Basic FAME instructions

Structures

Comments
Any line proceeded by a double hyphen "- -" is a comment. It can also follow a command.

Commands
These do the main tasks. The format is always
Command command_arguments
Command arguments are often separated by commas if multiple arguments are allowed

For example:

1) graph two variables:

graph e_eu, e_rc

2) open an existing database to read and write to it:

open "ss.db" as con

Options
Options generally provide details for commands, and frequently associated with formatting. Every command has a set of default options. The options can be changed either universally for all commands following it:

Option option_arguments

Or just for a specific command, by including it in <> in the structure of the command itself:

command <option option_arguments> command_arguments

For example, if you want to report a variable to two decimal places you can type either:

decimal 2
report e_eu, e_rc

Or:

report <decimal 2> e_eu, e_rc

For the purposes of graphs and reports there is a standard set of options that we normally use:

abort on
ignore on
pause off
length full
show vertical
object precision
decimal 2

What These Options Do:

abort on - the program will crash if there is a mistake in the FAME code
ignore on - any mathematical operation on the data will proceed, even if there are missing data points.
length full – Uses the full page for graphs and reports.
show vertical – Data is shown in columns (better for long time series).
pause off – When reporting to the screen, the output will not pause with a "hit any key" command to start a new page.
object precision – in calculations, all data uses as many decimal places as possible.
decimal 2 – when displaying the data, all data is shown up to two decimal places (even if those places are each zero).

Functions
Of course, there are also a whole array of mathematical functions, that are of the form *function(series)*

So if we have a time series *e_eu*, to type its average value we would use:
type ave(e_eu)

There are also a number of useful string functions that are good for a variety of purposes:
display name(e_eu)
displays the name of the series, *e_eu*

display string(e_eu [2002])+" is the exchange rate in 2008"
would display

```
0.6223 is the exchange rate in 2008
```

Files
FAME commands can be typed in by hand, but it is more useful to use files to run a bunch of commands. There are input files (such as **graphs.inp** in GIMF, which runs some reports) which are just groups of commands.

There are also procedures, which are files which are called, usually with extra arguments supplied. For example, to run graphs and tables in GIMF we use the procedure file **graphs.pro**.

First we compile it in fame using the compile command:
compile graphs.pro

And then we load the resulting FAME-readable file:
load graphs.pc

Graphs.pro contains many procedures which we can call by name. For example, we create steady state reports in 2001 using the control database with the procedure call:

$reportss "ss.db" "2001"

Accessing data directly

Use the open command, with the access option. You can either [r]ead, [up]date or [c]reate a database. If you create a database, it assumes the database does not yet exist:

open <acc r> "ss.db" as con
open <acc up> "ss.db" as con
open <acc create> "ss.db" as con

You could also create it simply by typing
create ss.db

The word "con" is what is known as the **database alias.** You can open up to fifty databases simultaneously. Provided you are only reading the database, you can open a single database more than once.

FAME always has a database available and ready to write to. Its database alias is **work**

Reporting data directly

You can report a series from the open databases by either typing, displaying or reporting.

type e_eu
display e_eu
report e_eu

To define the date range, use the option *date*, once you have defined the *frequency* of the data with the option frequency. Dates can expressed as [a]nnual, [q]uarterly, [m]onthly, [d]aily, or [b]usiness.

frequency a
date 2000 to 2003

This will define the data as annual data, and you will report it from 2000 until 2003. Although GIMF can be either an annual or a quarterly model, we use the annual frequency for data, because it is more convenient when building complex tables and graphs.

So here is the output of the three commands:

deci 4

type e_eu

0.6223

display e_eu

```
E_EU

2000     0.6223
2001     0.6223
2002     0.6223
2003     0.6223
```

report eu

```
     E_EU
     ----

2000       0.6223
2001       0.6223
2002       0.6223
2003       0.6223
```

Notice that the *type* command only shows the first observation of the series, without mentioning the period.

