Changing the look of Sherpa plots using setplot.sl

Sherpa Threads (CIAO 3.4)

Table of Contents

- ***Getting Started***
    - Downloading the setplot.sl script
    - Loading the setplot.sl script into Sherpa
    - Downloading the data
- ***Background Information***
- ***Changing "DATA" plots***
- ***Changing the "ARF" plot to match***
- ***Changing the "FIT" plots***
- ***Saving your changes***
- ***Notes & Caveats***
    - The sherpa.multiplot object
    - The prefunc & postfunc settings
- ***Summary***
- ***History***
- ***Images***
    - The plot produced by LP DATA with the default settings
    - The plot produced by LP DATA after changing the settings
    - The plot produced by LP DATA ARF
    - Applying a similar customisation to ARF and DATA plots
    - LP MODEL is also changed
    - The plot produced by LP FIT

Changing the look of Sherpa plots using setplot.sl

Sherpa Threads

Overview

Last Update: 1 Dec 2006 − reviewed for CIAO 3.4: no changes

Synopsis:

Sherpa provides a number of plots suitable for data analysis, and the Data Visualization thread shows you how you can use *ChIPS* commands to modify the plots once they are created.

It is also possible to customise the appearances of these plots automatically using the plot configuration variables in *Sherpa*. The use of the configuration variables provides great flexibility but is not simple to use for the occasional user. In this thread we provide a function that makes it easy to change the looks of plots; it is written in S−Lang, but you do not need to know this to use the script!

Read this thread if:

You want to change the look of a plot every time it is created.

Related Links:

- The Step−by−Step guide to changing the look of *Sherpa* plots thread shows how you can make the same changes to your plots using the *Sherpa* plotting variables.
- The Data Visualization thread.
- The help documents on the configuration variables (i.e. state objects) of *Sherpa* that control the plots: `sherpa.plot`, `sherpa.dataplot`, `sherpa.fitplot`, `sherpa.resplot`, and `sherpa.multiplot`.
- The Advanced customization of *Sherpa* plots thread shows how you can use the pre− and post− hooks in the configuration variables to provide almost total control over the look of the plot.
- The `save_state()` command for saving any changes you have made to the *Sherpa* configuration variables.

Proceed to the HTML or hardcopy (PDF: A4 | letter) version of the thread.

Getting Started

Downloading the setplot.sl script

The thread uses the `setplot.sl` script; for information about the script, consult the help file ("`ahelp setplot`"). The most recent version of `setplot.sl` is v 1.3 (02 Nov 2004):

```
unix% grep Id $ASCDS_CONTRIB/share/slsh/local−packages/setplot.sl
% $Id: setplot.sl,v 1.3 2004/11/02 16:19:23 dburke Exp $
```

Note that `$ASCDS_CONTRIB/share/slsh/local−packages/` is the default path in the standard CIAO scripts installation; see the Scripts page for more information. ***Please check that you are using the most recent version before continuing.*** If you do not have the script installed or need to update to a newer version, please refer to the Scripts page.

Loading the setplot.sl script into Sherpa

The `setplot.sl` script is loaded into a *Sherpa* session with the `evalfile` function:

```
sherpa> () = evalfile("setplot.sl");
```

If you wish the functions to always be available to *Sherpa*, add the following line to your `~/.sherparc` file:

```
() = evalfile("setplot.sl");
```

For more information on configuring *Sherpa*, see the Customizing Sherpa with a Resource File thread.

Downloading the data

The data used in this thread is available in the sherpa.tar.gz file, as described in the "Getting Started" thread.

For the various plots discussed below we use the same dataset and model as used in the Estimating Errors and Confidence Levels thread:

```
sherpa> data source_grouped_pi.fits
The inferred file type is PHA.  If this is not what you want, please
specify the type explicitly in the data command.
WARNING: statistical errors specified in the PHA file.
         These are currently IGNORED.  To use them, type:
         READ ERRORS "<filename>[cols CHANNEL,STAT_ERR]" fitsbin
RMF is being input from:
  /data/ciao/rmf.fits
ARF is being input from:
  /data/ciao/arf.fits
sherpa> ignore energy :0.5,8:
sherpa> source = xswabs[abs] * powlaw1d[p1]
abs.nH parameter value [0.1]
p1.gamma parameter value [1]
p1.ref parameter value [4]
p1.ampl parameter value [0.000149261]
sherpa> fit
 LVMQT: V2.0
 LVMQT: initial statistic value = 4583.05
 LVMQT: final statistic value = 83.2873 at iteration 8
           abs.nH  2.4061  10^22/cm^2
```

```
            p1.gamma  1.51851
            p1.ampl  0.000241434
```

Background Information

Plots are created in *Sherpa* by the `LPLOT` command.

