Financial Econometrics
MFE MATLAB Introduction

Kevin Sheppard
University of Oxford

September 27, 2010

Contents

Lesson 1

Preliminaries

This lesson covers:

- Connecting to the server
- Launching MATLAB
- Launching the editor
- Creating a `startup` file
- Using MATLAB with a laptop or other computer
- Moving files to and from the server

Task 1.1 (Complete the Registration Form) Enter your name, single sign-on ID and

Connecting to the Server

If using the economics department computer lab, connecting is simple: click on

```
mrb-teachinglab.manor-road.ox.ac.uk
```

This will connect the thin client to one the servers. There are a number of physical servers behind the cluster name and the system will automatically load balance users. Following a few basic principles will ensure that using the servers is fairly painless:

1. Save relatively often. Occasionally connections to the server are unexpectedly interrupted and work can be lost. Work saved in personal storage (U:\) will not.
2. Always save code in the editor. The command history should be preserved across logons but unexpected disconnections may results in changes not being saved.
3. If accidentally disconnected, try reconnecting. In many cases, the previous session will be restored and anything unsaved will be preserved.

4. Despite the point above, do not count on being reconnected to the previous session if unintentionally disconnected – hence, save often.

5. When finished using the server, always logoff by either clicking the logoff icon on the desktop or selecting Start>Logoff. This ensures that your settings, such as the command history in MATLAB, will be restored when you logon again.

These principles apply whether connecting from a laptop or from the computer lab.

Task 1.2 (Connecting to the server) Connect to the server, logon and change your password to something memorable but not easy to guess.

Note: You must complete the Student Registration form before you receive an account. If you have not completed this form you will be using a temporary username and password.

Remember your password. You should probably write it down somewhere safe. If you misplace your username and/or password, please contact itsupport@manor-road.ox.ac.uk for assistance.

Task 1.3 (Create/Verify the Work Directory) Open My Computer and navigate to the U drive. The default directory should be U:\MATLAB. If this directory exists then there is nothing to do. If this directory does not exist, create a folder with the name MATLAB on the U drive.

Note: If you are using one of the temporary account you will have a H drive (not U).

Launching MATLAB

To launch MATLAB, select Start>Programs>Economics>MATLAB>R2010a>MATLAB R2010a.[1] When MATLAB opens, a window similar to figure 1.1 should be present, although the contents of the left panes may vary.

Task 1.4 (Launching MATLAB) Open MATLAB on your terminal.

Launch the Editor

Once MATLAB is up and running, launch the editor. There are two methods to accomplish this task

- Enter `edit` in the command window
- Use the menu via File>New>M-File.

The editor should appear similar to figure 1.2.

Task 1.5 (Launch the Editor) Open the editor using one the methods above. Once the editor is open, create a file with the following contents:

```
x = exp(1)
y = log(x)
```

and save it as `lesson1.m`. Go to the command window and enter `lesson1`. The command window should show

[1]The version may be different depending on which server being used.

