Matlab

MSc Finance

June 2014

Mathieu ZARADZKI

Lecture 1
A quick tour

OUTLINE

What is Matlab?

- spreadsheet table is like linear algebra
- not suitable for any kind of information but for a large majority
- Matlab brings the two together

What can we use Matlab for?

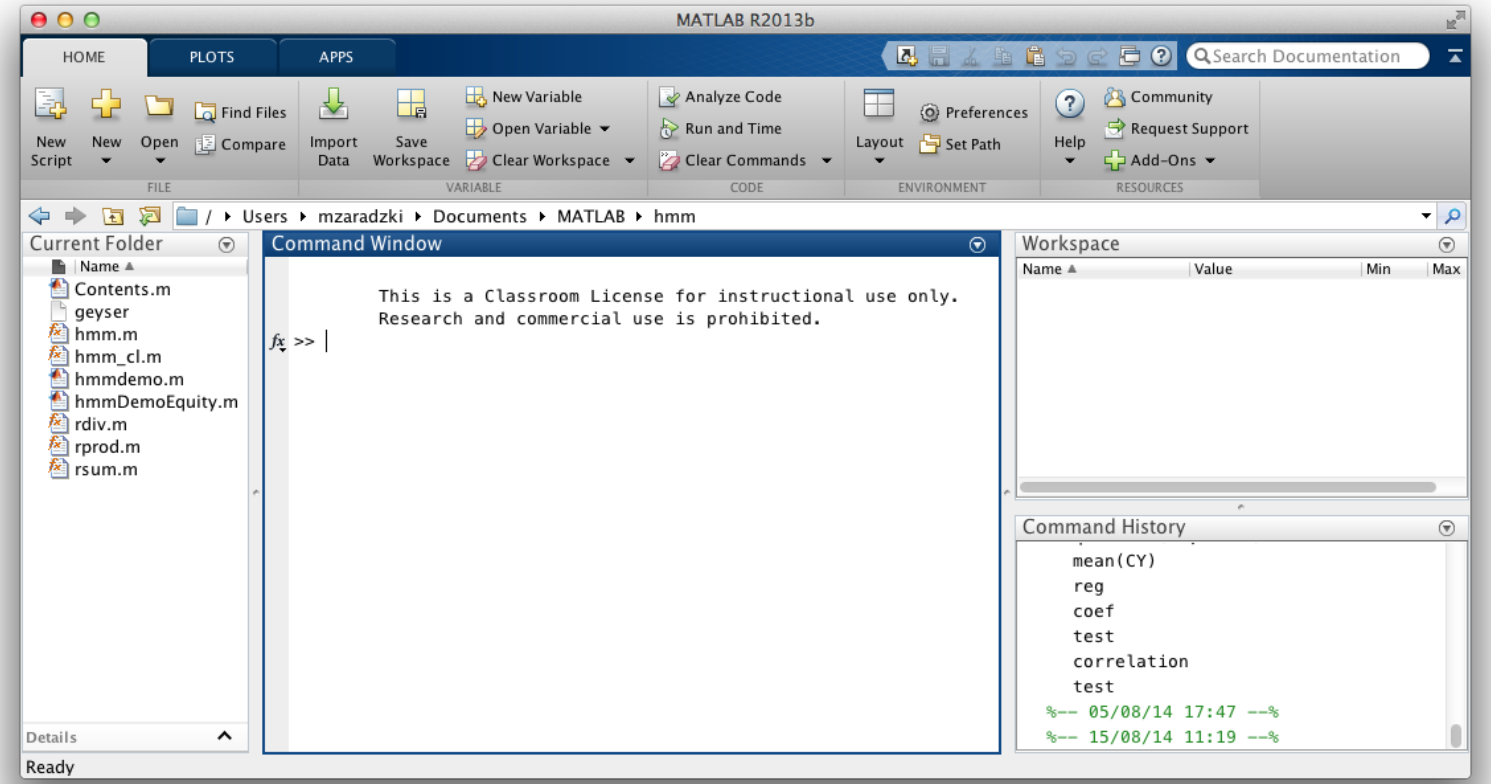
- much more productive than spreadsheet
- less error / operational risk thanks to visibility
  - think of the Reinhart & Rogoff's error
  - or even FT review of Piketty's best-seller book
- easier to re-use and to share

Main applications in finance

- econometrics
- optimization such as Portfolio construction
- simulations such as Risk analysis
- or (almost) anything you want ...
  - the limit being usage scaling and computation speed
    - so no live/production application such as HF trading
    - but any research investigation
    - or even prototyping before going full scale e.g. Barclays Live

