PHYS-4007/5007: Computational Physics

Using IDL in Command Line Mode

1 Editing a New IDL Procedure File

There are two ways to run IDL under Linux: (1) the IDL Workbench Graphic User Interface (GUI) and (2) the IDL Command Line mode. This document will instruct you in this second method. First, log into your Linux user account on the TAF machines in Brown Hall 264 and open a Terminal window. Change directory to your idl subdirectory (*i.e.*, cd idl). Next issue the emacs command at the Linux prompt with the file name of the file you wish to create (*e.g.*, emacs mycode.pro &), the & sign places the emacs session in background mode so you can get the Linux prompt back.

Unlike Fortran 77, IDL is a *free-field* programming language – commands can start in any column on a line. Enter the following code in your editor:

```
PRO MYCODE
;
; This is a test code using the IDL Command Line utility.
;
X = FINDGEN(1000)
Y = SIN(X/10)
Y2 = Y * EXP(-X/300)
;
PLOT, X, Y
;
RETURN
END
```

Now from the emacs GUI, save this file with the name "mycode.pro" (do not include the double quotes in this name).

2 Starting IDL in Command Line Mode

To start IDL in Command Line mode, just issue the "idl" command (minus the double quotes) from the Linux prompt. You will now see the 'IDL>' prompt instead of the Linux

prompt. Now compile your code to make sure there are no errors:

```
IDL> .run mycode
```

(the period '.' in front of 'run' is required). This should produce the following output:

```
% Compiled module: MYCODE.
```

If however IDL reports errors, go back to the emacs editor and fix your errors. Now run your procedure with

```
IDL> mycode
```

You should now see a new window appear with a sine-wave plot in it.

3 Improving the Appearance of Your Plot

Add the following 'keywords' to your PLOT command, , XTITLE='!6X', YTITLE='!6SIN(X/10)' such that your code now becomes

```
PRO MYCODE
;
; This is a test code using the IDL Command Line utility.
;
X = FINDGEN(1000)
Y = SIN(X/10)
Y2 = Y * EXP(-X/300)
;
PLOT, X, Y, XTITLE='!6X', YTITLE='!6SIN(X/10)'
;
RETURN
END
```

Recompile and rerun the code:

```
IDL> .run mycode
IDL> mycode
```

You will now see axis labels on your plot. We will next increase the font size of your labels with the following commands inserted prior to the PLOT command: !X.CHARSIZE = 1.6 & !Y.CHARSIZE = 1.6 and !P.POSITION = [0.16, 0.16, 0.9, 0.9] such that your code now reads:

```
PRO MYCODE
;
; This is a test code using the IDL Command Line utility.
;
X = FINDGEN(1000)
Y = SIN(X/10)
Y2 = Y * EXP(-X/300)
;
!X.CHARSIZE = 1.6  &  !Y.CHARSIZE = 1.6
!P.POSITION = [0.16, 0.16, 0.9, 0.9]
;
PLOT, X, Y, XTITLE='!6X', YTITLE='!6SIN(X/10)'
;
RETURN
END
```

Note that the !P.POSITION system variable sets the borders of the plot window in fraction of the plotting space in the form [XMIN, YMIN, XMAX, YMAX]. This was needed to allow enough room for the axis labels to fully appear on the plot. We will next add a main title to the plot and set its font size with !P.CHARSIZE.

```
PRO MYCODE
;
; This is a test code using the IDL Command Line utility.
;
X = FINDGEN(1000)
Y = SIN(X/10)
Y2 = Y * EXP(-X/300)
;
!P.CHARSIZE = 1.6  &  !X.CHARSIZE = 1.6  &  !Y.CHARSIZE = 1.6
!P.POSITION = [0.16, 0.16, 0.9, 0.9]
;
PLOT, X, Y, XTITLE='!6X', YTITLE='!6SIN(X/10)', TITLE='!17My Plot'
;
RETURN
END
```

As can be seen, the character sizes of the x and y axes got bigger. This is due to the fact that these axes font sizes are determine with a multiplication of !X.CHARSIZE*!P.CHARSIZE and !Y.CHARSIZE*!P.CHARSIZE. As such, change the font sizes to

```
!P.CHARSIZE = 2.0  &  !X.CHARSIZE = 0.8  &  !Y.CHARSIZE = 0.8
```

As can now be seen, we need to adjust the plot window size again to produce an attractive plot.

```
!P.POSITION = [0.18, 0.18, 0.86, 0.86]
```

