

# Astronomy 421 – Introduction to Plotting with SM

## The Value of a Plot

Let's face it: Astronomy is an abstract science. It is often very challenging to convey your ideas in English or Math. Plots are often the best way to show your point. Whenever we read a paper, we usually examine the plots before we go reading any of the text. Most the time the crux of any paper is in the figures and figure captions. Therefore learning how to make precise and meaningful plots is one of the most important skills you will learn.

## Why SM?

Like IRAF, SM is a very widely distributed, free software program. It may not be the easiest or the slickest, but it is the old standby. Other plotting packages, such as IDL (which costs a lot of \$\$) are out there, and you may find that you eventually prefer these; but most of the basic features will be the same.

The general format of SM is to read in columns of data, process the data in some way, format the appearance of the plot, and then plot the data on some medium (usually the screen or a file). Not surprisingly we often do this with the use of macros, usually saved in a file with a **.sm** extension.

## SM Basics

Let's take a look at the most important SM commands.

device <i>devicename</i>	Direct the plot to <i>devicename</i>
data <i>filename</i>	Designate the source of information
xc <i>coll</i> yc <i>col2</i>	Read in <i>x</i> and <i>y</i> directly
read { <i>var1 coll ... varn coln</i> }	Read in columns of data
set <i>var = expr</i>	Mathematical operations
limits <i>var1 var2</i>	Set the boundaries of axes
box	Draw a box
points <i>var1 var2</i>	Put points on the graph
connect <i>var1 var2</i>	Connect the points with a line
xlabel <i>string</i>	Label the x-axis
ylabel <i>string</i>	Label the y-axis
erase	Start over
expand <i>factor</i>	Make points and text bigger/smaller
relocate <i>x y</i>	Move to new point in plot ( <i>x,y</i> defined by axes)
draw <i>x y</i>	Draw a line from current position to <i>x,y</i>
label <i>string</i>	Place string on the plot at the current position
ctype <i>color</i>	Change plot color ( <i>color</i> is the name of a color)
error_y <i>x y yerr</i>	Plot error bars in y
error_x <i>x y xerr</i>	Plot error bars in x
macro read <i>filename</i>	Read in a macro
help <i>command</i>	Get help on <i>command</i>
hardcopy	Force creation of a postscript file
#	Comment

First let's talk about the device. Usually you will only want to make two types of plots: screen plots, or printable plots. To output to the screen the *devicename* is *x11*, to output to a printable file you type

```
:device postencap file.ps
```

where the *postencap* designates the output as a postscript, and *file.ps* is the name you want the plot have. The rest of the commands are pretty self-explanatory, with a few examples first.

### Example 1

As a simple example let's look at this macro called `ex1.sm` (the indentation for the lines following the macro name is required!). Edit and save a file called `ex1.sm` (this is for question 1 of the exercise part), putting in the following lines:

```
ex1
  set t=0,2*pi,0.1
  set y=sin(t)
  limits t y
  box
  points t y
  xlabel Time
  ylabel y
```

What would you expect this script to do? The two `set` commands create 2 arrays of data. Note that with `set` we can, in one swoop, create an array. Note also that “pi” can be explicitly stated. The second set command creates a new array, `y`, which is just the sine of `t`. In order to plot this, you would type from your Linux command line prompt:

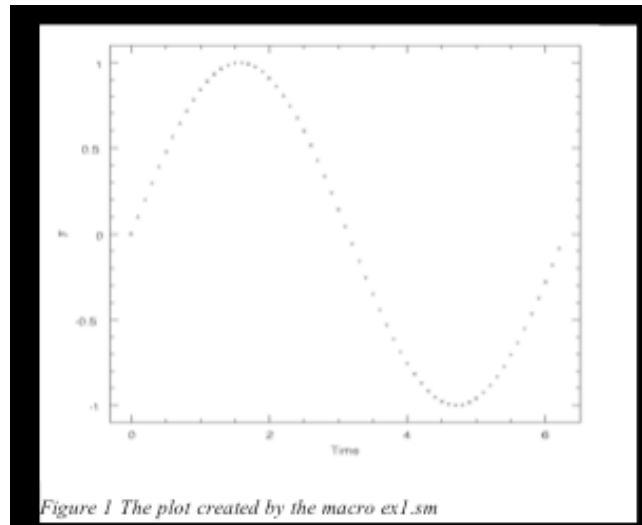
```
> sm
:macro read ex1.sm          # this identifies the macro file name
:device x11                 # postencap ex1.ps if printing
:ex1                       # this calls the macro. Note, you can have
                           # more than one macro in a macro file.
```

This would make the plot in Figure 1. Notice that the first line of my macro file is the name of the macro.

As we can see from the implementation above, you can mix macros and single commands. We could have included the device command in the macro, but usually we will want to plot first to `x11`, correct any mistakes, and then plot to the postscript.

Another common mistake is to forget to “flush” the graphics. Because of the nature of plotting, you must explicitly tell SM when you are done. This is usually only important when making postscripts. There are three ways to flush your plot, the `hardcopy` command, redefining your device (either as `x11` or another postencap), or quitting SM.

```
>sm
:macro read ex1.sm
:device postencap ex1.ps
:ex1
:hardcopy
:quit
```



SM does have some additional aggravating features, but these are easily overlooked once recognized, since its ability to plot very complicated graphs far outweighs any annoyances.