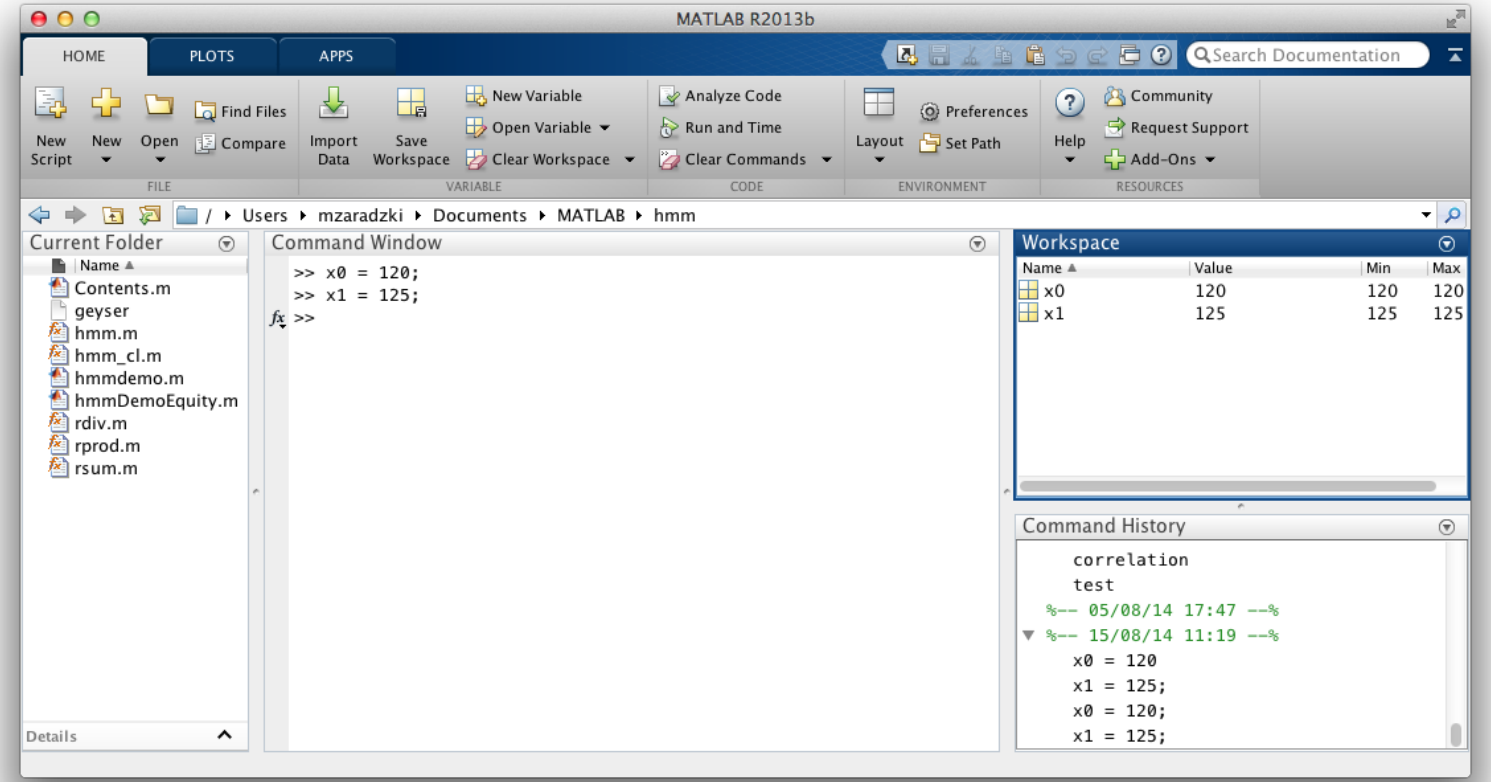
Environment overview: Command Window

- the Command Window (or shell) to run calculations on the flow



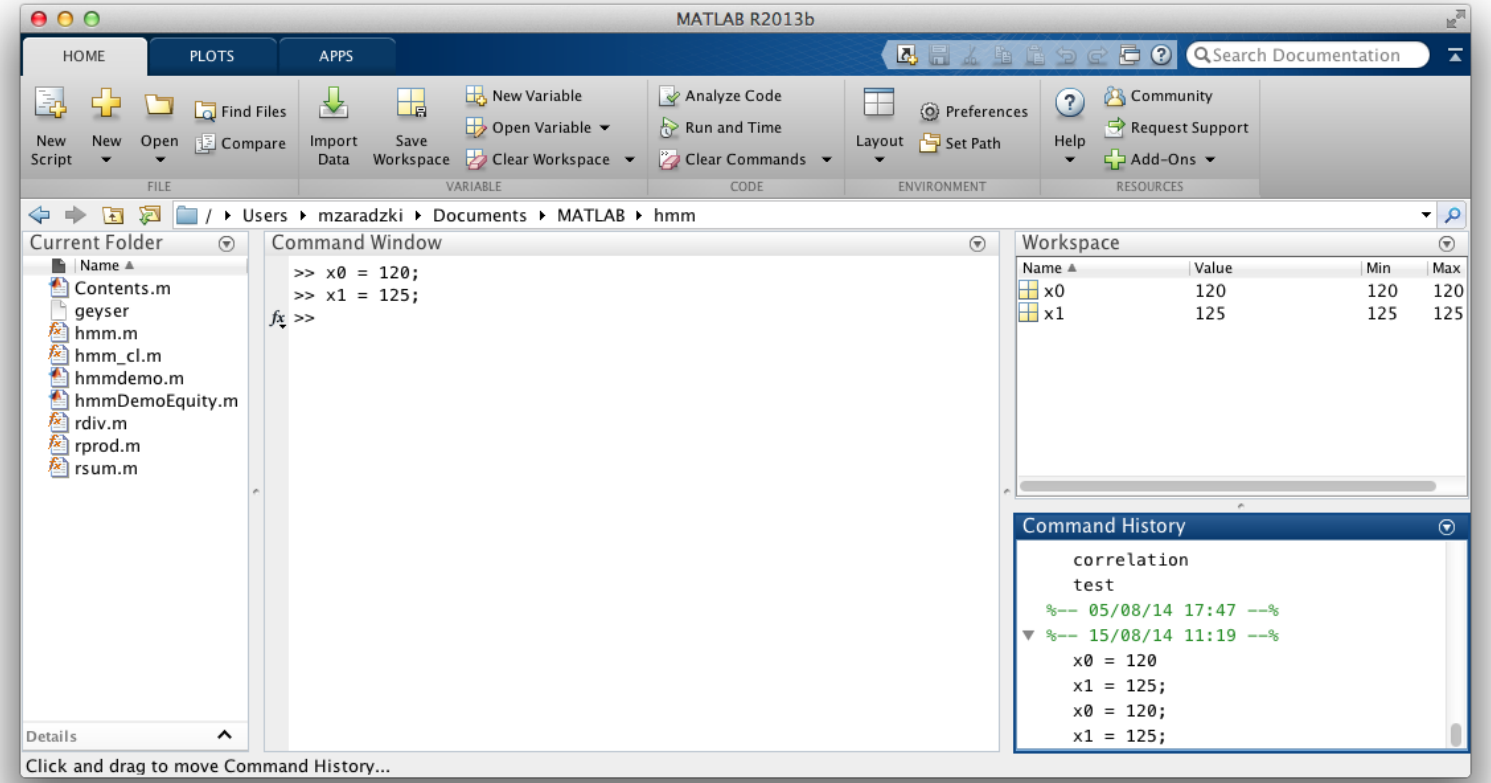
Environment overview: Workspace

- the Workspace to see existing variables and data storage



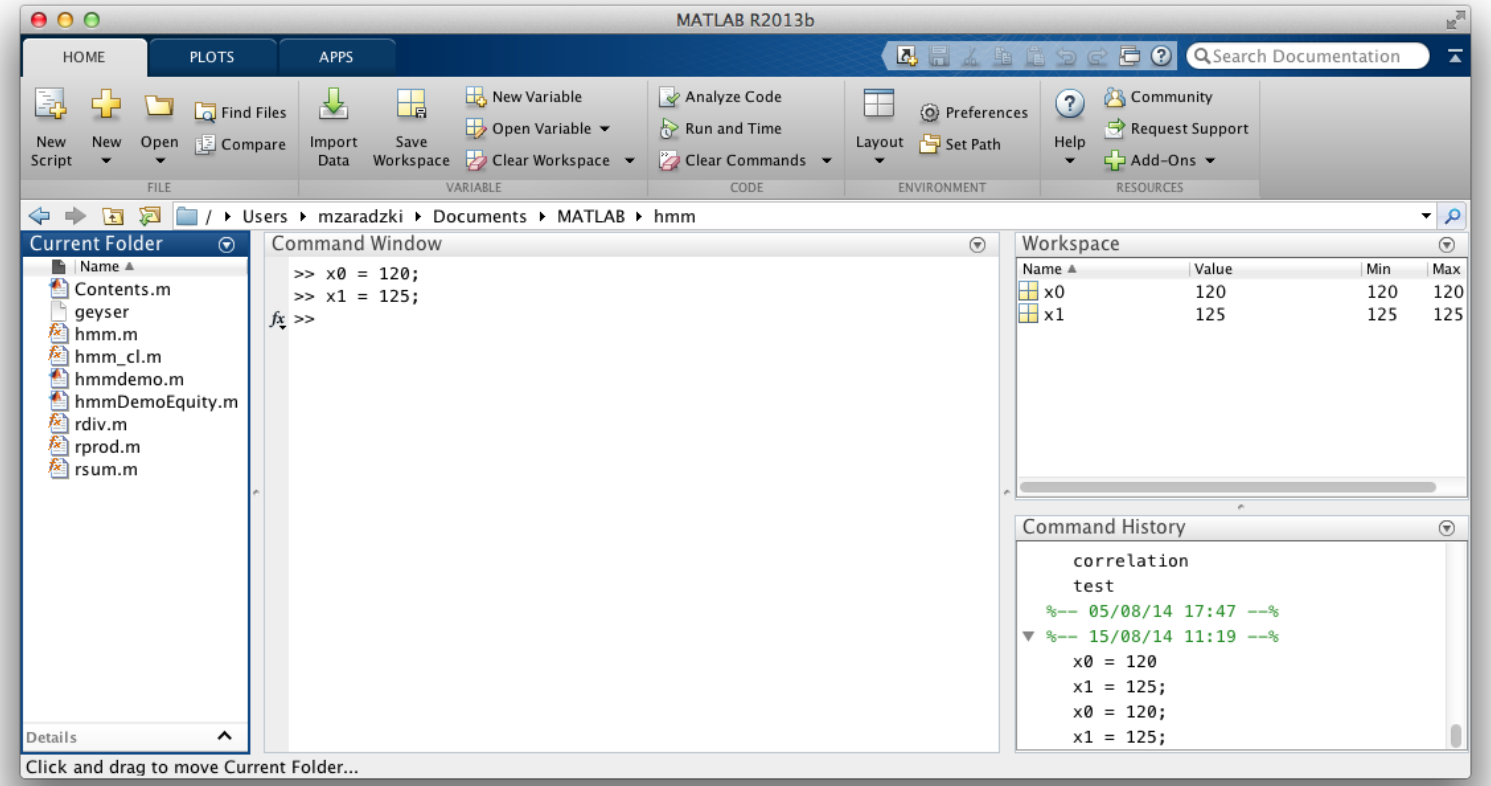
Environment overview: Command History

- the Command History to quickly review the command sequence or to re-run a command