4 Additional IDL Programming Exercises

We will now include a second plot to this program. Note however that if one simply includes another PLOT command after the original command, the graphics containing the first plot will immediately be overwritten with the second plot not giving you enough time to view the first plot. As such, to add a second plot to this code, enter the lines marked with '***' at the end of the lines.

```
PRO MYCODE
;
; This is a test code using the IDL Command Line utility.
;
X = FINDGEN(1000)
Y = SIN(X/10)
Y2 = Y * EXP(-X/300)
;
!P.CHARSIZE = 2.0  &  !X.CHARSIZE = 0.8  &  !Y.CHARSIZE = 0.8
!P.POSITION = [0.18, 0.18, 0.86, 0.86]
;
PLOT, X, Y, XTITLE='!6X', YTITLE='!6SIN(X/10)', TITLE='!17My Plot'
;
ASK = ''   ; ***
READ, ASK  ; ***
;
PLOT, X, Y2, XTITLE='!6X', YTITLE='!6SIN(X/10)*e!U-X/300!N', $  ; ***
    TITLE='!17My 2nd Plot'  ; ***
```

```
;
RETURN
END
```

Note that the '$' sign at the end of the second plot command tells IDL that the command continues on the next line. In the YTITLE keyword, '!U' tells IDL to place the following characters in exponent model (U for 'up') and '!N' places the string back in normal font mode. The READ command was inserted so that IDL will wait for user interaction before continuing. IDL assumes that all variables are FLOATs if they are not defined. As such, we first defined ASK as a string variable so we simply can press the 'Enter' button on the keyboard to continue, otherwise, IDL would wait until we actually entered a number at the keyboard had we not defined ASK as a string variable.

What if we want to place this second plot on the same plot as the first plot? To do this we use the OPLOT command instead of the PLOT command. Let's go ahead and comment out the READ statement too since only one plot will be generated.

```
PRO MYCODE
;
; This is a test code using the IDL Command Line utility.
;
X = FINDGEN(1000)
Y = SIN(X/10)
Y2 = Y * EXP(-X/300)
;
!P.CHARSIZE = 2.0  &  !X.CHARSIZE = 0.8  &  !Y.CHARSIZE = 0.8
!P.POSITION = [0.18, 0.18, 0.86, 0.86]
;
PLOT, X, Y, XTITLE='!6X', YTITLE='!6SIN(X/10)', TITLE='!17My Plot'
;
ASK = ''
; READ, ASK
;
OPLOT, X, Y2
;
RETURN
END
```

Perhaps we now wish to make the plot in our second function a dashed line instead of a solid line. To do this we simply add the LINE keyword to the OPLOT command:

```
OPLOT, X, Y2, LINE=2
```

Of course our y-axis title now does not reflect what is being plotted. There are several ways to remedy this. One way would be to change the y-axis title to something appropriate, and making use of the XYOUTS command to place text inside the plotting region. So that the text in the plot does not fall on top of the displayed sine curves, we'll need to add a little extra blank space at the top and bottom of the y-axis. We do this by adding the YRANGE keyword to the PLOT command. At the same time we will change the y-axis title to 'Functions':

```
PLOT, X, Y, XTITLE='!6X', YTITLE='Functions', TITLE='!17My Plot', $
    YRANGE=[-1.5,1.5]
```

where the y-axis will now range between –1.5 and 1.5. After the OPLOT command include the next two lines:

```
XYOUTS, 100, -1.3, '!6Solid: SIN(X/10)', SIZE=1.2
XYOUTS, 100,  1.2, '!6Dashed: SIN(X/10)*e!U-X/300!N', SIZE=1.2
```

At this point, start making use of the IDL Help utility by entering a question mark (?) at the IDL prompt to learn more about IDL. Once the GUI appears, select the IDL Users' Guide and review what is presented here. Note that one pages through the sections by clicking on the arrow buttons on the lower right of this GUI.

I have included an IDL Procedures Web Page on the Course Web Page. Follow the directions on this page to save the sampleplot.pro procedure. Look over this procedure to explore further into IDL programming. The procedure sampleplot.pro is a very robust code which allows the user to make very attractive plots.

5 Finishing Up

Note that one can always check out what files are located on the current directory while in IDL by entering the following command at the IDL> prompt:

```
IDL> $ ls
```

where the '$' sign tells IDL that this is an operating system command and 'ls' is the Unix/Linux command for a directory listing.

One can also find out the name of the current directory one is in by issuing the following at the IDL> prompt:

```
IDL> cd, current=mydir
IDL> help, mydir
```

where the 'current' keyword tells the 'cd' command to place the name of the current directory in the string variable 'mydir'. The help command will then give you the details of the 'mydir' variable. One could have also just printed the value stored in mydir with

```
IDL> print, mydir
```

That's it for now. Make sure you exit out of IDL before logging off of your machine. One simply does this by entering exit at the IDL prompt. Note that if you had started the IDL Help utility, close that GUI too by clicking on the 'X' button at the top right of the GUI.

We could have also retrieved the current directory from Unix/Linux with the following command in IDL:

```
IDL> $ pwd
```

where 'pwd' is the Unix command for 'print working directory.'

That's it for now. Make sure you exit out of IDL before logging off of your machine. One simply does this by entering exit at the IDL prompt.