Many aspects of the appearance of the plots can be changed by setting values in the *Sherpa* configuration variables (state objects). This is discussed in the Configuration of Sherpa with Sherpa State Objects section of `ahelp sherpa`, and the variables relevant to plotting are: `sherpa.plot`, `sherpa.dataplot`, `sherpa.fitplot`, `sherpa.resplot`, and `sherpa.multiplot`.

By changing the values of the fields in these variables you change how new plots of the given type will be drawn. There are also several utility routines − such as `set_log`, `set_lin`, `set_erron`, and `set_erroff` to change some of the common options.

This approach provides great power and flexibility, but may not be easy to remember for many users. In this thread we describe how you can use the `setplot()` function from the `setplot.sl` script to easily set the options for the different plot types. The function prompts you for all parameters relevant to the requested plot and sets the appropriate fields in the *Sherpa* plot variables. The rest of the thread describes how to use this function.

Changing "DATA" plots

The default settings for "DATA" plots produces this plot for the dataset used in this example. The look of the plot is controlled by the fields of the `sherpa.dataplot` plotting variable. These can be viewed using the `print()` command:

```
sherpa> print(sherpa.dataplot)
x_errorbars      =  0
y_errorbars      =  1
errs_style       =  bar
errs_type        =  both
x_log            =  0
y_log            =  0
curvestyle       =  noline
curvecolor       =  default
symbolstyle      =  square
symbolcolor      =  default
symbolsize       =  2
xlabel_size      =  1.5
ylabel_size      =  1.5
zlabel_size      =  1.5
title_size       =  1.5
tickvals_size    =  1.5
prefunc          =  NULL
postfunc         =  NULL
```

We decide we want the y axis to be drawn in a logarithmic scale and the symbols to be drawn in magenta whilst keeping the remaining options unchanged. Although we could change the fields manually, we use the `setplot()` command to prompt us for the settings:

```
sherpa> setplot("data")
Should the X-axis have logarithmic spacing? (no):
Should the Y-axis have logarithmic spacing? (no): yes
Draw error bars along the X-axis? (no):
Draw error bars along the Y-axis? (yes):
Style of error bars? (bar|standard) (bar):
Type of error bars? (both|none|up|down|dn) (both):
Line style to connect points (step|histo|noline|simpleline) (noline):
Symbol used to mark points (none|bigpoint|...|uptri) (square):
Color of symbols (default|...|yellow) (default): magenta
Size of symbols (0:) (2):
Size of labels along X axis (0:) (1.5):
Size of labels along Y axis (0:) (1.5):
Size of title (0:) (1.5):
Size of numbers along axes (0:) (1.5):

The parameters for data (and related) plots has been set.
Try 'lplot data'

sherpa> lplot data
```

The new version of the plot looks like this. To avoid the screen text being too wide the list of options for the symbol style has been cut down to just "`none|bigpoint|...|uptri`" and the color range to just "`default|...|yellow`".

The bold entries in the following indicate the changed fields in the `sherpa.dataplot` variable:

```
sherpa> print(sherpa.dataplot)
x_errorbars      =  0
y_errorbars      =  1
errs_style       =  bar
errs_type        =  both
x_log            =  0
y_log            =  1
curvestyle       =  noline
curvecolor       =  default
symbolstyle      =  square
symbolcolor      =  magenta
symbolsize       =  2
xlabel_size      =  1.5
ylabel_size      =  1.5
zlabel_size      =  1.5
title_size       =  1.5
tickvals_size    =  1.5
prefunc          =  NULL
postfunc         =  NULL
```

We now decide that we want to make further changes to the plot and re−run the `setplot()` command:

```
sherpa> setplot("data")
Should the X-axis have logarithmic spacing? (no): yes
Should the Y-axis have logarithmic spacing? (yes):
Draw error bars along the X-axis? (no):
Draw error bars along the Y-axis? (yes):
Style of error bars? (bar|standard) (bar):
Type of error bars? (both|none|up|down|dn) (both):
Line style to connect points (step|histo|noline|simpleline) (noline): step
Color to draw line connecting points (default|...|yellow) (default): cyan
Symbol used to mark points (none|bigpoint|...|uptri) (square): none
Size of labels along X axis (0:) (1.5):
Size of labels along Y axis (0:) (1.5):
```

```
Size of title (0:) (1.5): 2
Size of numbers along axes (0:) (1.5):

The parameters for data (and related) plots has been set.
Try 'lplot data'

sherpa> lplot 2 data arf
```