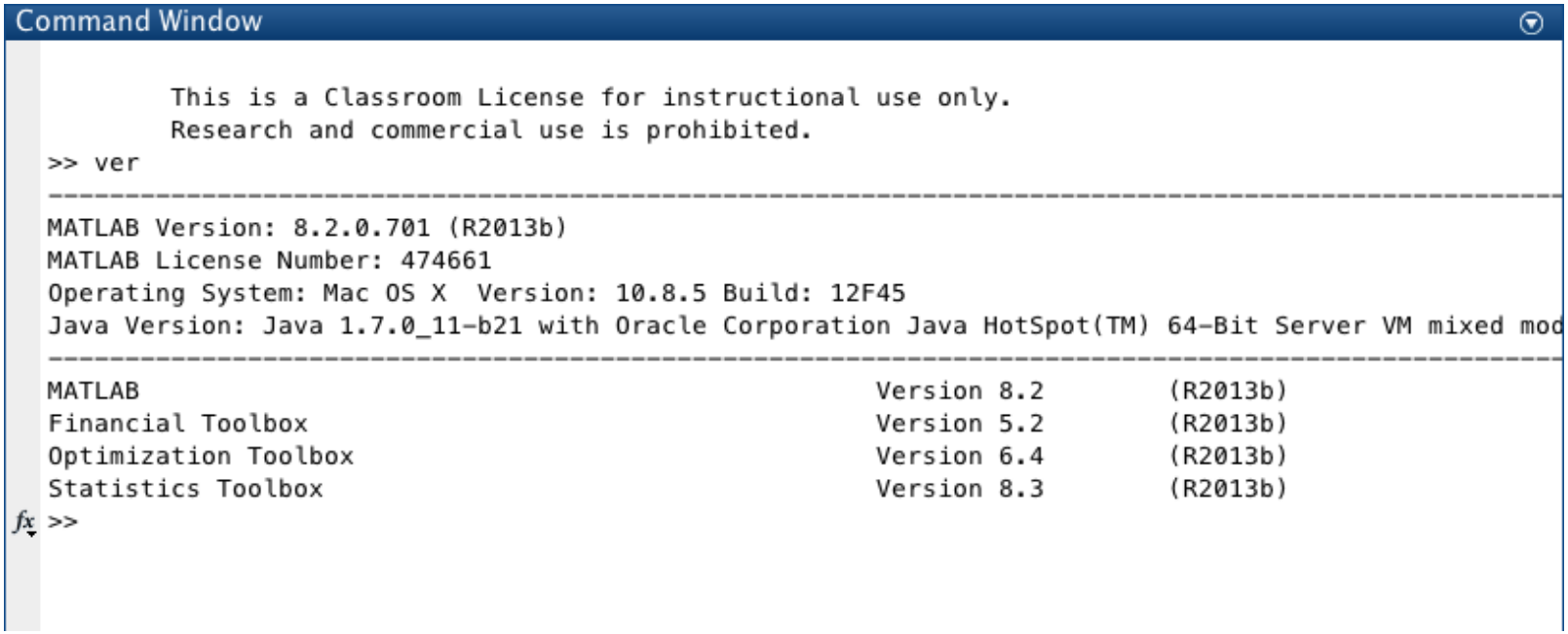


Environment overview: Current Folder

- the Folder browser allows you to change your working folder to load/save data and calculations



Toolbox addins



You need help?

- just ask for it
  - *help dataset* for quick reminder
  - *Search Documentation* widget (top/right of the UI) for full information
- look on the community web site at http://www.mathworks.fr/matlabcentral/
- look on *StackOverFlow* web site (ma favourite)

Typing instructions

- simply type in the Command Window and press return
- note that with ";" at the end of a line the output will not show
- to fix a typo (or any other bug) just key the up arrow to alter pas commands

Mathematical functions

- stantard maths functions are (obviously) available with intuitive notations

```
>> log (2.5) % Neperian
>> exp(1.3)
>> sqrt(4)
>> sin( pi ) % Radian base
```

- exponentiation as an operator or as a function

```
>> 2^3
>> pow(2, 3)
```

Creating and naming variables

- the best way to understand what you (or your colleagues) did ...

```
>> x0 = 120 % setting x0 value
>> x1 = 125
>> dx = x1 - x0 % creating a variable from other variables
```

- dont have to clutter your window with array-like variables

```
>> manyintegers = [1 : 10000];
>> sum(manyintegers)
```

- variable names are case sensitive and thus X0 is not x0
- dont re-use existing names !!!

```
>> which dx
```

Saving variables (optional)

- Matlab clears out your variables from the workspace when you exit
- HOWEVER you can save them using .mat files for later use

```
>> x0 = 120 % setting x0 value
>> x1 = 125
>> save( 'C:/WORK' , 'x0' , 'x1' ) % note the optional folder path
>> clear % deletes all current variables
>> load('C:/WORK', 'x0', 'x1')
```

Vectors

- vectors and matrices are arrays that can contain only one datatype: numeric, character or logical.
- they can have 1, 2 ... or even 3 dimensions
- vector and matrix elements are enclosed in square brackets [ ]
  - the elements of a row vector are separated by commas (or white spaces)
  - the elements of a column are separated by semicolumns

```
>> somerow = [ 2 , -3 , 5 ]
>> somecolumn = [ 2 ; -3 ; 5 ]
>> thesamerow = [ 2 -3 5 ] % here we simply used white-spaces
```

Vector - shift and lag

```
>> somevector = 1 : 3 : 100;
>> somevector( end - 5 : end ) % the 5 last elements
```

Vector - functions

- Matlabs likes so much vectors you can do "anything" you need with them
- you need the cosine of a vector ?
  - does not make any sense to a math teacher
  - but Matlab is pragmatic about it

```
>> cos(somevector)
```

Matrices

- to define a matrix you must use both commas and semicolumns

```
>> somematrix = [ 1 , 2 , 3 ; 1 , 4 , 9 ; 1 , 8 , 27 ]
```

- as you would expect ...
  - all rows must have the same length
  - all columns must have the same length

Matrix - size

- the "size" function provides you with the number of rows and the number of columns of a matrix

```
>> [ nbrows, nbcols ] = size( somematrix )
nbrows = 3
nbcols = 3
```

- the "numel" function provides you with the numbe of elements that is the product of dimensions

Matrix - element

- elements are accessed by specifying row-column indices or linear indices

```
>> somematrix( 2 , 3 )
```

- matrices are read column after column

```
>> somematrix( 6 )
```

- WARNING in Matlab indices starts at 1 and not at 0 like with many other languages

Matrix - submatrix

```
>> somematrix( : , 2 ) % extract the 2nd column
>> somematrix( end - 2 : end , : ) % extract the 3 last rows
```

Matrix - usual suspects

```
>> zeros( 3 , 3 ) % easy to guess what this is
>> ones( 3 , 3 )% easy to guess what this is
>> eye( 3 ) % the identity matrix
```

Matrix - algebra

- your usual sums and differences
- your usual products
- and the element-wise product denoted .*

Matrix - algebra (bis)

```
>> det( somematrix ) % easy to guess what this is
>> inv( somematrix ) % easy to guess what this is
>> somematrix' % the TRANSPOSED matrix
```

Matrix - collation

- you can append a matrix on the RHS of another one

```
>> [ somematrix , 2*somematrix ] % horizontal concatenation
```

- you can append a matrix at the bottom of another one

```
>> [ somematrix ; 2*somematrix ] % vertical concatenation
```

Matrix - mutation (?)