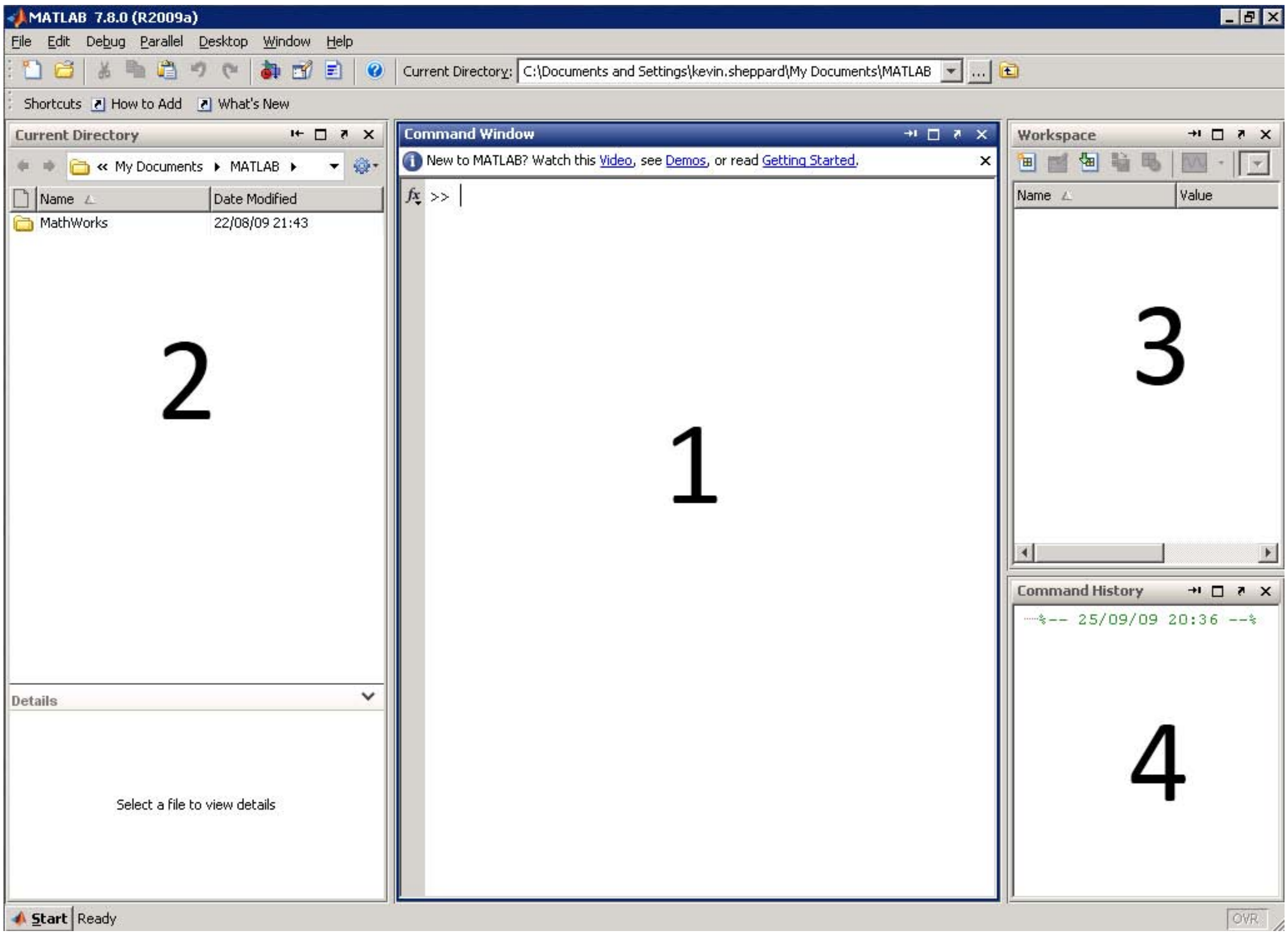

MATLAB 7.8.0 (R2009a)
File Edit Debug Parallel Desktop Window Help
Current Directory: C:\Documents and Settings\kevin.sheppard\My Documents\MATLAB
Shortcuts How to Add What's New
Current Directory
« My Documents ▸ MATLAB ▸
Name
Date Modified
MathWorks
22/08/09 21:43
2
Details
Select a file to view details
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
>>
1
Workspace
Name
Value
3
Command History
%-- 25/09/09 20:36 --%
4
Start Ready
OVR

Figure 1.1: Basic MATLAB Window. The standard setup has four pains. **1**: The command window, **2**: Current Directory, **3**: Workspace, and **4**: Command History

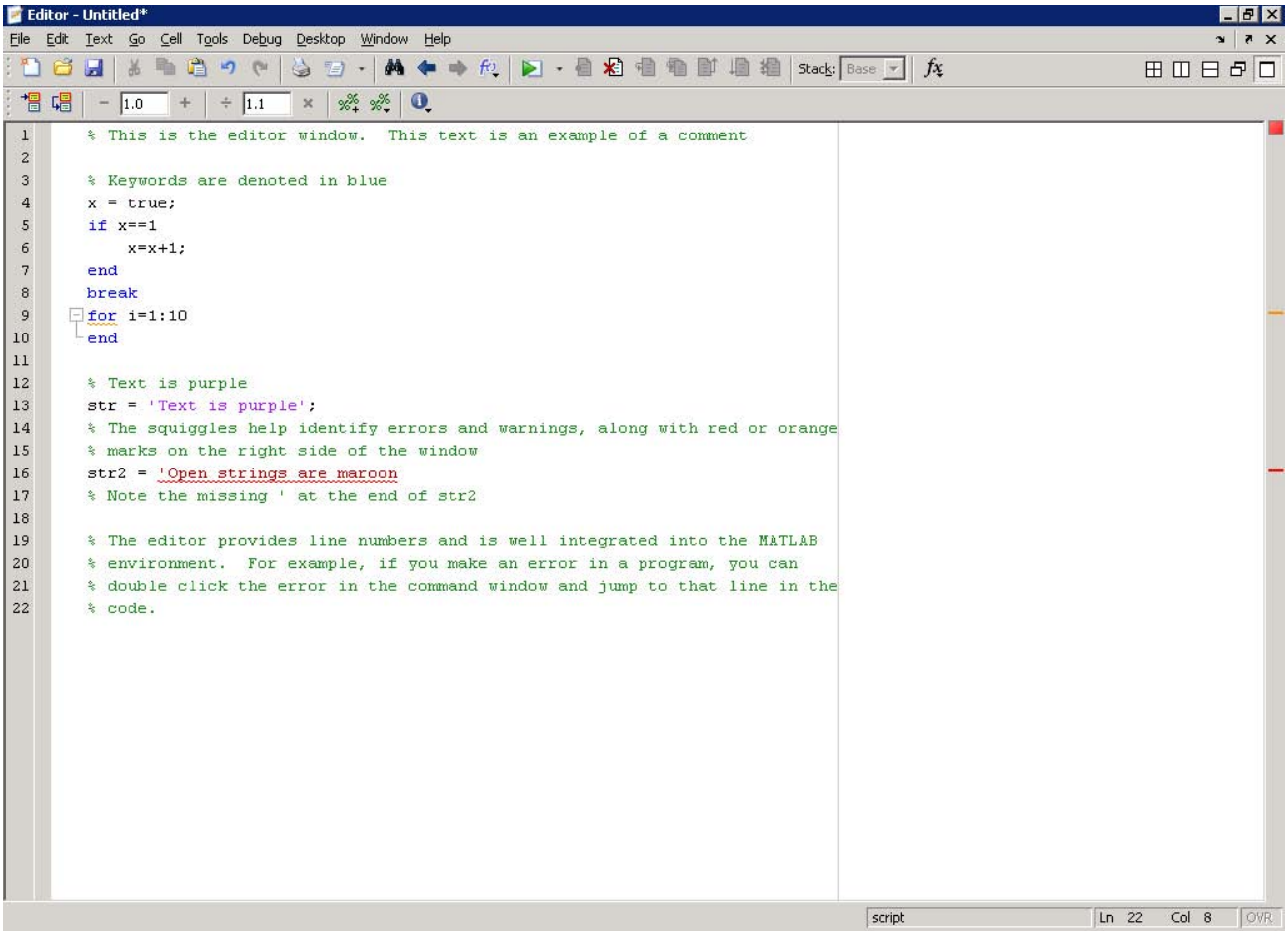
```
x =
    2.7183

y =
     1
```

Task 1.6 (Edit an Existing File) Next, close the editor by either clicking on the "X" in the top right corner or selecting File>Close Editor. A specific file can be opened in the editor by entering `edit` *filename* in the command window. To verify this works, try the following:

```
>> edit lesson1
```