Note that the default values for the prompts have changed to reflect the current settings, and that the variables being prompted for have changed to reflect the settings: there are no questions about the symbol color or size since the symbol style was set to `none`, but the curve color was asked for since it is now being drawn.

The resulting plot looks like this. The top plot has been changed to match the settings asked for in the `setplot()` call. The bottom plot − of the ARF − does not use any of these settings, because the ARF plot is controlled by the `sherpa.plot` plot variable.

Although the plot title was changed to a size of 2, from its default value of 1.5, the new plot does not have a title. This is because we draw two plots here and *Sherpa* only sets the title when an individual plot − i.e. "`lplot data`" − is created.

Changing the "ARF" plot to match

In the previous section we changed *Sherpa* so that all "DATA" plots have a garish color scheme. However, other plots − such as of the ARF − do not share this scheme. We can use the setplot() call to change this:

```
sherpa> setplot("arf")
Should the X−axis have logarithmic spacing? (no): yes
Should the Y−axis have logarithmic spacing? (no): yes
Draw error bars along the X−axis? (no):
Draw error bars along the Y−axis? (no):
Line style to connect points (step|histo|noline|simpleline) (step):
Color to draw line connecting points (default|...|yellow) (default): cyan
Symbol used to mark points (none|...|uptri) (none):
Size of labels along X axis (0:) (1.5):
Size of labels along Y axis (0:) (1.5):
Size of title (0:) (1.5): 2
Size of numbers along axes (0:) (1.5):

The parameters for arf (and related) plots has been set.
Try 'lplot arf'

sherpa> lplot 2 data arf
```

which produces this plot.

Since the `sherpa.plot` plot object controls the look of many plots, not just those of the ARF, the following changes will also be seen in these other plots. For instance:

```
sherpa> lp model
```

The resulting plot matches the style of the ARF plot. Since there is only one plot here we can see that the title has indeed been increased.

Changing the "FIT" plots

Although there are several different plot variables:

- `sherpa.plot` for general plots, such as the ARF and model components
- `sherpa.dataplot` for plotting the source and background data
- `sherpa.resplot` for plotting "residual" style plots
- and `sherpa.fitplot` for plotting fits

they are all essentially identical − *except* for fits. This is because the `sherpa.fitplot` object contains extra variables to control how the fitted model is displayed.

Here we change the fit−style plots; although most of the prompts are the same as seen with previous plots, there are extra ones (beginning with "`Fit:`"):

```
sherpa> setplot("fit")
Should the X−axis have logarithmic spacing? (no): yes
Should the Y−axis have logarithmic spacing? (no): yes
Draw error bars along the X−axis? (no):
Draw error bars along the Y−axis? (yes):
Style of error bars? (bar|standard) (bar):
Type of error bars? (both|none|up|down|dn) (both):
Line style to connect points (step|histo|noline|simpleline) (noline):
Symbol used to mark points (none|...|uptri) (square): bigpoint
Color of symbols (default|...|yellow) (default): blue
Size of symbols (0:) (2): 1
Fit: line style to connect points (step|histo|noline|simpleline) (step):
Fit: color to draw line connecting points (default|...|yellow) (red): green
Fit: symbol used to mark points (none|...|uptri) (none):
Size of labels along X axis (0:) (1.5):
Size of labels along Y axis (0:) (1.5):
Size of title (0:) (1.5):
Size of numbers along axes (0:) (1.5): 2

The parameters for fit (and related) plots has been set.
Try 'lplot fit'

sherpa> lp fit
```

The resulting plot looks like this.

Saving your changes

The `save_state()` command can be used to save your settings so that they can be used in other *Sherpa* sessions.

If called with no arguments, `save_state()` will write out the contents of the *Sherpa* configuration variables to the file `$HOME/.sherpa−state−rc`. This file will be over−written without warning (so you should not make any manual changes to it). When *Sherpa* starts, it will automatically load in the settings from this file, so your plots will retain the look you have set for them.