- stack the elements of a matrix in one column vector and transpose it

```
>> matrixasonerow = somematrix( : )'
```

- recover the initial matrix by reshaping the vector as a 3-by-3 matrix

```
>> reshape( matrixasonerow , 3 , 3 ) % back in the initial shape
```

From algebra to matryoshka

- vectors and matrices are very powerfull ways of manipulating data
- but they "constrain" you in two ways
  - you can only store elements of the same type (e.g. only numbers, only text strings)
  - all rows (or columns) must have the same dimension (square, cube, ...)
- Matlab provides you a way around this if you want to store an heterogeneous data set

```
>> prof = struct ;
>> prof.name = 'mz' ;
>> prof.teaching = 'all stars' ;
>> prof.promotions = [2013, 2014] ;
```

Matlab cuisine

- Matlab IS like algebra
  - has vectors and all
  - is actually very good (fast) at it
- Matlab IS NOT like algebra
  - can mix different type of variables
  - allows for "loose" but "sensible" calculations such as exp(vector)

Lecture 2
Customization

OUTLINE

Iteration statements

- computers are fantastic students
  - you show them once and they remember for ever
- computers are fantastic workers
  - they don't get bored of doing things again and again and again
- the "for loops" are perfect ways to specify repeating tasks

```
>> efact = 2;
>> for en = 2 : 2 : 16
     efact = efact * en;
end
>> efact
```

- you can also do "while loops" but they are dangerous so lets skip that

Conditional statements

- computer are very good soldier they don't question instructions unless they are instructed to do so
- sometimes a task will requires some conditions to be met
  - e.g. "buy a IBM stocks IF it is below a price target"

```
>> res = 2;
>> for n = 2 : 16
    if (mod(n, 2) == 0)
        res = res * n;
    else
        res = res + 1;
    end
end
>> efact
```

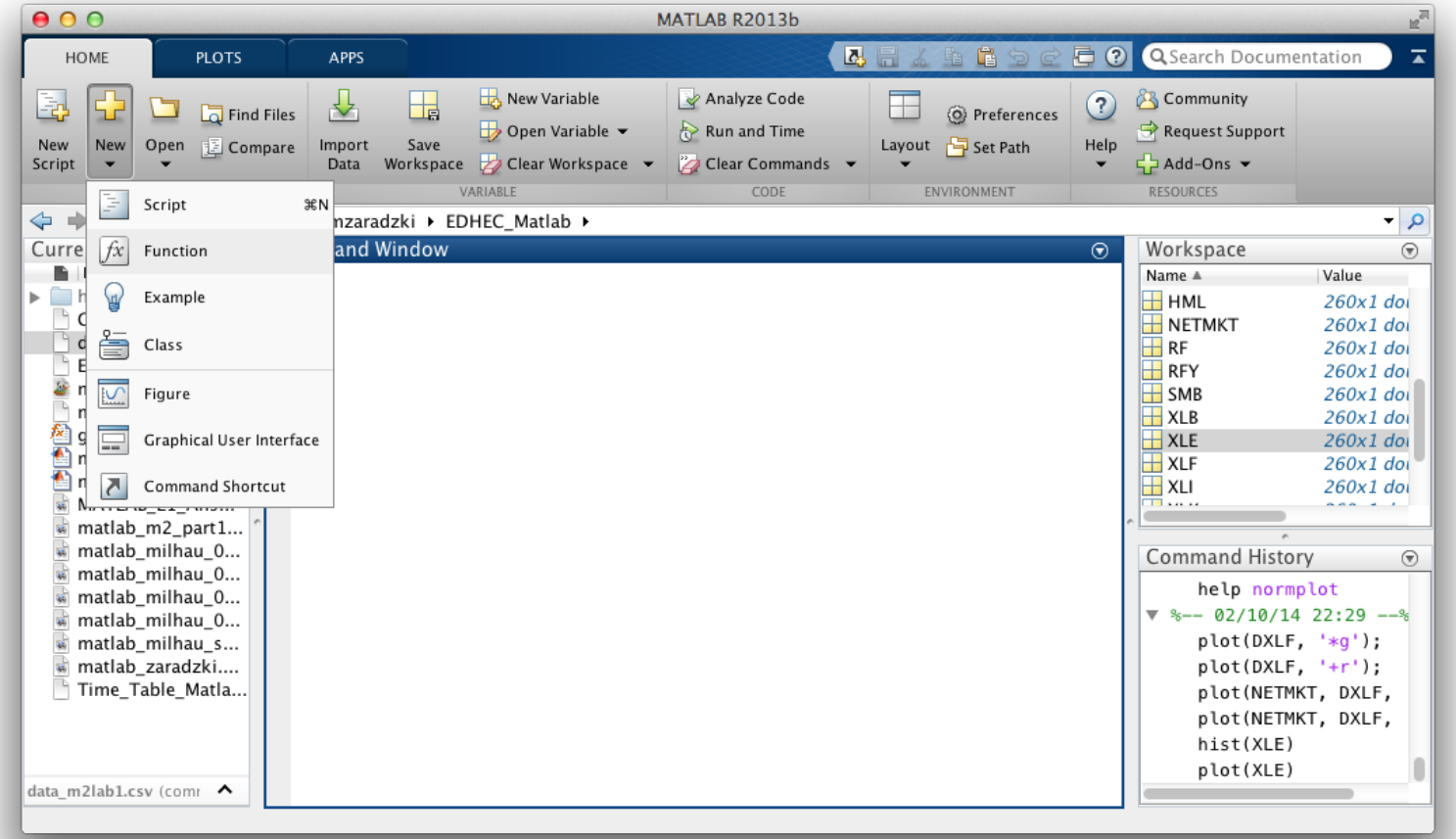
What is an m-file?

- and m-file contains a sequence of valid matlab instructions
  - you can have calculations e.g. exp(-2)
  - you can have comments e.g. % this is a dull comment
  - you can have for-loops and if-tests
  - you can have function declaration
- for example the first class questions were comments in an m-file
- you could have written your matlab code in it

Why use an m-file?

- the best way to save your work is to write it in a file
- it will then be easy to "correct a mistake"
- it allows to re-use your excellent Matlab work for later work
- it makes it easy to share it with colleagues
- ....
- ... I hope you are convinced about m-file usefulness
- ... because from now on I only want to see m-file on your PC's

Create a blank m-file



Saving your m-file

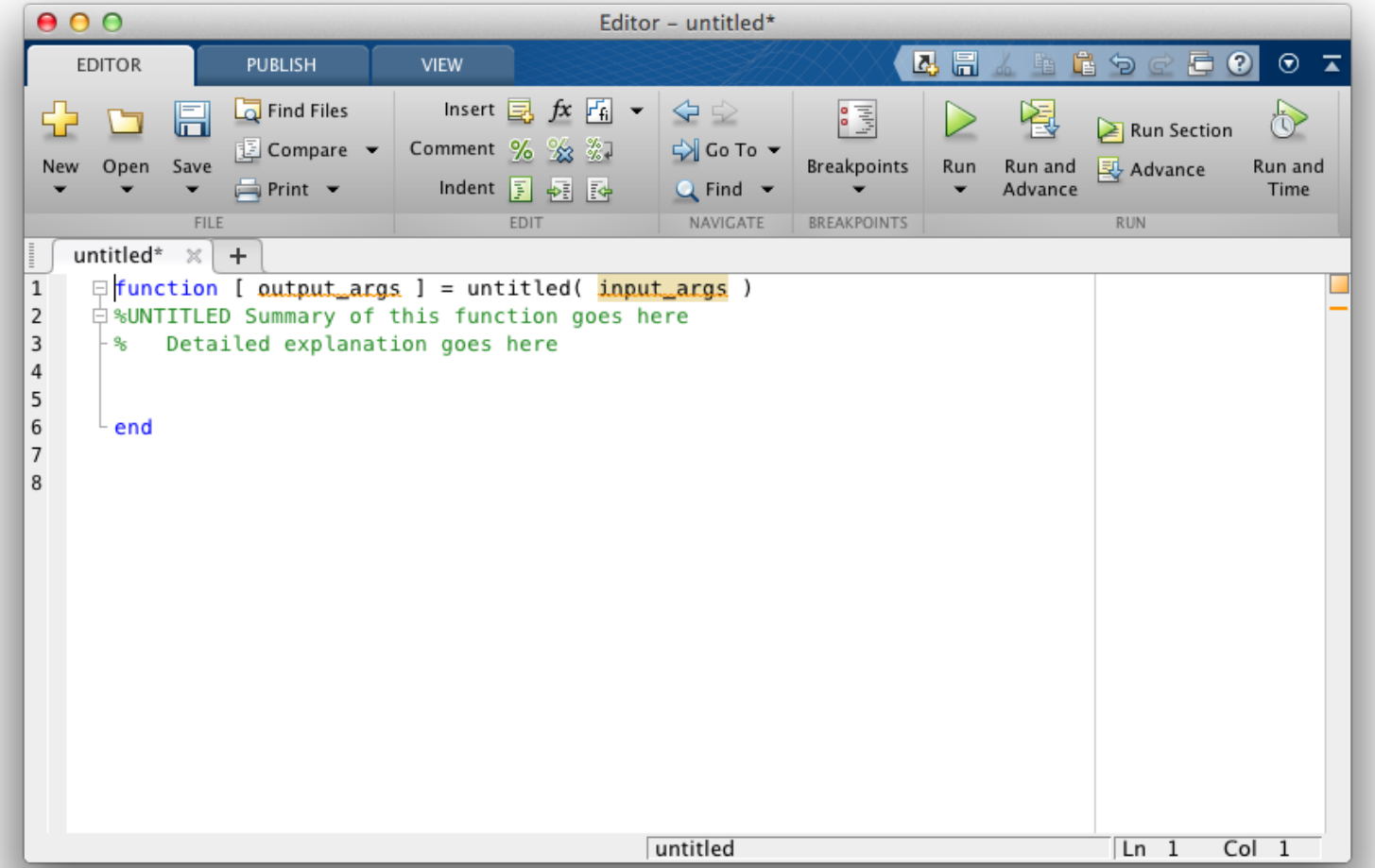
- why this slides?
- saving a file is obvious ...
- ... thinking of doing it but everything crashes is less obvious
- you have been warned, save your work regularly

