1 Graphical User Interfaces (GUIs)

1.1 Handles

When you are drawing things in Matlab, you need to have *some* way to control the looks of your *Window* (e.g. the Backgroundcolor, if a Menu is shown, etc.), the thickness of the *Lines* you draw, the font of your *Text* labels, etc. The way this is commonly done in Matlab (and also in other graphical environments) is through ***"Handles"***.

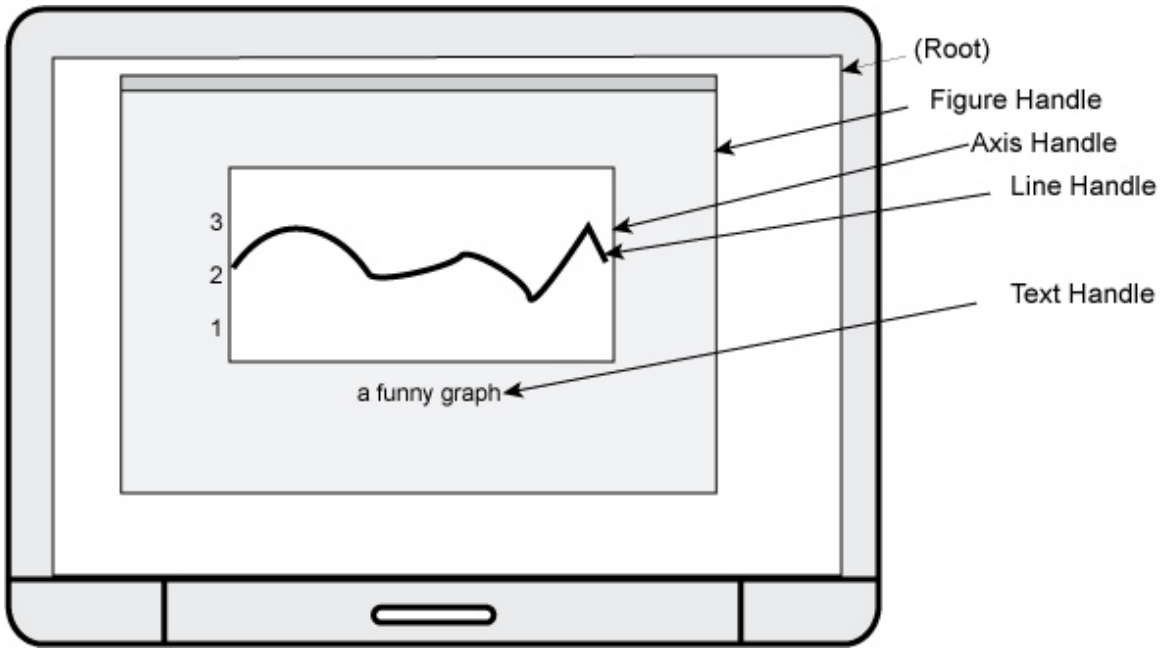

Figure 1 Organization of *Handles* in a simple figure.

Figures

For example, the *figure handle* controls parameters like the

- Location of the window on the screen.
- Background color
- Title of the Window
- If it is visible
- …

The following commands are helpful for working with the figure handle:

Cur_FigHandle = gcf	Gets the current figure handle
get(gcf)	Gets the properties of the *current figure*
set(gcf)	Shows you the options of the properties that you can set for your *current figure*
Cur_Position = get(gcf, 'Position')	Get the position of the *current figure*
set(gcf, 'Position', [1 1 720 150])	Set the new position of the *current figure*
set(gcf, 'Tag', 'MyFigure')	Set the *Tag* property of the *current figure*
findobj('Tag', 'MyFigure')	Find something (here the figure with the *Tag* "MyFigure")
Set(gcf, 'UserData', [MyXData; MyYData]	Set the *UserData* property of the *current figure*

Note that most of these commands are not specific for the *figure handle*. For example, you can aalso use the command "findobj" to find a *line handle*. Note that you can have as many *figures* as you want. The *current figure* is the one on which you draw when you type for example "plot(1:10)".

Axes

Similarly, the command

```
gca
```

provides the *handle* for the *current axis*.

