with fields:
    Ticker
    Date
    Open
    High
    Low
    Close
    Volume
    AdjClose
>>
```

Workspace

- Matlab has its own Econometrics toolbox which rich functionality
- There are also third party toolboxes that can be installed which can help for summer project if it involves time series (most will)
- The two toolboxes I use are MFEToolbox and jplv7 which can be found on http://www.kevinsheppard.com andhttp://www.spatial-econometrics.com respectively

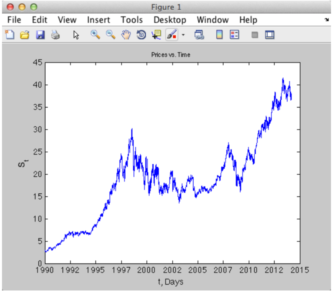
- As we did for `hist_stock_data` put the toolboxes in a folder sensible such as the Matlab folder in "My Documents" or Documents.

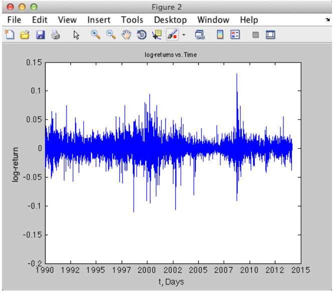


- Click "Add with Subfolders..." (red arrow) and Locate the two toolboxes and save.

FINANCIAL SERIES

- A good to start when analysing financial time series is to simple plot the price against time quantises such as price, log-returns, volume, etc...





- Sample statistics will give you an idea about the statistical behaviour of such as the moments and underlying distribution

Editor – /Users/peterbebbington/Documents/MATLAB/Course/Session 3/sample_statistics.m

sample_statistics.m

```
mean(lreturns(:,1))
std(lreturns(:,1))
min(lreturns(:,1))
max(lreturns(:,1))
skewness(lreturns(:,1))
kurtosis(lreturns(:,1))
sacf(lreturns(:,1),1,[],0)
sacf(lreturns(:,1).^2,1,[],0)
jarquebera(lreturns(:,1))
[q1, pval1] = ljungbox(lreturns(:,1),20)
[q2, pval2] = ljungbox(lreturns(:,1).^2,20)
kstest(lreturns(:,1))
```

NORMALIZING

- Any Gaussian distributed random variable can be normalized:

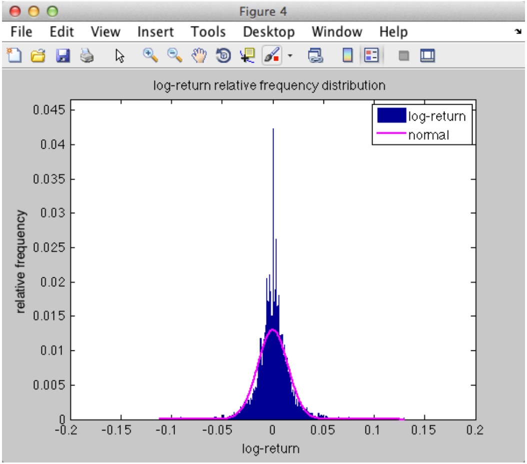
$$X \sim \mathcal{N}(\mu, \sigma^2)$$

$$Z = \frac{X - \mu}{\sigma}$$

$$X = \sigma Z + \mu$$

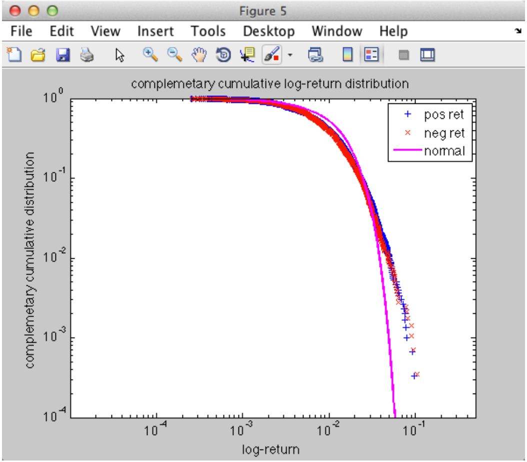
- Analysis of return time series is better in this form for comparison between different time series such as a portfolio

- Here we make a comparison of the empirical histogram against a parametrized normal distribution

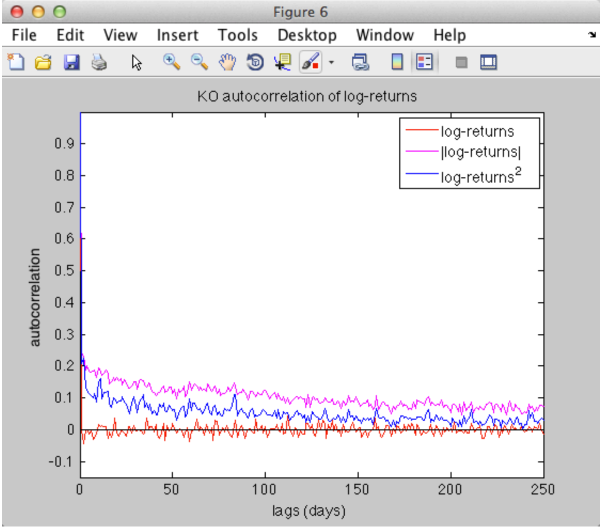


COMPLEMENTARY CUMULATIVE DISTRIBUTION