Matlab functions

- you remember you can easily "erase" a Matlab function ?
  - FYI between the two groups 5 of you did it last time
- it is because Matlab functions do not have special status
- they are just m-files like the function we will write today
- you can even open their file to learn how it works
- or even to modify Matlab ...
  - ... but don't do it during my class

Your own functions

- Matlab saves you a bit of time by adding the necessary function "place holders" to your m-file if you want



Calling a function

- as Matlab does not make a disctinction between its own ready-to-use functions and yours you can use your function directly in the Command panel OR in other m-files
- the only difference between you and Matlab is that Matlab will never delete your work

Date representation

- for Matlab dates are just integer numbers

```
>> today
>> 735885 % 13-Oct-2014
```

Date formats

- obviously it is possible to display dates in human readable form too

```
>> datestr(today)
>> 13-Oct-2014
```

- and depending on where that human being comes from Matlab can accomodate

```
>> datestr(today, 'dd/mm/yy') % european
>> datestr(today, 'mm/dd/yy') % USA
```

Date utility functions

- day of week

```
>> weekday( today ) % 1=sunday, 7=saturday
```

- last day of month

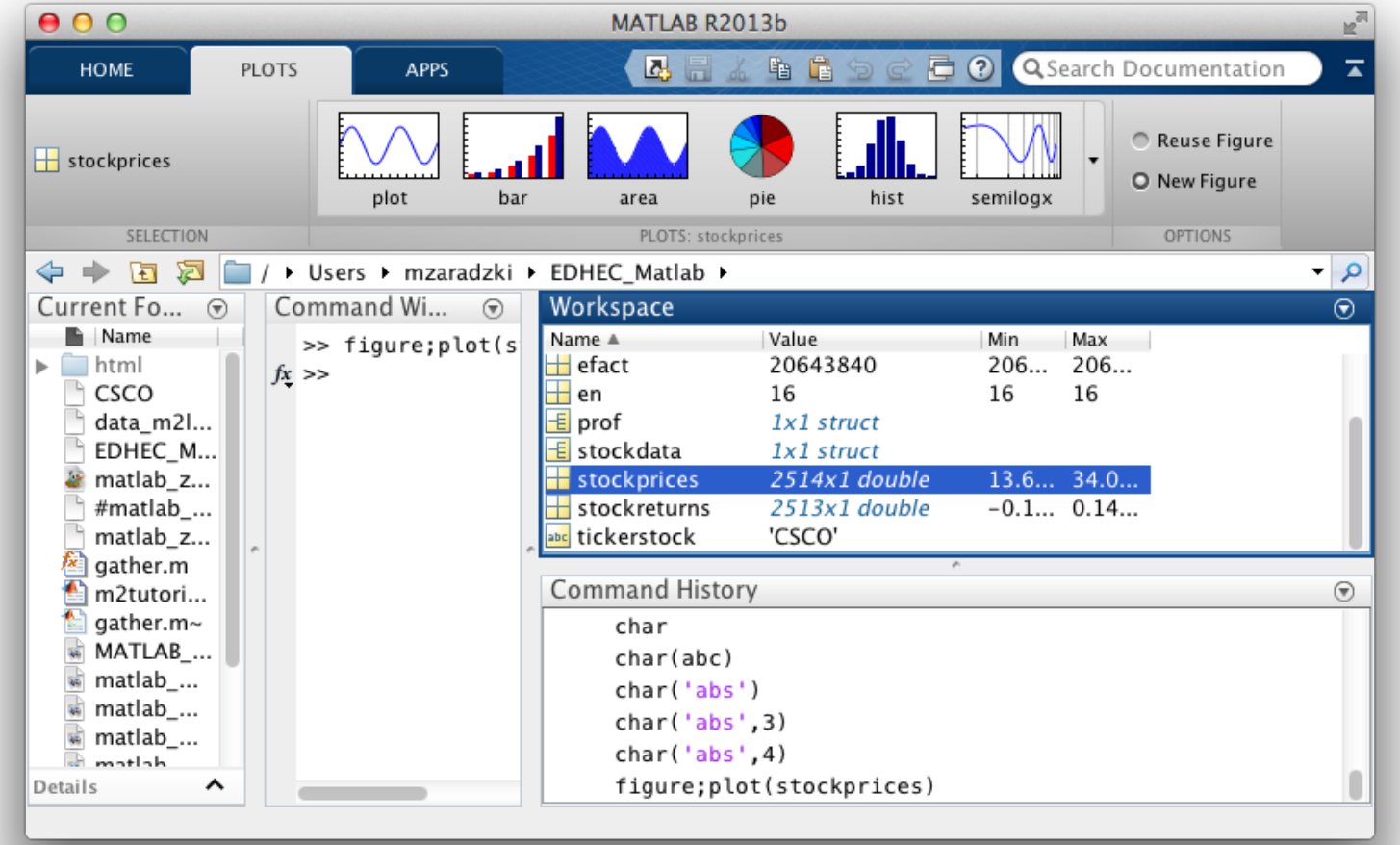
```
>> eom(2014, 11 )
```

Warning and errors

- on some very rare occasions your Matlab instructions will not work
- IF and WHEN that (EVER) happens you should
  - read carefully Matlab message to understand the problems
  - not lose patience if Matlab keeps complaining after your changes
    - an error may be followed by other ones
  - if you don't make sense of Matlab message split your task
    - ever heard of "divide and conquer" ? I think an engineer came up with it

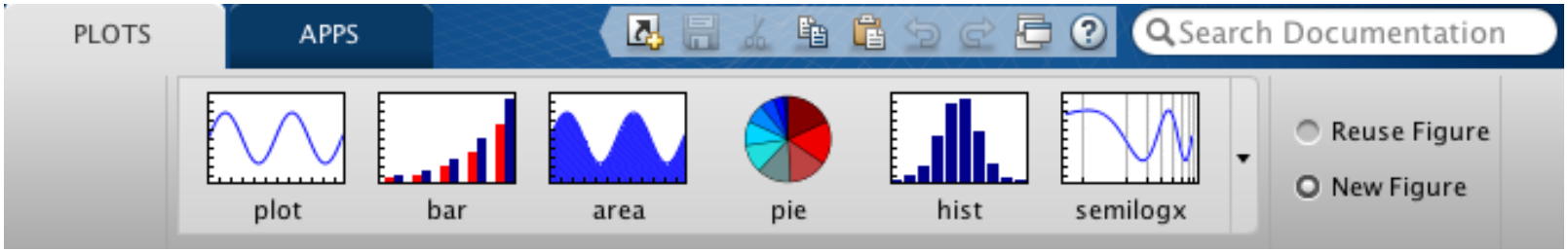
Creating

- one way is to select the data you wan to chart in the Workspace
  - don't try to chart an number, it only works if it makes sense to do it