Create a `startup` file

Note: If you are using one of the temporary account you will have a H drive (not U), and so all references to U:\ should be H:\. It is important to repeat this step on your own once you have a permanent account.



```
% This is the editor window.  This text is an example of a comment

% Keywords are denoted in blue
x = true;
if x==1
    x=x+1;
end
break
for i=1:10
end

% Text is purple
str = 'Text is purple';
% The squiggles help identify errors and warnings, along with red or orange
% marks on the right side of the window
str2 = 'Open strings are maroon
% Note the missing ' at the end of str2

% The editor provides line numbers and is well integrated into the MATLAB
% environment.  For example, if you make an error in a program, you can
% double click the error in the command window and jump to that line in the
% code.
```

Figure 1.2: The editor

A large number of functions designed for use on the course have been written and stored in a common location. To ensure that these files are available every time MATLAB is run, create an m-file names startup in U:\MATLAB. To accomplish this task, return to the MATLAB command prompt and input

```
>> cd U:\MATLAB
```

If you receive an error you may need to create the folder, which can be accomplished using command windows by entering

```
>> mkdir('U:\MATLAB')
```

If you needed to make the directory, you should rerun the `cd` command above and ensure that the Current Directory windows at the top of the MATLAB desktop shows U:\MATLAB. To create the `startup` file, open it in the editor by inputting

```
>> edit startup
```

You will be prompted to to create this file. Select Yes.

```
path('Q:\Economics\MFE',path)
path('\\manor-road.ox.ac.uk\Store\Shared\Economics\MFE',path)
```

When finished, save the file in U:\MATLAB (that path should already be correct) as U:\MATLAB\startup.m. To verify that you were successful, **close** and **reopen** MATLAB, the run the following command

```
>> which acf -all
```

The output should be

```
\\manor-road.ox.ac.uk\Store\Shared\Economics\MFE\acf.m
Q:\Economics\MFE\acf.m                                % Shadowed
```

If you see `'acf' not found.` something has gone wrong.

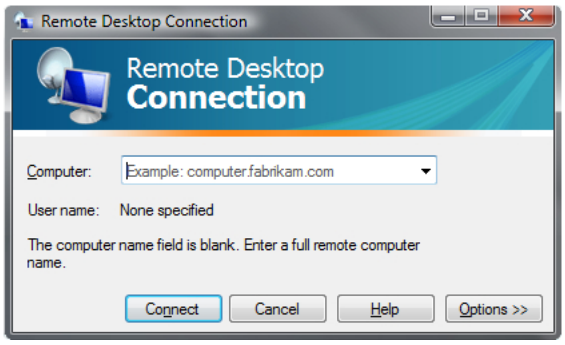
Using MATLAB with a Laptop or Other Computer

Note: *These instructions are for the latest Windows Remote Desktop client. It is important that the latest client is used. If running Windows 7 or Vista, the client is current. If running Windows XP, run Windows Update to determine if a newer client is available. If running Linux or OSX, consult the material on the course website to connect to the server.*

The Terminal Servers are grouped into a load balancing pool and all servers within a pool are similarly configured. You should usually connect to `mrb-studentlab.manor-road.ox.ac.uk` which balances between:

DNS Name	IP Address	Processor	Available Memory
MRB-StudentTS01	129.67.98.235	2GHz Xeon	12 GB
MRB-StudentTS02	129.67.98.236	2GHz Xeon	12 GB

To setup a connection, select Start>All Programs>Accessories>Remote Desktop Connection or run `mstsc` from a command prompt. This will launch the Remote Desktop Client (RDC) and the following window will appear:

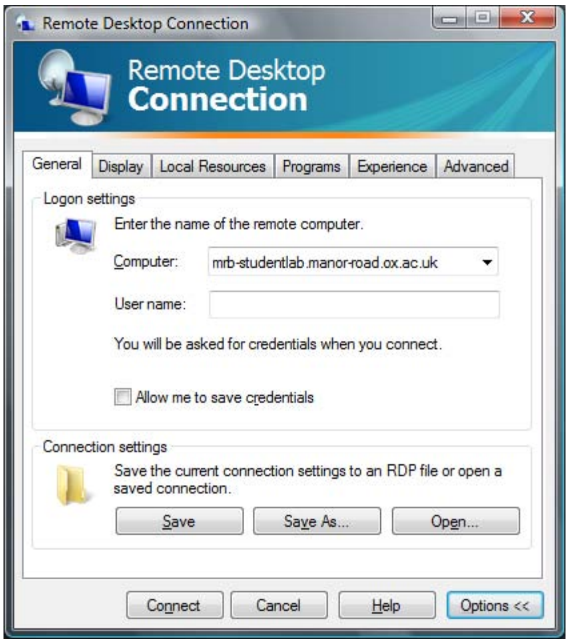


Computer: Example: computer.fabrikam.com

User name: None specified

The computer name field is blank. Enter a full remote computer name.