- We see in this log-log plot the empirical time series differs from the tails of a normal distribution



AUTOCORRELOGRAM



VOLATILITY

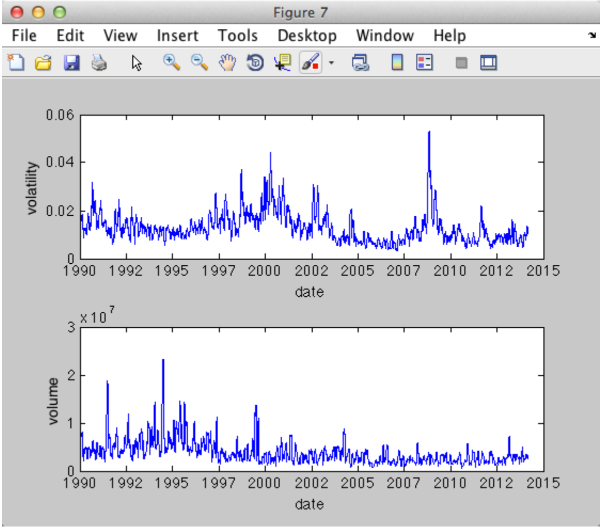
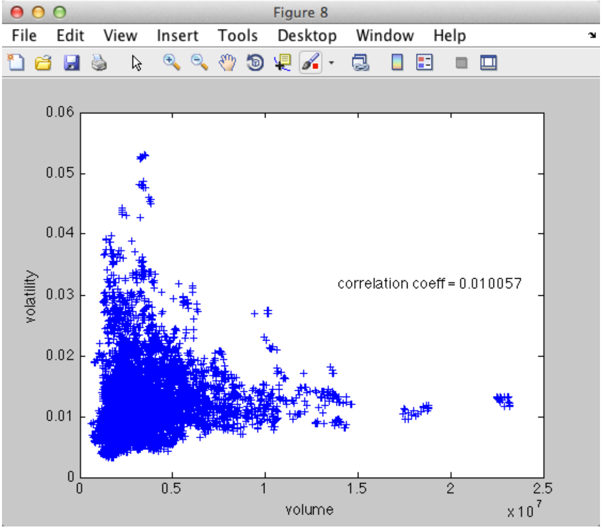


Figure 7
File Edit View Insert Tools Desktop Window Help
0.06
0.04
0.02
0
volatility
1990 1992 1995 1997 2000 2002 2005 2007 2010 2012 2015
date
x 10^7
3
2
1
0
volume
1990 1992 1995 1997 2000 2002 2005 2007 2010 2012 2015
date

VOLATILITY Vs. VOLUME



QQPLOT

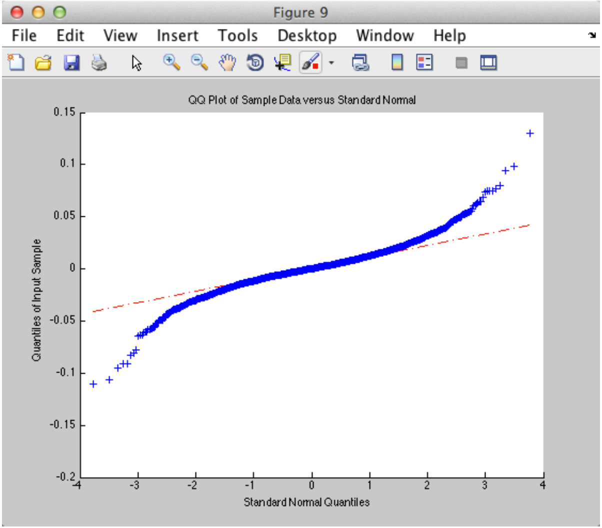


Figure 9
File Edit View Insert Tools Desktop Window Help
QQ Plot of Sample Data versus Standard Normal
Quantiles of Input Sample
Standard Normal Quantiles
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-4
-3
-2
-1
0
1
2
3
4

VAR PARAMETRIC

- Given some confidence level $\alpha \in (0, 1)$ Value-at-Risk is defined as the following quantile:

$$\text{VaR}_\alpha \triangleq \inf\{l \in \mathbb{R} : F_L(l) \geq \alpha\}$$

- Value-at-Risk is estimated parametrically assuming a normal distribution in Matlab with the following function `ValueAtRisk = portvrisk(PortReturn, PortRisk, RiskThreshold, PortValue)`
- Note that one should calculate this quantity empirically to compare to the parametric estimation.

PARAMETRIC CVAR

- For a loss (L) with $\mathbb{E}[|L|] < \infty$ and distribution for F_L, the expected shortfall at the confidence level $\alpha \in (0,1)$ is defined

$$\mathrm{ES}_\alpha = \frac{1}{1-\alpha}\int_\alpha^1 \mathrm{VaR}_u(F_L)\,\mathrm{d}u$$

- If the loss distribution is assumed to be a normal, one can calculate the parametric CVaR as the follows

/Users/peterbebbington/Documents/MATLAB/Course/Session 3/CVaR.m

```
display('------------------------------')
display('Parametric CVaR')
m = mean(lreturns(:,1));
s = std(lreturns(:,1));
CVaR_95=m+s*(normpdf(-norminv(0.05,m,s),0,1))./(1-0.95);
CVaR_99=m+s*(normpdf(-norminv(0.01,m,s),0,1))./(1-0.99);
display(sprintf('CVaR 95: %.9f', CVaR_95))
display(sprintf('CVaR 99: %.9f', CVaR_99))
display('------------------------------')
```