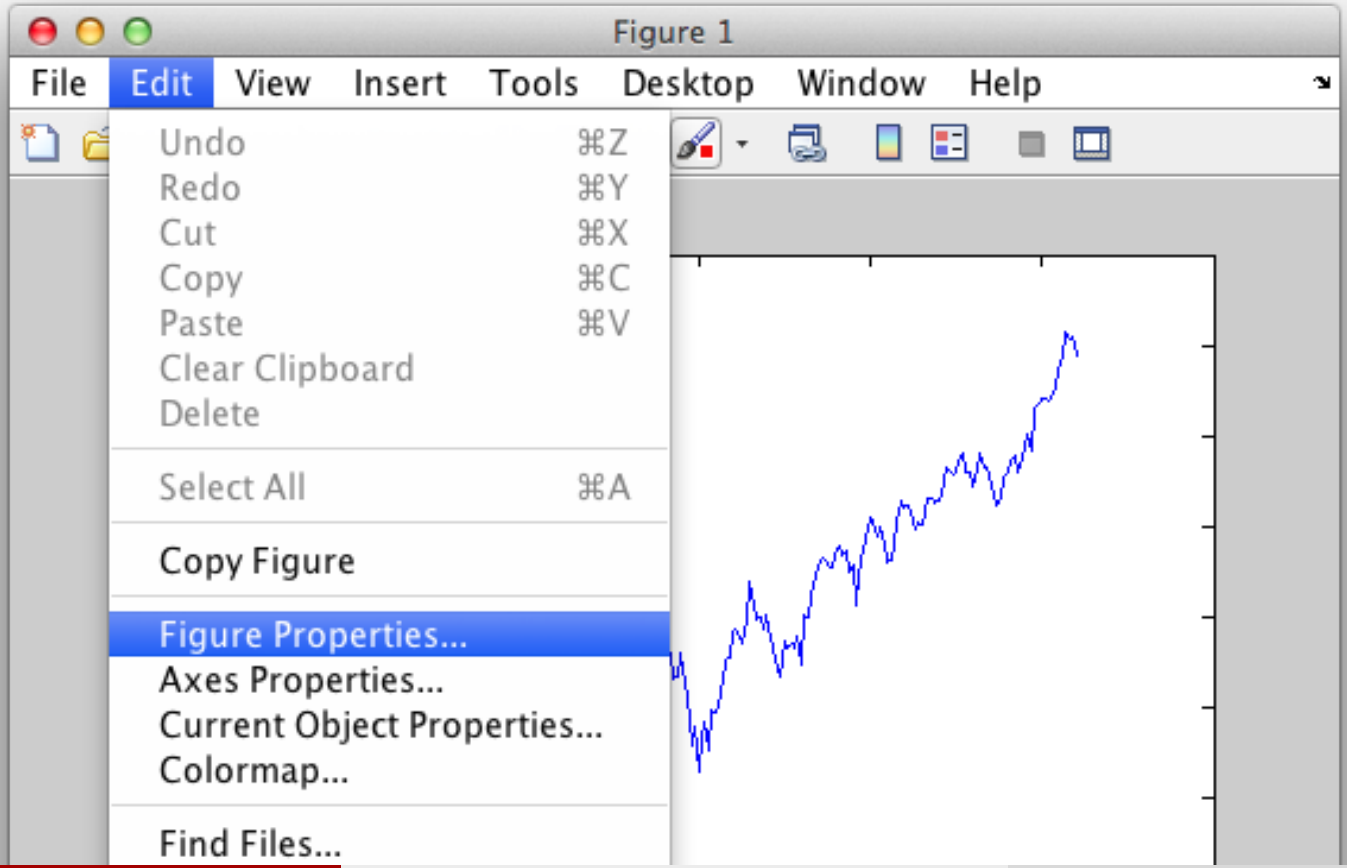
Types

- Matlab obviously offers you any type of chart you are used to see in Excel
  - timeseries, bars, bubles, pies
  - but don't go overboard with it as it gets time consuming
  - simpler is often better