If called with an argument, `save_state()` will write out the settings to the file name (its argument) instead of `$HOME/.sherpa-state-rc`. This file can then be read into a *Sherpa* session using the `USE` command; this is useful if you want to set up different plot styles for use in different situations.

The beginning of the file will look something like:

```
% Sherpa state for ciaouser, Wed Jul 30 17:45:13 2006

sherpa.plot.x_errorbars = 0
sherpa.plot.y_errorbars = 0
sherpa.plot.errs_style = "bar"
sherpa.plot.errs_type = "both"
sherpa.plot.x_log = 1
sherpa.plot.y_log = 1
sherpa.plot.curvestyle = "step"
sherpa.plot.curvecolor = "cyan"
sherpa.plot.symbolstyle = "none"
sherpa.plot.symbolcolor = "default"
sherpa.plot.symbolsize = 2
sherpa.plot.xlabel_size = 1.5
sherpa.plot.ylabel_size = 1.5
sherpa.plot.zlabel_size = 1.5
sherpa.plot.title_size = 2
sherpa.plot.tickvals_size = 1.5
sherpa.plot.prefunc = NULL
sherpa.plot.postfunc = NULL

...
```

(the values will depend on your responses to any `setplot()` calls you have made).

Notes & Caveats

The sherpa.multiplot object

The `sherpa.multiplot` plot variable controls certain aspects of the plots in *Sherpa* that are not in the other `sherpa.*plot` variables. At present this information can not be changed by the `setplot()` function.

The prefunc & postfunc settings

All the `sherpa.*plot` variables that can be changed by `setplot()` contain `prefunc` and `postfunc` fields. These can not be set by the `setplot()` function. More information on these fields is available in "`ahelp sherpa-plot-hooks`" and the Advanced customization of *Sherpa* plots thread.

Summary

The `setplot()` command provides a simple way to change the default look of plots in *Sherpa*. It does not provide unlimited customization; for this you need to use the `prefunc` and `postfunc` variables as described in the Advanced customization of *Sherpa* plots thread.

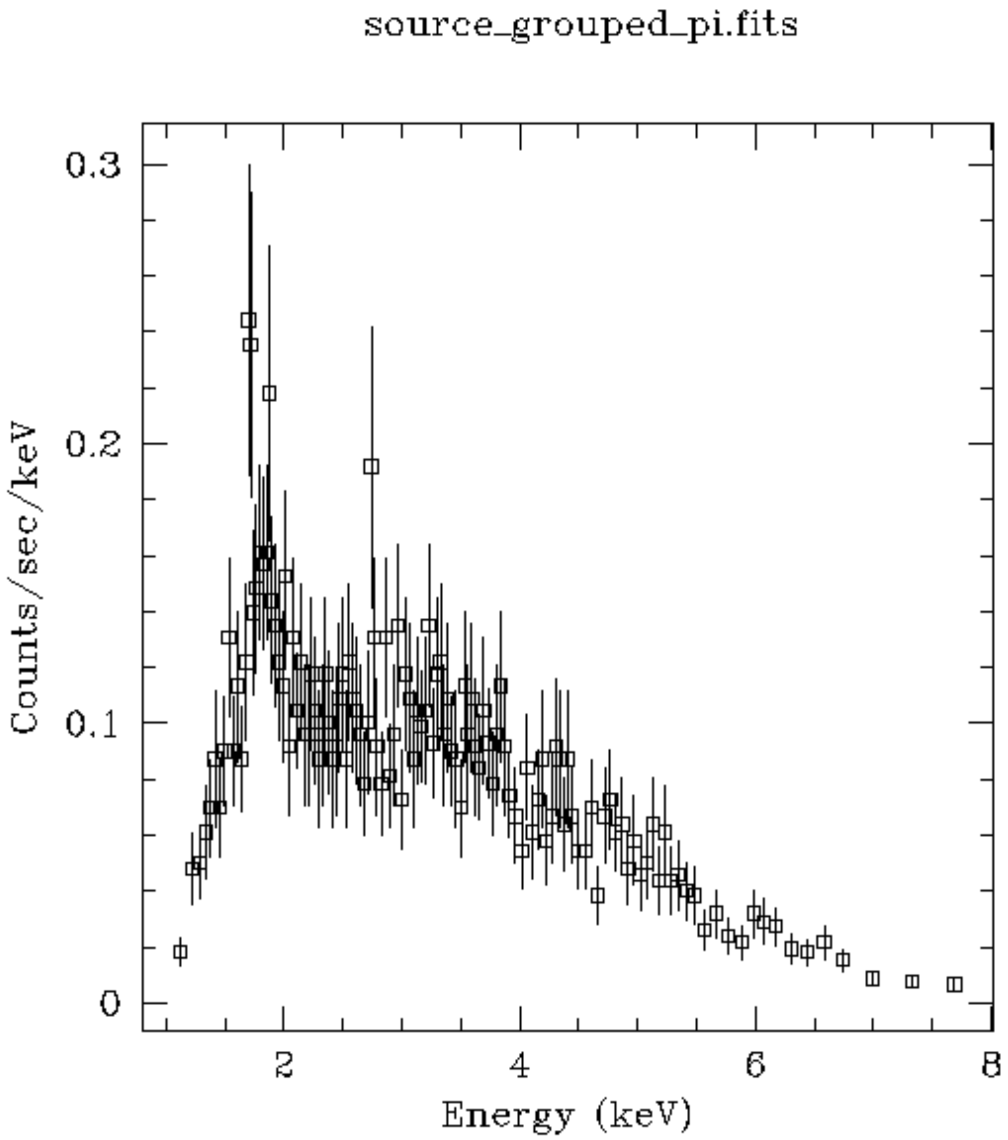
The Changing the look of Sherpa plots using setplot.sl thread shows how you can change the *Sherpa* plotting variables directly rather than using `setplot()`. Note that both methods can be used at the same time.