Note that there is a hierarchy:

- Each *figure* can have many *axes*, and each *axis* belongs to exactly one *figure*.
- Each *axis* can have many *lines*, and many *text objects*

Lines

For example, the command

```
lineHandle = plot(1:10);
set(lineHandle, 'LineWidth', 2, …
                'Color', [1 0 0]);
```

draws a thick, red line.

1.2 Callbacks

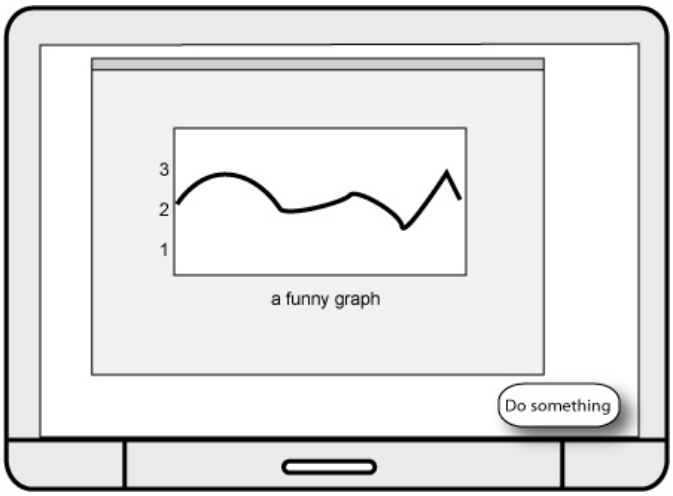

Figure 2 A *Callback* is a request for performing a certain task or action.

When you have a button on your *figure*, you must tell MATLAB what it should do when that button is pressed. This is done through *callbacks*. *Callbacks* are nothing else but functions that get executed when you do *something*. Note that this "something" can be

- The push of a button
- Moving a slider
- Clicking the mouse
- Typing a key
- Closing the window
- ….

Note that this is the epitomy of "object-oriented programming"!

You see that there are many such callbacks – and in general you just don't want to deal with them. To make is simpler for you to nevertheless make simple graphical applications, MATLAB provides the program GUIDE.

1.3 Tips for the User-Interface

- Who is using the program? Is it you (or some other *expert* in the field)? Or is it someone who has no experience in the application?
- Is this program used on a daily basis? In that case it should be optimized for efficiency.
- Or is it just once-in-a while? In that case the interface should be as self-explanatory as possible.

1.4 Layout of Scientific Graphs

Try to make your printouts such that when you look at the printout in one year from now, you will still know what is shown there.

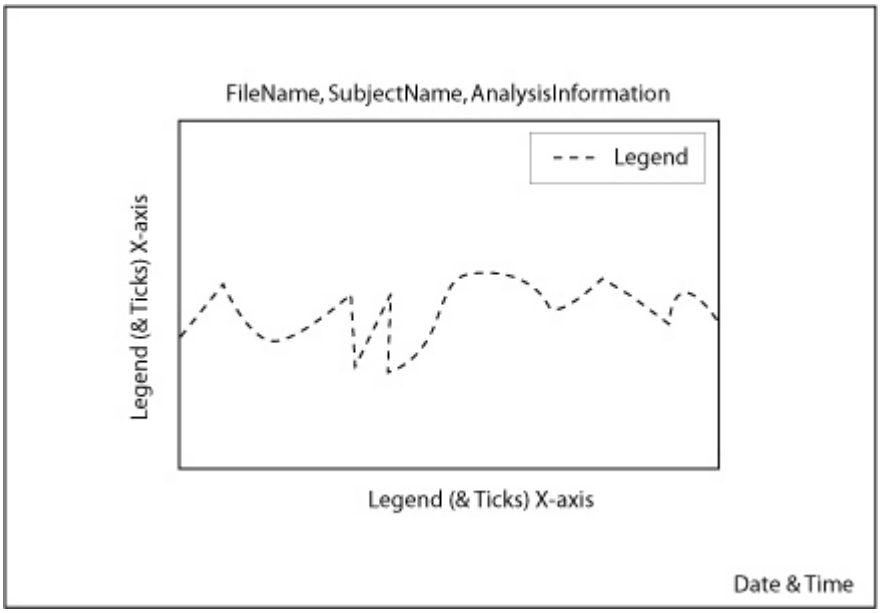

Figure 3 Graphs should always contain enough information to be understood on their own, without any external additional information.