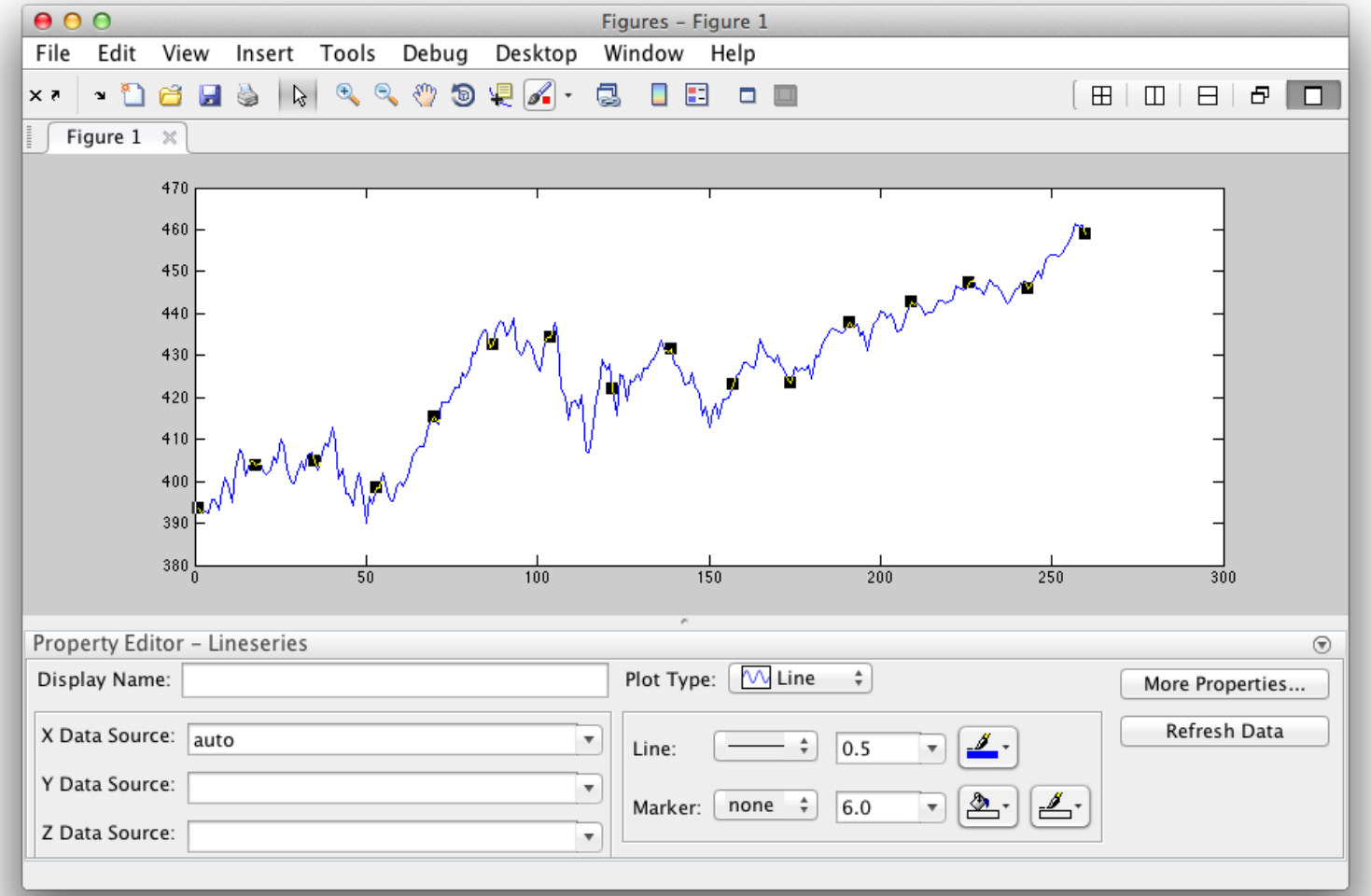
Customization

- from the figure window follow these
  - Edit
  - Figure Properties



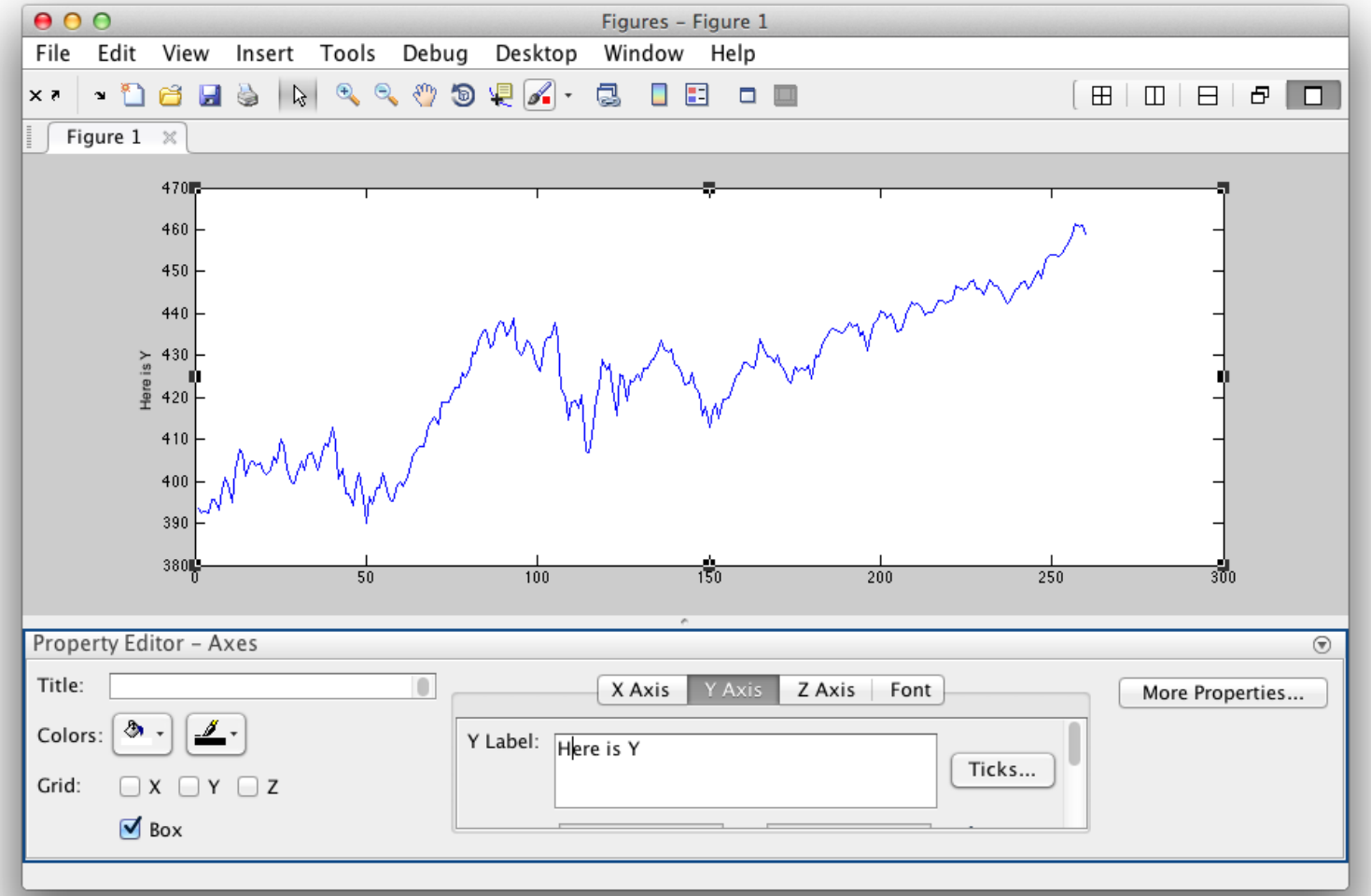
Custom line

- simply click on the line to see the required options



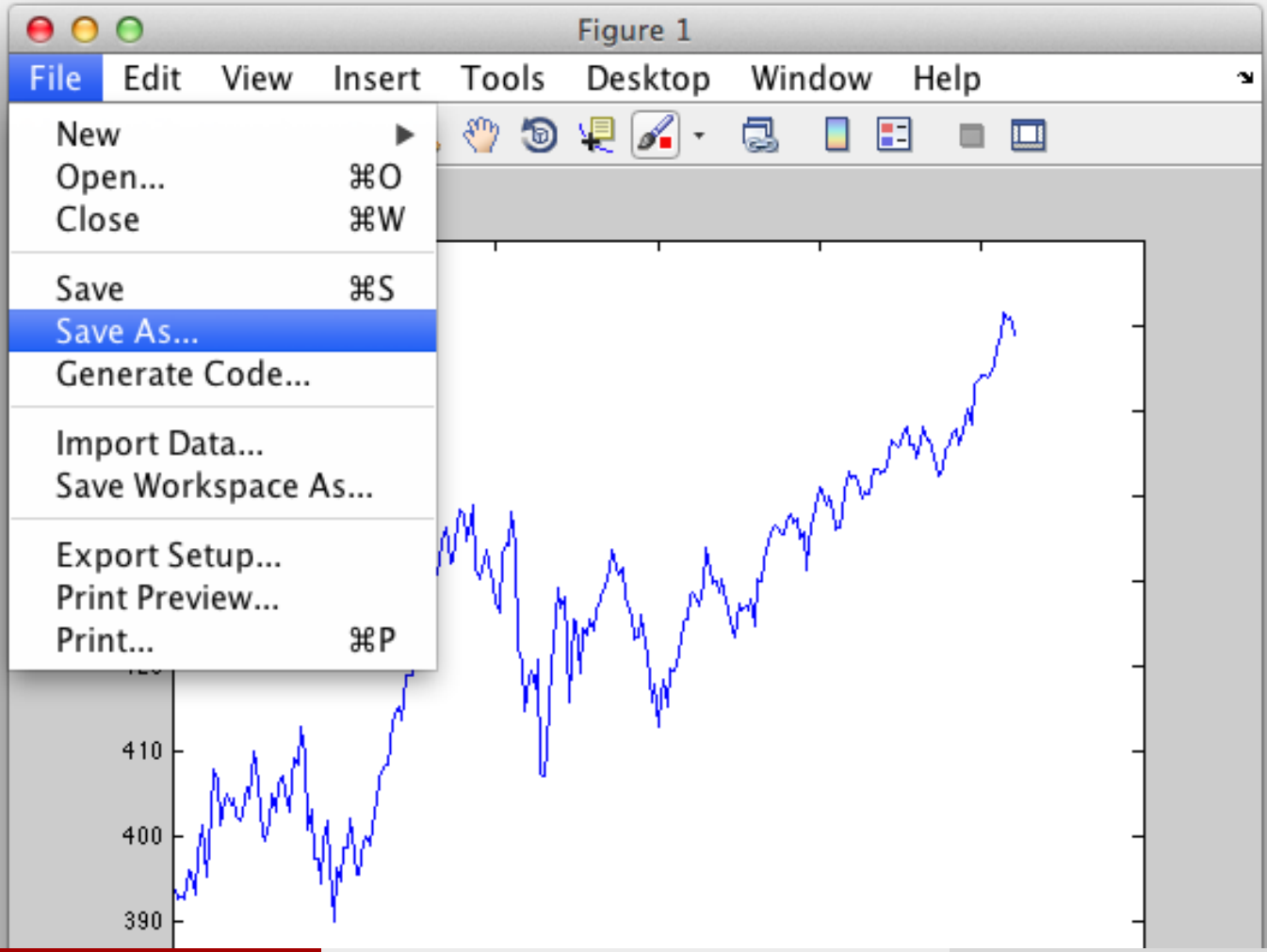
Custom x and y axis

- simply click on the axis to see the required options



Saving charts

- simply click on the "file" menu



Command line charts

- when you have long reports to do the GUI is not good enough
- use your programming skills for the charts too

```
>> plot(DXLF, '*g'); % g for green
>> plot(DXLI, '+r'); % r for red
>> plot(NETMKT, DXLF, '*b'); % a blue scatter plot
```

- commands for fancy charts?
  - look closely on the previous chart their is a "Generate Code" option

Lecture 3
Matlab goodies - 1

OUTLINE

Motivation

- rational implies optimal
- finance is about allocation, decision
  - from the entrepreneur deciding between Kapital and Labour
  - to Markowitz, Sharpe and Black Litterman

Examples

- one dimensional search

```
>> x = fminsearch(@(x)sin(x^2), x0); % one dimension
```

- multi-dimensional search

```
>> banana = @(x)100*(x(2)-x(1)^2)^2+(1-x(1))^2; % that is really the function name
>> [x,fval] = fminsearch(banana,[-1.2, 1])
```

- as usual just to "help fminsearch" in Matlab for more information

Why random computation?