History

14 Jan 2005	reviewed for CIAO 3.2: no changes
21 Dec 2005	reviewed for CIAO 3.3: no changes
01 Dec 2006	reviewed for CIAO 3.4: no changes

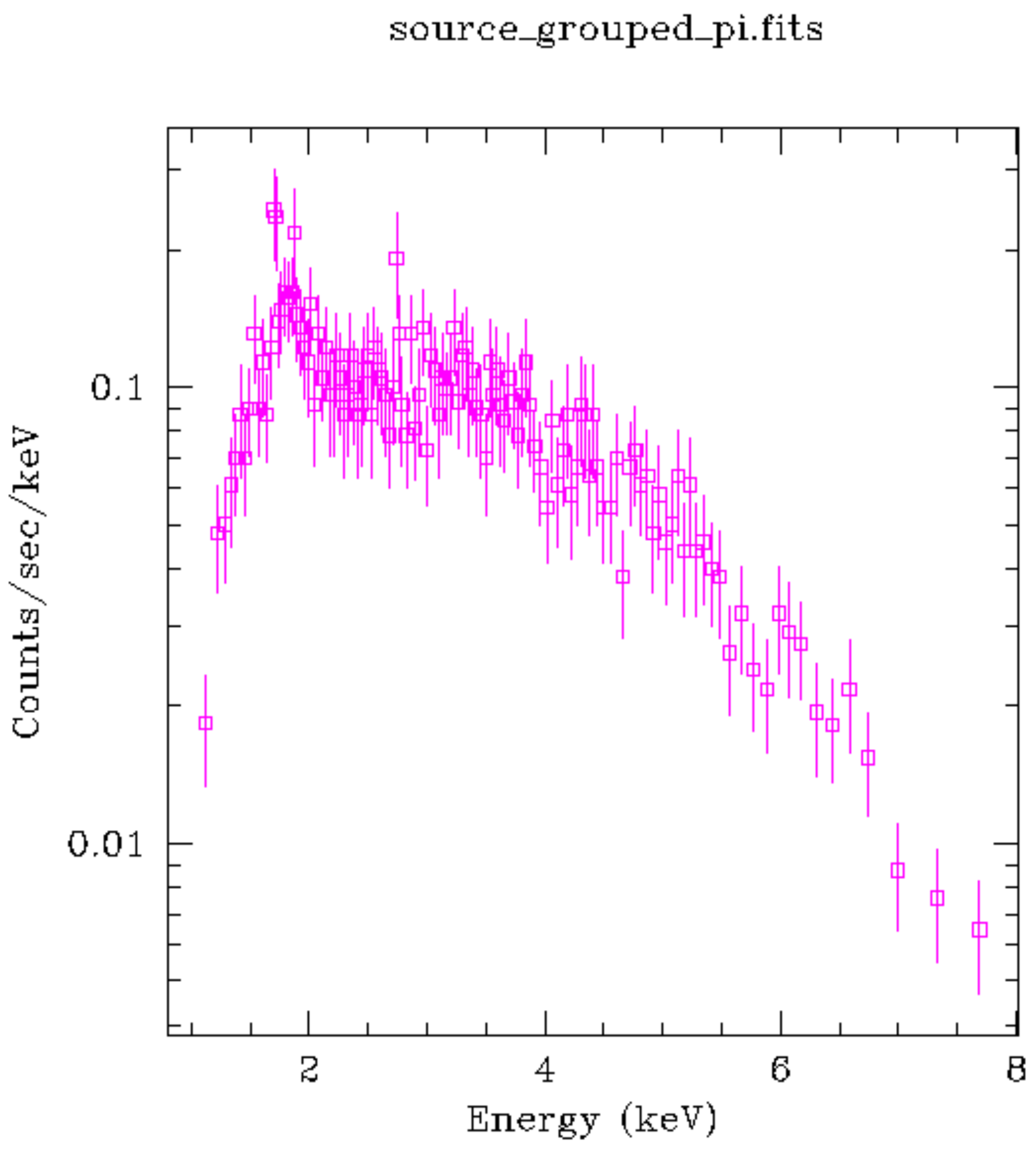
URL: http://cxc.harvard.edu/sherpa/threads/setplot/ Last modified: 1 Dec 2006