Once the client is open, click Options>>. This should produce:



Logon settings

Enter the name of the remote computer.

Computer: mrb-studentlab.manor-road.ox.ac.uk

User name:

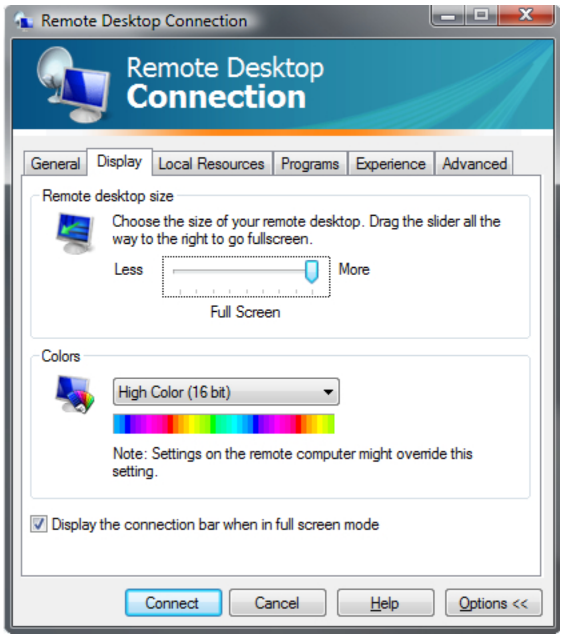
You will be asked for credentials when you connect.

☐ Allow me to save credentials

Connection settings

Save the current connection settings to an RDP file or open a saved connection.

In the box labeled computer, enter `mrb-studentlab.manor-road.ox.ac.uk`. Next click on the Display tab.



Remote desktop size

Choose the size of your remote desktop. Drag the slider all the way to the right to go fullscreen.

Full Screen

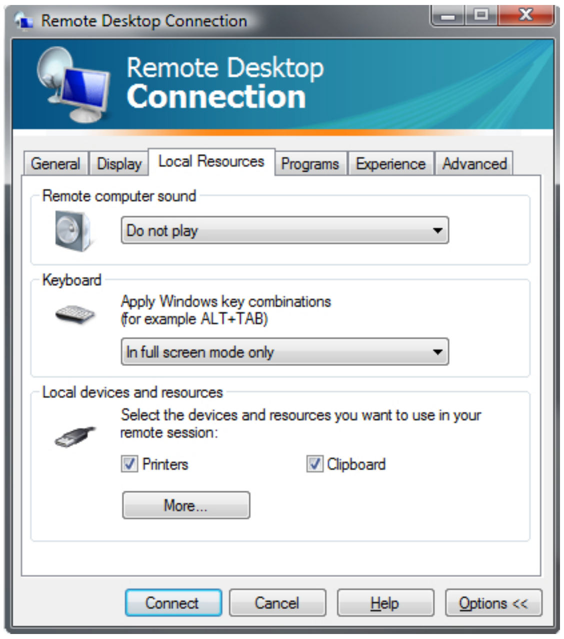
Colors

High Color (16 bit)

Note: Settings on the remote computer might override this setting.

☑ Display the connection bar when in full screen mode

The default options should be acceptable, although if you are having problems you may want to decrease the color bit depth or lower the resolution to something less than Full Screen. These two options may improve performance on slow connections. Next click on the Experience tab.


Remote Desktop Connection

Remote Desktop Connection

General | Display | Local Resources | Programs | Experience | Advanced

Remote computer sound

Do not play

Keyboard

Apply Windows key combinations
(for example ALT+TAB)

In full screen mode only

Local devices and resources

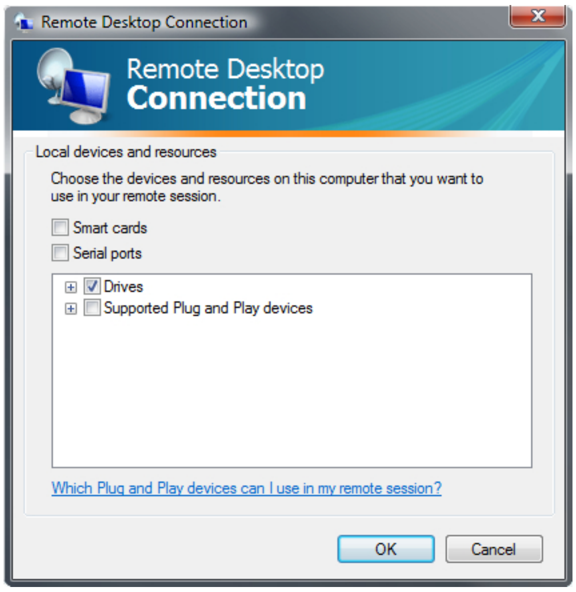
Select the devices and resources you want to use in your remote session:

☑ Printers ☑ Clipboard

More...

Connect | Cancel | Help | Options <<

First set the sounds to Do Not Play. Next make sure Printers and Clipboard are ticked. Finally click More... which will produce the dialog below.


Remote Desktop Connection

Remote Desktop Connection

Local devices and resources

Choose the devices and resources on this computer that you want to use in your remote session.