- isn't the purpose of computer to be all predictible?
- history provides us with some patterns, such as correlation
- but not every possible event occured
  - what about combination of events?
  - what about "tail" event?
- that is why we need random models
- and most models cannot be computed with formula, thus the need for random simultations

Random variable

- a vector of random UNIFORM variables

```
>> r = a + (b-a) * rand(100,1);
```

- a vector of random GAUSSIAN variables

```
>> r = gmean + gstd * randn(100,1);
```

Lecture 4
Matlab goodies - 2

OUTLINE

Descriptive stats (part 1/2)

- the main statistics

```
>> mockdata = -3 + 2 * randn(1,100) + rand(1,100); % we blend a
Normal and a Uniform distribution
>> mean( mockdata )
>> var( mockdata )
>> std( mockdata )
>> median( mockdata )
>> range( mockdata ) % the high-low spread
```

Descriptive stats (part 2/2)

- more advanced statistics

```
>> skewness( mockdata ) % the THIRD moment
>> kurtosis( mockdata ) % the FOURTH moment
```

- key levels

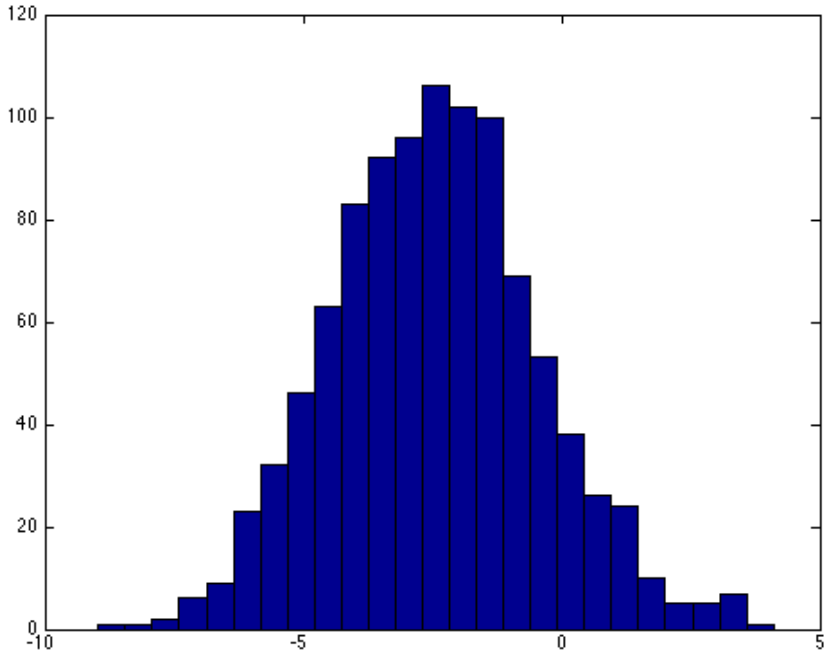
```
>> prctile( mockdata, 10 )
>> prctile( mockdata, 90 )
```

- N(0,1) like equivalent levels

```
>> mockdata = -4 + 3 * randn(1,100);
>> zscore( mockdata )
```

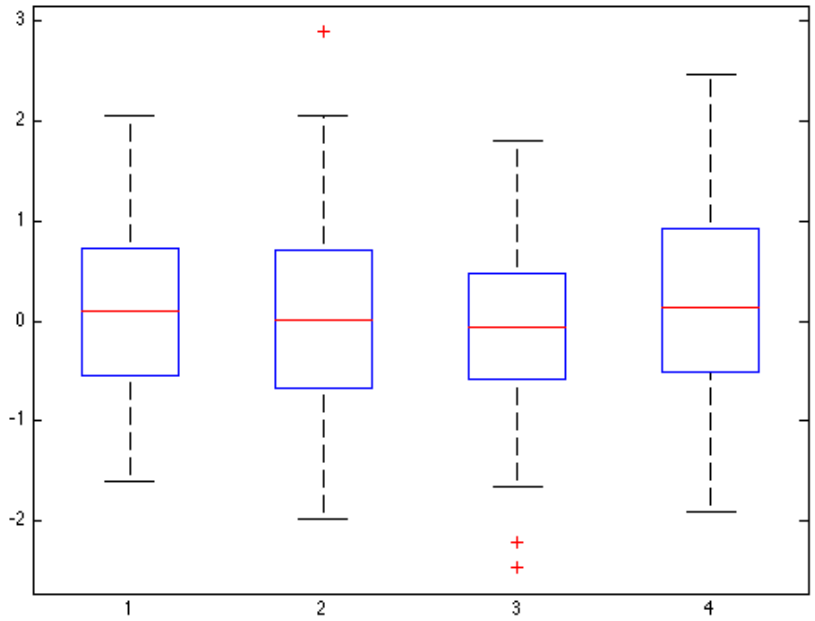
Histograms

```
>> mockdata = -3 + 2 * randn(1,1000) + rand(1,1000); % same as previous but more points
>> hist( mockdata, 25 ); % 25 represents the number of chart BARs
```



Box plot

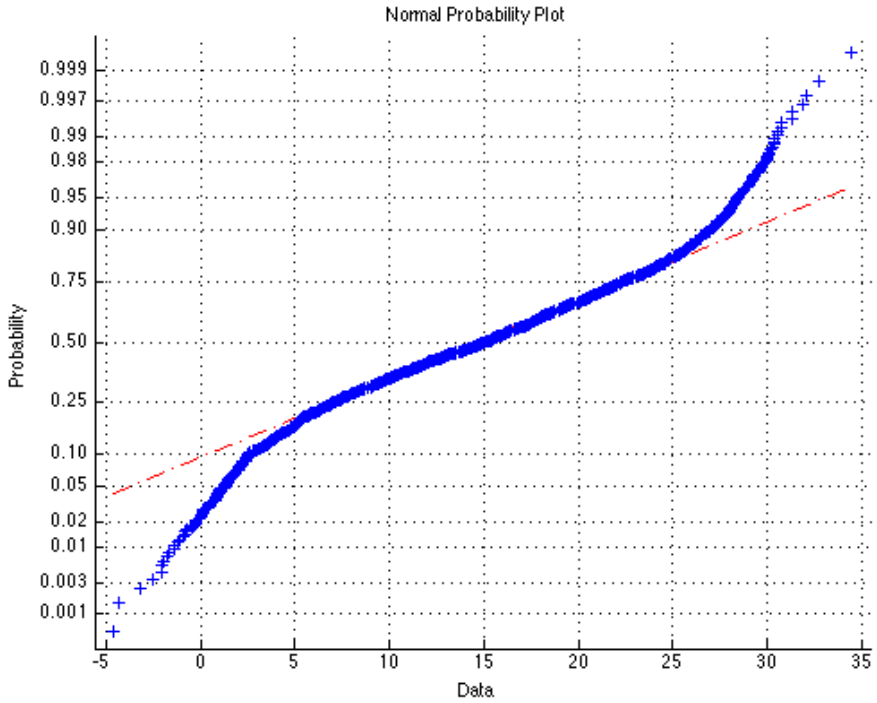
```
>> mockdata = randn(50, 4); % 4 columns of 50 randn each
>> boxplot( mockdata ) % also called the mustache plots
```



NormPlot

- NormPlot are the special case QQ-Plot for the Normal distribution

```
>> strangedata = 2 * randn(1,1000) + 30 * rand(1,1000);
>> normplot( strangedata ) % also called Quantile-Quantile Plot
```



Lecture 5
Industrialization

OUTLINE

Why random computation?

- history provides us with some patterns, such as correlation
- but not every possible event occured
  - what about combination of events?
  - what about "tail" event?
- that is why we need random models
- and most models cannot be computed with formula, thus the need for random simultations

How to emulate randomness?

Deterministic randomness

Random variable

- a vector of random UNIFORM variables

```
>> r = a + (b-a) * rand(100,1);
```

- a vector of random GAUSSIAN variables

```
>> r = gmean + gstd * randn(100,1);
```

Stochastic processes

Brownian motion

- we know Gaussian law is everywhere
- so we need a process equivalent of it
  - a process whose increments between two "dates" are gaussian!

Correlated processes