Image 1: The plot produced by LP DATA with the default settings



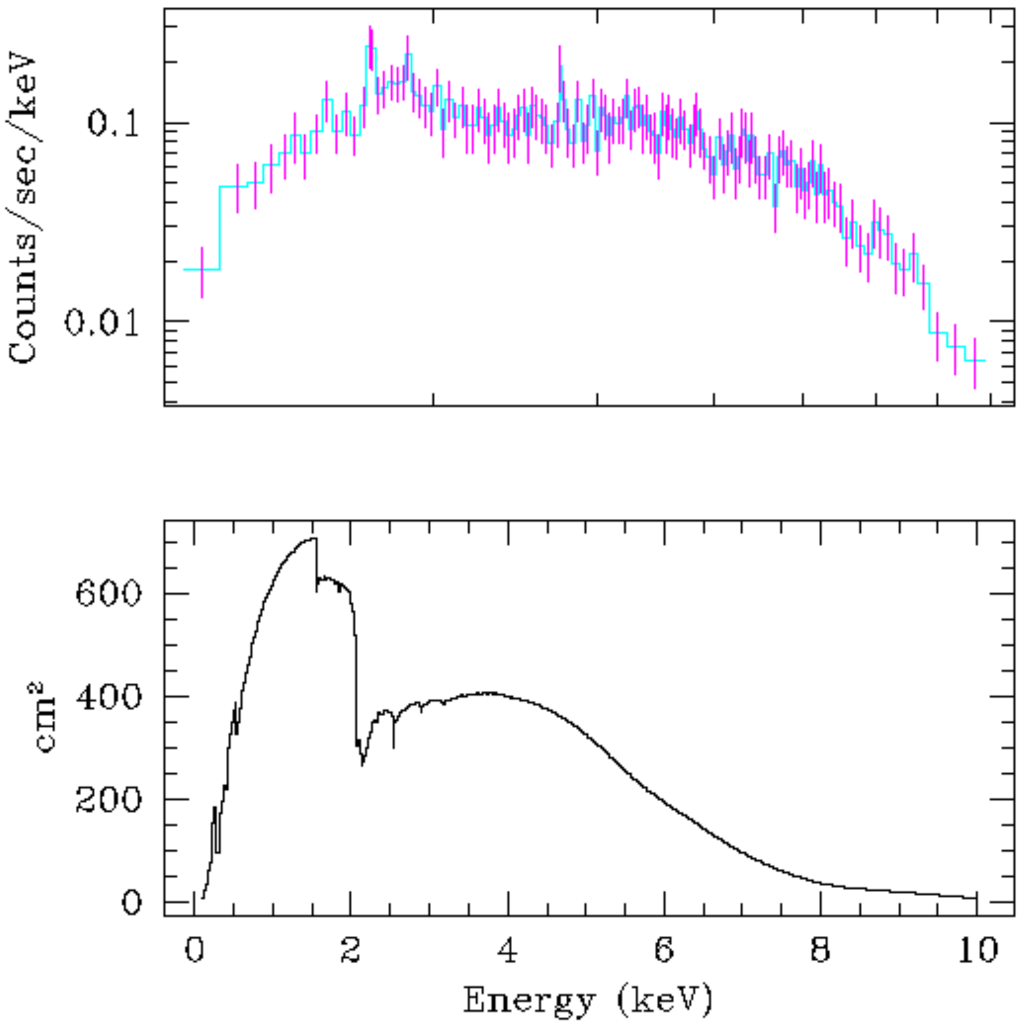
This shows the plot produced by the `LPLOT DATA` command before changing any *Sherpa* plot variables.