☐ Smart cards

☐ Serial ports

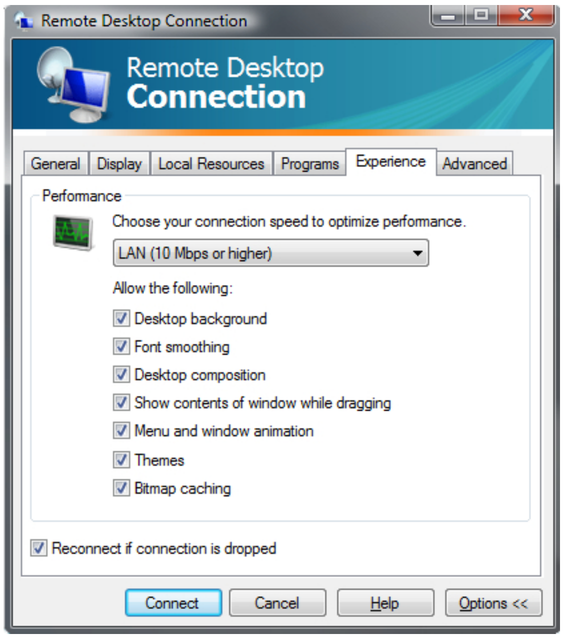
⊞ ☑ Drives

⊞ ☐ Supported Plug and Play devices

Which Plug and Play devices can I use in my remote session?

OK | Cancel

In this dialog box, check the box next to disks. This will allow you to easily move files between your own computer and the server. When available, your local drives (those on your desktop or laptop) will appear in My Computer on the server. Next click on the Local Resources tab.


Remote Desktop Connection

Remote Desktop Connection

General | Display | Local Resources | Programs | Experience | Advanced

Performance

Choose your connection speed to optimize performance.

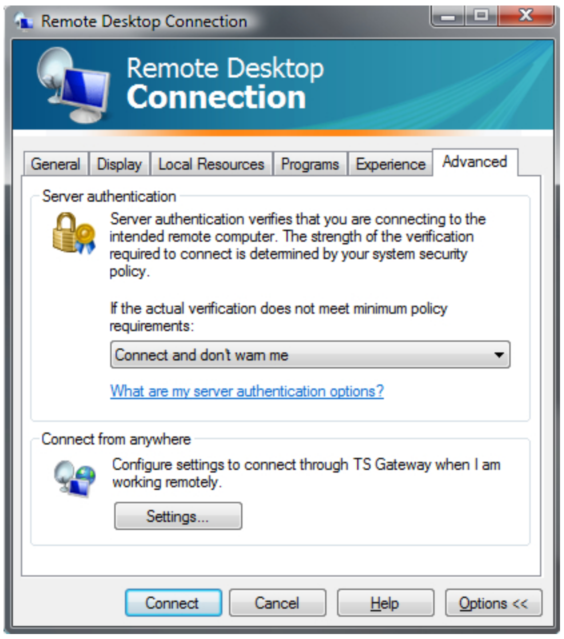
LAN (10 Mbps or higher)

Allow the following:

- [✓] Desktop background
- [✓] Font smoothing
- [✓] Desktop composition
- [✓] Show contents of window while dragging
- [✓] Menu and window animation
- [✓] Themes
- [✓] Bitmap caching

[✓] Reconnect if connection is dropped

You should start off with LAN. If the connection to the server is sluggish or you are not on campus, you may try one of the other settings which may improve performance. Finally, click on the Advanced tab.


Remote Desktop Connection

Remote Desktop Connection

General | Display | Local Resources | Programs | Experience | Advanced

Server authentication

Server authentication verifies that you are connecting to the intended remote computer. The strength of the verification required to connect is determined by your system security policy.

If the actual verification does not meet minimum policy requirements:

Connect and don't warn me

What are my server authentication options?

Connect from anywhere

Configure settings to connect through TS Gateway when I am working remotely.

On the advanced tab, make sure that “Connect and don’t warn me” is selected. When finished, returns to the General tab and save the connection using a meaningful name (e.g. Economics Server).

***Note**: You are required to change your password the first time you connect. Passwords must contain*

- *At least 7 characters*
- *At least 1 UPPER and 1 lower case letter*
- *At least one number (1, 2, . . .) or 1 symbol (#, $, %, etc.)*

In addition, passwords cannot be common English language words.

Disconnecting your session

It is crucial that you log off when finished, even for a few minutes, and do not disconnect. There are no guarantees that disconnecting will preserve the session or any unsaved work. It is also important to save regularly in case the network connection is interrupted.

Moving files to and from the server

There are two primary methods to move files to and from the server. The first, using the Remote Desktop Client, is described above. Once sharing drives is enabled, the drives of the local laptop or desktop will appear in My Computer on the server. These can be browsed like any other drive in Windows and files can be cut, copied and pasted between the server and the local computer.

Mapping a network drive

Open My Computer and select Tools>Map Network Drive. Select the drive letter you would like to use and enter \\manor-road.ox.ac.uk\store\economics\students\data\user.name (where user.name is you username, e.g. kevin.sheppard) for the folder and select Finish. When prompted for a username and password, enter MANOR-ROAD\user.name and your password.
Note: It may not be possible to map or access this resource from all locations due to firewalls. As a results, you should be certain that your local drives are shared through the Remtoe Desktop Connection. As a backstop, a simple an effective method to move files is to use a web mail program. Create a draft email on your local PC and upload an attachment. The draft can then be accessed on the server through a browser.

Exercises

Exercise 1.7 Setup and save a connection to the server on your computer.

Exercise 1.8 Connect to the server and make sure MATLAB is working.

Exercise 1.9 Use one or both of the methods described to move a file from the local computer to the server and a file from the server to the local computer.

Exercise 1.10 Explore alternative views of the MATLAB desktop by changing the configuration using the Desktop menu.