The report command above is in its unstructured form. The report command can also be used to built more complicated tables. For example, let's report e_eu, first as a horizontal table, and then as a vertical table. To do so, we open a report with the report command, and then define the columns with the select command and the rows with the print command.

```
report
  select 2000 as "00"
  select 2001 as "01"
  print value e_eu
end report
```

```
    00       01
------   ------

0.6223   0.6223
```

```
report
  select e_eu
  print date
end report
```

```
              E_EU
              ----

2000        0.6223
2001        0.6223
2002        0.6223
2003        0.6223
```

Graphing the data

We can also graph the data using the graph command. Assume we have specified the date range:

```
date 2000 to 2020
```

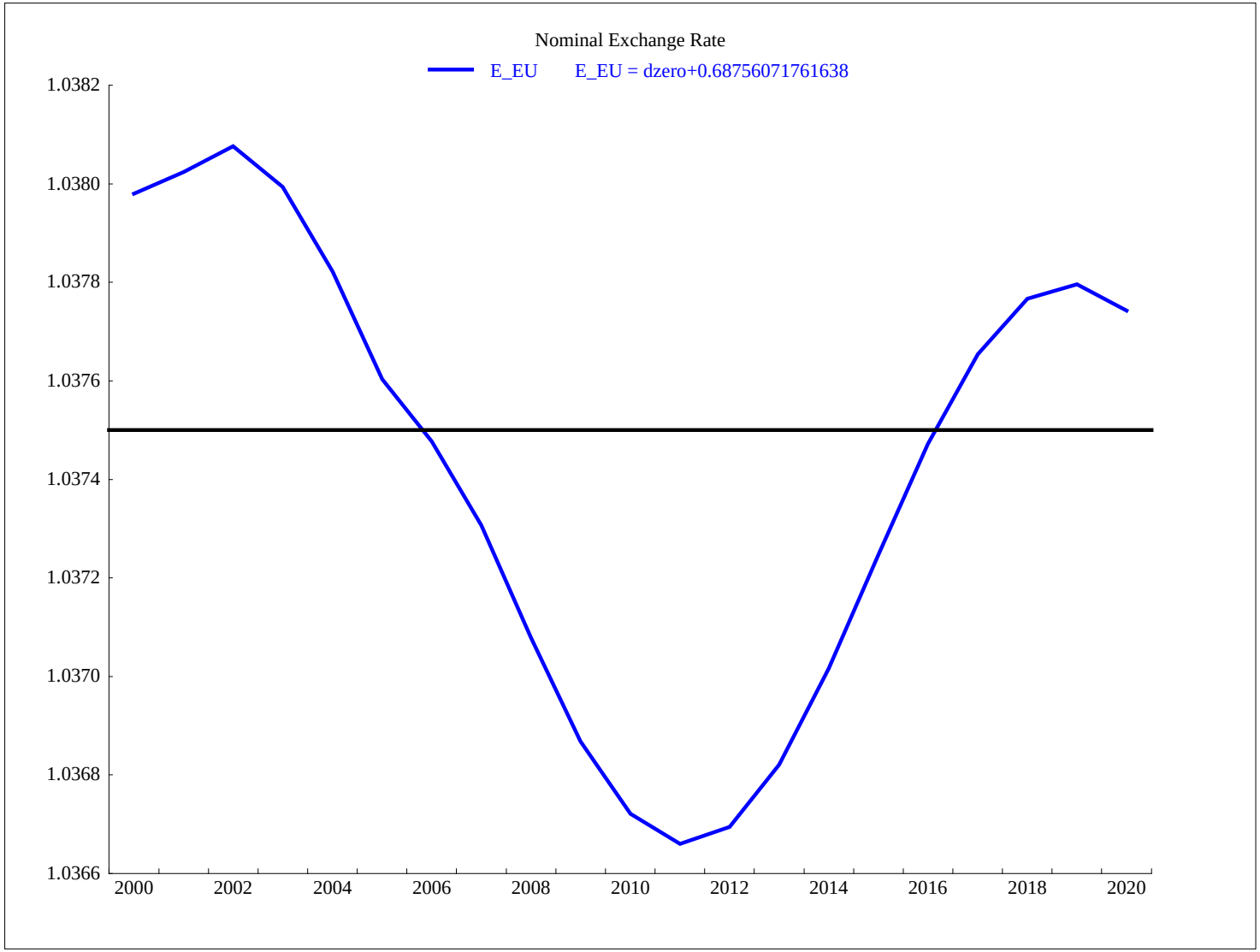
There is the simple graph:

```
graph e_eu
```

Or the structured graph:

```
graph
  title #1 text "Exchange Rate", size small
  data e_eu
  draw marking <depict axis left> at 1.037
end graph
```

The structured graph allows us to enter extra commands. In this case, we are adding a title to the graph, plus drawing a line at 0.7 on the graph.


Nominal Exchange Rate
E_EU
E_EU = dzero+0.68756071761638
1.0382
1.0380
1.0378
1.0376
1.0374
1.0372
1.0370
1.0368
1.0366
2000
2002
2004
2006
2008
2010
2012
2014
2016
2018
2020