Lesson 2

Basic Input and Operators

This lesson covers:

- Manually inputting data in scalars, vectors and matrices
- Basic mathematical operations
- Saving and loading data

August 2009 returns (computed using closing prices) for the first 10 trading days for IBM, Microsoft, and Hewlett-Packard are listed in table 2.1.

Task 2.1 (Input scalar data) Create 3 variables, one labeled `IBM`, one labeled `MSFT` and one labeled `HPQ` that contain the August 1 return of the name. For example, to enter the IBM data,

```
>> IBM =  0.0174
IBM =
    0.0174
```

Task 2.2 (Semicolon (;)) Re-enter the data in the previous task but this time use a semicolon (;) to suppress output. Verify that the value is correct by entering the ticker symbol alone on the command prompt (and without a semicolon). For example,

Returns in August 2007

Date	IBM Return	MSFT Return	HPQ Return
3-Aug	0.0174	0.0170	0.0018
4-Aug	-0.0026	-0.0025	0.0009
5-Aug	-0.0094	0.0016	-0.0036
6-Aug	-0.0045	-0.0148	-0.0243
7-Aug	0.0164	0.0042	0.0308
10-Aug	-0.0052	-0.0059	0.0029
11-Aug	-0.0076	-0.0125	-0.0064
12-Aug	0.0126	0.0172	0.0180
13-Aug	0.0024	0.0038	0.0038
14-Aug	-0.0084	0.0029	-0.0058

Table 2.1: IBM, Microsoft and Hewlett-Packard return data for the first 10 trading days of August 2009.

```
>> IBM = 0.0174;
>> IBM
IBM =
    0.0174
```

Task 2.3 (Input a Row Vector) Create row vectors for each of the days in Table 2.1 named `Aug`*XX* where *XX* is the numeric date. For example,

```
>> Aug03 = [0.0174  0.0170  0.0018];
>> Aug04 = [-0.0026  -0.0025  0.0009];
```

Task 2.4 (Input a Column Vector) Create column vectors for each of the ticker symbols in Table 2.1 named `IBM`, `MSFT` and `HPQ`, respectively. For example,

```
>> IBM = [ 0.0174; -0.0026; -0.0094; -0.0045;  0.0164;  -0.0052; -0.0076;  0.0126;  0.0024; -0.0084 ];
```

Task 2.5 (Input a Matrix) Create a matrix named `returns` containing Table 2.1. A matrix is just a column vector containing row vectors. For example,the first two days worth of data are

```
>> returns = [0.0174  0.0170  0.0018; -0.0026  -0.0025  0.0009];
```

Task 2.6 (Construct a Matrix from Row and Column Data) Create a second matrix named `returns_row` from the row vectors previously entered such that the results are identical to `returns`. For example, the first two days worth of data are

```
>> returns_row = [Aug03;Aug04];
```

Create a third matrix named `returns_col` from the 3 column vectors entered such that the results are identical to `returns`

```
>> returns_col = [IBM MSFT HPQ];
```

Verify that all three matrices are identical by entering

```
>> returns_col - returns
>> returns_row - returns
```

and that all elements are 0.

Task 2.7 (Saving Data) Save all data to a file named myfirstmat. Next save only the returns matrix, `returns`, to a file named returnsonly.

```
>> save myfirstmat
>> save returnsonly returns
```

Verify that your data was saved by clearing all variables using

```
>> clear all
```

and then loading the files using the `load` command.

Task 2.8 (Addition and Subtraction) Add the returns of the three series together. Add the returns in `Aug01` to the returns of `IBM`. What happens?

Task 2.9 (Multiplication) Multiply the returns of IBM by 2.

Task 2.10 (Constructing portfolio returns) Set up a vector or portfolio weights

$$w = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$$

Compute the returns to an equally weighted portfolio
Note: Division uses the slash operator (/).

Task 2.11 (Mean, Standard Deviation and Correlation) Using the function `mean`, compute the mean of the three returns series one at a time. For example

```
>> IBMmean = mean(IBM)
```

Next, compute the mean of the matrix of returns using

```
>> retmean= mean(returns)
```

What is the relationship between these two? Repeat this exercise for the standard deviation (`std`). Finally, compute the correlation of the matrix of returns (`corr`).

Task 2.12 (Summing all elements) Compute the sum of the columns of `returns`. How is this related to the mean computed in the previous step?

Task 2.13 (Maximum and Minimum Values) Compute the minimum and maximum values of the columns of returns using the `min` and `max` commands.

Task 2.14 (Rounding Up, Down and to the Closest Integer) Rounding up is handled by `ceil`, rounding down is handled by `floor` and rounding to the closest integer is handled by `round`. Try all of these commands on 100 times returns. For example,

```
>> round(100*returns)
```

Task 2.15 (Element-by-Element Multiplication) Mathematical commands in MATLAB obey the rules of matrix algebra. This is why the portfolio returns could be easily computed as above. MATLAB also supports *element-by-element* operations using the “dot” operations, .* (multiplication) and ./ (division). Multiply the returns of IBM and MSFT together using the dot operator.

Lesson 3

Functions

This lesson covers:

- Calling function with more than one input and output
- Calling functions when some inputs are not used
- Writing a custom function

Begin by clearing all data from the workspace and loading the data in deciles.mat

```
>> clear all
>> load deciles
```

This data set contains 2 years of data on the 10 CRSP decile portfolios from 2006-2007. The variables are named `dec`*XX* where *XX* ranges from 01 (smallest) to 10 (largest). Also included is a variable named `date` that contains the date in MATLAB date format.

Task 3.1 (Calling Functions) Functions were used in the previous lesson. Get used to calling functions by computing the `mean`, `std`, `kurtosis`, `max`, and `mean` of the 10 portfolios in deciles.mat. Also, explore the help available for calling functions using the `help` and `doc` commands. For example,

```
>> help std
>> doc std
```

Task 3.2 (Calling Functions with 2 Outputs) Some useful functions return 2 or more outputs. One example is `sort` which will sort a vector or matrix (column-by-column) and returns the sorted data in the first output and the indices of the elements in the second element. Sort the elements of `dec01` and `dec10`. What is the index of the largest and the smallest elements?

```
>> sort(dec10)
>> [dec10_sorted,dec10_indices] = sort(dec10)
```

Task 3.3 (Calling Functions with 2 Inputs) *Many* functions take two or more inputs. Like outputs, the inputs are simply listed in order separated by commas. Use `linspace` to produce a series of 11 points evenly spaced between 0 and 1. The help for `linspace` is listed below.

```
>> help linspace
 LINSPACE Linearly spaced vector.
    LINSPACE(X1, X2) generates a row vector of 100 linearly
    equally spaced points between X1 and X2.

    LINSPACE(X1, X2, N) generates N points between X1 and X2.
    For N < 2, LINSPACE returns X2.

    Class support for inputs X1,X2:
       float: double, single

    See also logspace, :.

    Reference page in Help browser
       doc linspace
```

Task 3.4 (Calling Functions with Empty Inputs and Outputs) On occasion it is necessary to call a function where some, but not all, inputs are required. One such example is `std`, which takes the data as its first argument, has a flag for a bias correction as its second argument, and takes the dimension as its third. Compute the standard deviation of each decile, then compute the standard deviation across the individual days (Hint: Down is dim 1, across is dim 2).

```
>> help std
 STD Standard deviation.
    For vectors, Y = STD(X) returns the standard deviation.  For matrices,
    Y is a row vector containing the standard deviation of each column.  For
    N-D arrays, STD operates along the first non-singleton dimension of X.

    STD normalizes Y by (N-1), where N is the sample size.  This is the
    sqrt of an unbiased estimator of the variance of the population from
    which X is drawn, as long as X consists of independent, identically
    distributed samples.

    Y = STD(X,1) normalizes by N and produces the square root of the second
    moment of the sample about its mean.  STD(X,0) is the same as STD(X).

    Y = STD(X,FLAG,DIM) takes the standard deviation along the dimension
    DIM of X.  Pass in FLAG==0 to use the default normalization by N-1, or
    1 to use N.

    Example: If X = [4 -2 1
                     9  5 7]
      then std(X,0,1) is [3.5355 4.9497 4.2426] and std(X,0,2) is [3.0 2.0]
    Class support for input X:
       float: double, single

    See also cov, mean, var, median, corrcoef.

    Overloaded methods:
       timeseries/std
       fints/std
       ProbDistUnivParam/std

    Reference page in Help browser
       doc std
```

Task 3.5 (Writing a Custom Function) Custom functions will play an important role later in the course when estimating parameters. Construct a custom function that takes two arguments, μ and σ^2 and computes the likelihood function of a normal random variable

$$f(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Save this function as normal_likelihood.m and compute the likelihood of $x = 0, \mu = 0, \sigma^2 = 1$.

Exercises

Exercise 3.6 (Custom Function) Write a function named `summary_stats` that will take a single input, `x`, and return a matrix with 4 columns and as many rows as there were columns in the original data where the columns contain the mean, standard deviation, skewness and kurtosis of `x`.

Exercise 3.7 (Custom Function) Extend your function in the final task of this lesson to allow for vector values of x, μ, and σ^2 (all the same size). Test your function with

```
>> x=[-3;0;1];
>> mu = [-3;0;2];
>> sigma2=[9 4 1];
```

Lesson 4

Accessing Elements in Matrices

This lesson covers:

- Accessing specific elements in vectors and matrices

Accessing elements in a vector or matrix is a common task. To begin this lesson, clear the workspace, set up a two vectors, `y` and `z`, and a 5×5 matrix `x` using the following commands:

```
>> clear all
>> x=1:25;
>> x=reshape(x,5,5)

x =

     1     6    11    16    21
     2     7    12    17    22
     3     8    13    18    23
     4     9    14    19    24
     5    10    15    20    25
>> y = 1:5

y =

     1     2     3     4     5

>> z = (1:5)'


z =

     1
     2
     3
     4
     5
```

These vectors and matrix will make it easy to determine which elements are selected by a command.

Task 4.1 (Picking an Element out of a Matrix) Select the third element of all three, x, y and z. Next select the 11th element of x. Finally, using double index notation, select the (1,3) and the (3,1) element of x such as in

```
>> x(1,3)
>> x(3,1)
```

Which index is rows and which index is columns? Does MATLAB count across first then down or down first then across?

Task 4.2 (Selecting Entire Rows) Select the 2nd row of x using the colon (:) operator. Select the 2nd column of z then y using the same syntax. What happens?

Task 4.3 (Selecting Entire Columns) Select the 2nd column of x using the colon (:) operator.

Task 4.4 (Selecting Specific Rows or Columns) Select the 2nd and 3rd columns of x using the colon (:) operator. Select the 2nd and 4th rows of x. How can these be combined to select columns 2 and 3 and rows 2 and 4?

Exercises

Exercise 4.5 (Selecting Data by Date) Load the data in deciles.mat and construct a matrix containing all of the vectors. How can all returns on a particular day be selected? How can all returns for a particular decile be selected?

Lesson 5

Program Flow

This lesson covers:

- for loops
- Nested loops

Task 5.1 (Basic For Loops) Construct a for loop to sum the numbers between 1 and N for any N. A for loop that does nothing can be written

```
N = 10
for i=1:N
end
```

Task 5.2 (Compute a compound return) The compound return on a bond that pays interest annually at rate r is given by

$$cr_t = \prod_{i=1}^{T}(1+r) = (1+r)^T$$

Use a for loop compute the total return for £100 invested today for 1,2,. . .,10 years. Store this variable in a 10 by 1 vector `cr`.

Task 5.3 (Simulate a random walk) (Pseudo) Normal random variables can be simulated using the command `randn(`N`,`M`)` where N and M are the dimensions of the desired random numbers. Simulate 100 normals in a 100 by 1 vector and name the result `e`. Initialize a vector `p` containing zeros using the function `zeros`. Add the 1st element of `e` to the first element of `p`. Use a for loop to simulate a process

$$y_i = y_{i-1} + e_i$$

When finished plot the results using

```
>> plot(y)
```

Task 5.4 (Nested Loops) Begin by clearing the workspace and loading deciles.mat. Begin by adding 1 to the returns to produce *gross returns.*[1] Use two loops to loop both across time and across the 10 deciles to

[1] A gross return is the total the value in the current period of £1 invested in the previous period. A net return subtracts the original investment to produce the net gain or loss.

compute the total compound return. For example, if only interested in a single series, this

```
cr=zeros(size(dec01));
gr = 1 + dec01;
cr(1) = 1+dec01(1);
T=10;
for t=2:T
    cr(t)=cr(t-1)*gr(t);
end
```

would compute the cumulated return. When finished, plot the cumulative returns using `plot(cr)`. After finishing this assignment, have a look at `doc cumsum` and `doc cumprod`.

Exercises

Exercise 5.5 Simulate a 1000 by 10 matrix consisting of 10 standard random walks using both nested loops and `cumsum`. Plot the results. If you rerun the code, do the results change? Why?

Lesson 6

Logical Operators

This lesson covers:

- Basic logical operators
- Compound operators
- Mixing logic and loops
- `all` and `any`

Begin by clearing all data and loading the data in deciles.mat

Task 6.1 (Basic Logical Statements) For decile 1 and decile 10, count the number of elements that are < 0, ≥ 0 and exactly equal to 0. Next count the number of times that the returns in decile 5 are greater, in absolute value, that 2 times the standard deviation of the returns in that decile.

Task 6.2 (Compound Statements) Count the number of times that the returns in *both* decile 1 and decile 10 are negative. Next count the number of times that the returns in deciles 1 and 10 are *both* greater, in absolute value, that 2 times their respective standard deviations.

Task 6.3 (Logical Statements and `for` Loops) Use a `for` loop along with an `if` statement to simulate an asymmetric random walk of the form

$$y_i = y_{i-1} + e_i + I_{[e_i<0]} e_i$$

where $I_{[e_i<0]}$ is known as an indicator variable that takes the value 1 if the statement in brackets is true. Plot y.

Task 6.4 (Selecting Elements using Logical Statements) For decile 1 and decile 10, select the elements that are < 0, ≥ 0 and exactly equal to 0. Next select the elements where both deciles are less than 0.

Task 6.5 (Using `find`) Use `find` to select the index of the elements in decile 5 that are negative. Next, use the `find` command in its two output form to determine which elements of the decile return matrix are less than -2%.

Exercises

Exercise 6.6 (`all` and `any`) Use `all` to determine the number of days where all of the decile returns were negative. Use `any` to compute the number of days with at least 1 negative return and with no negative returns (Hint: use negation ($\sim$)).

Lesson 7

Loading Data

This lesson covers:

- Preparing data for import
- Importing data
- Converting dates

Begin by clearing all data from the workspace.

Task 7.1 (Formatting Data in Excel for Import) Format the return data contained in excel.xls for import into MATLAB and save the file as excel_for_import.xls. The Excel file can be downloaded from

`http://www.kevinsheppard.com/wiki/MFE_MATLAB_Introduction`.

Task 7.2 (Importing Data) Import the file created in the previous step and save the data to excel_imported.mat.

Task 7.3 (Converting Dates) Convert the dates imported in the previous step using `x2mdate`.

Exercises

Exercise 7.4 (Practice) Getting data into and out of MATLAB is very important for your success in the computing portion of the course. Practice on the file excel_practice.xls which is available on the website,

`http://www.kevinsheppard.com/wiki/MFE_MATLAB_Introduction`.

Lesson 8

Graphics

This lesson covers:

- Basic plotting
- Editing plots
- Subplots
- Histograms

Begin by clearing all data from the workspace and loading the data in `hf.mat`. This data set contains high-frequency price for IBM and MSFT on a single day and times in MATLAB format.

Task 8.1 (Basic Plotting) Plot the series labeled `IBMprice` which contains the price of IBM. Add a title and label the axes. Use the interactive tool to add markers and remove the line.

Task 8.2 (Subplot) Create a 2 by 1 subplot with the price of IBM in the top subplot and the price of MSFT in the bottom subplot.

Task 8.3 (Plot with Dates) Plot the price of IBM against the series `IBMdate`. Use `datetick` to reformat the x-axis.

Task 8.4 (Histogram) Produce a histogram of MSFT returns (Hint: you have to produce the Microsoft returns first).