Image 2: The plot produced by LP DATA after changing the settings



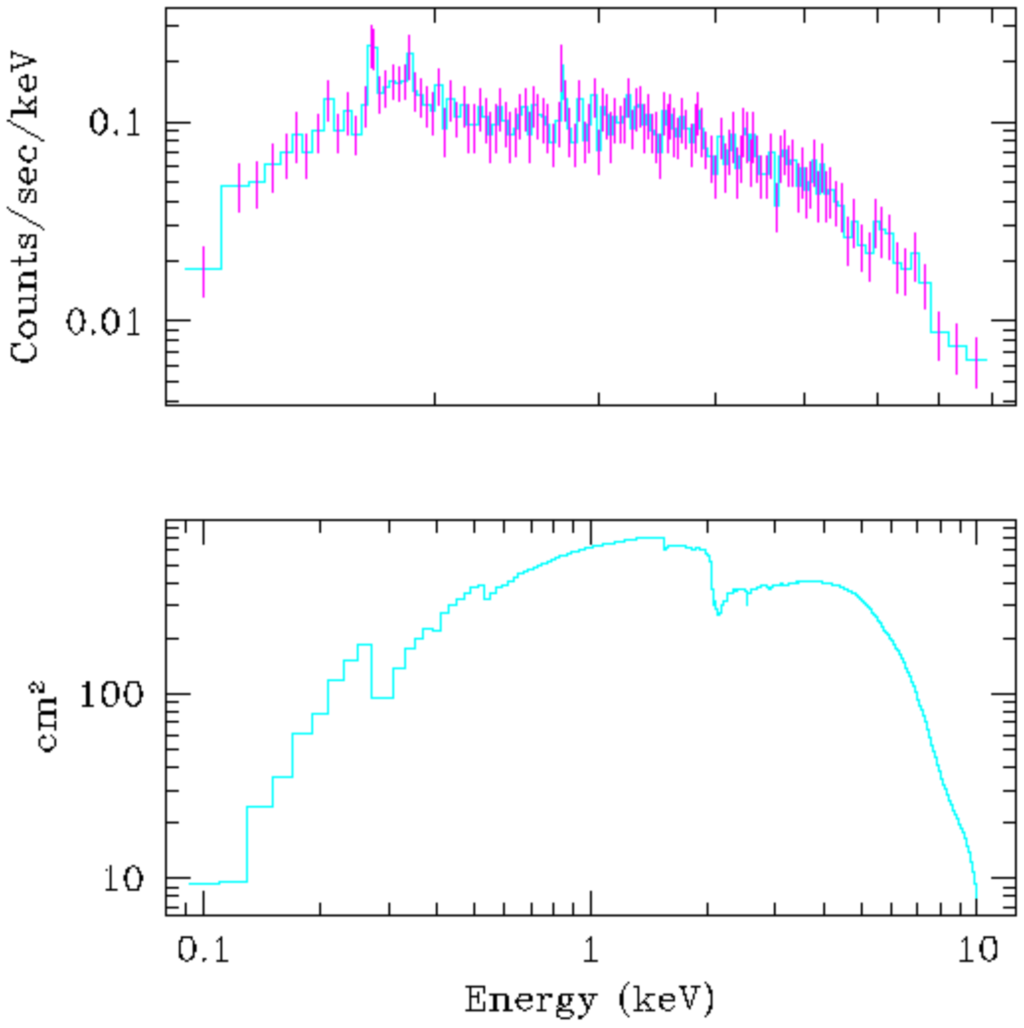
This shows the plot produced by the `LPLOT DATA` command after changing the settings with the `setplot()` function. The y axis is drawn with a logarithmic scale and the symbols (and error bars) are drawn in magenta.

Image 3: The plot produced by LP DATA ARF



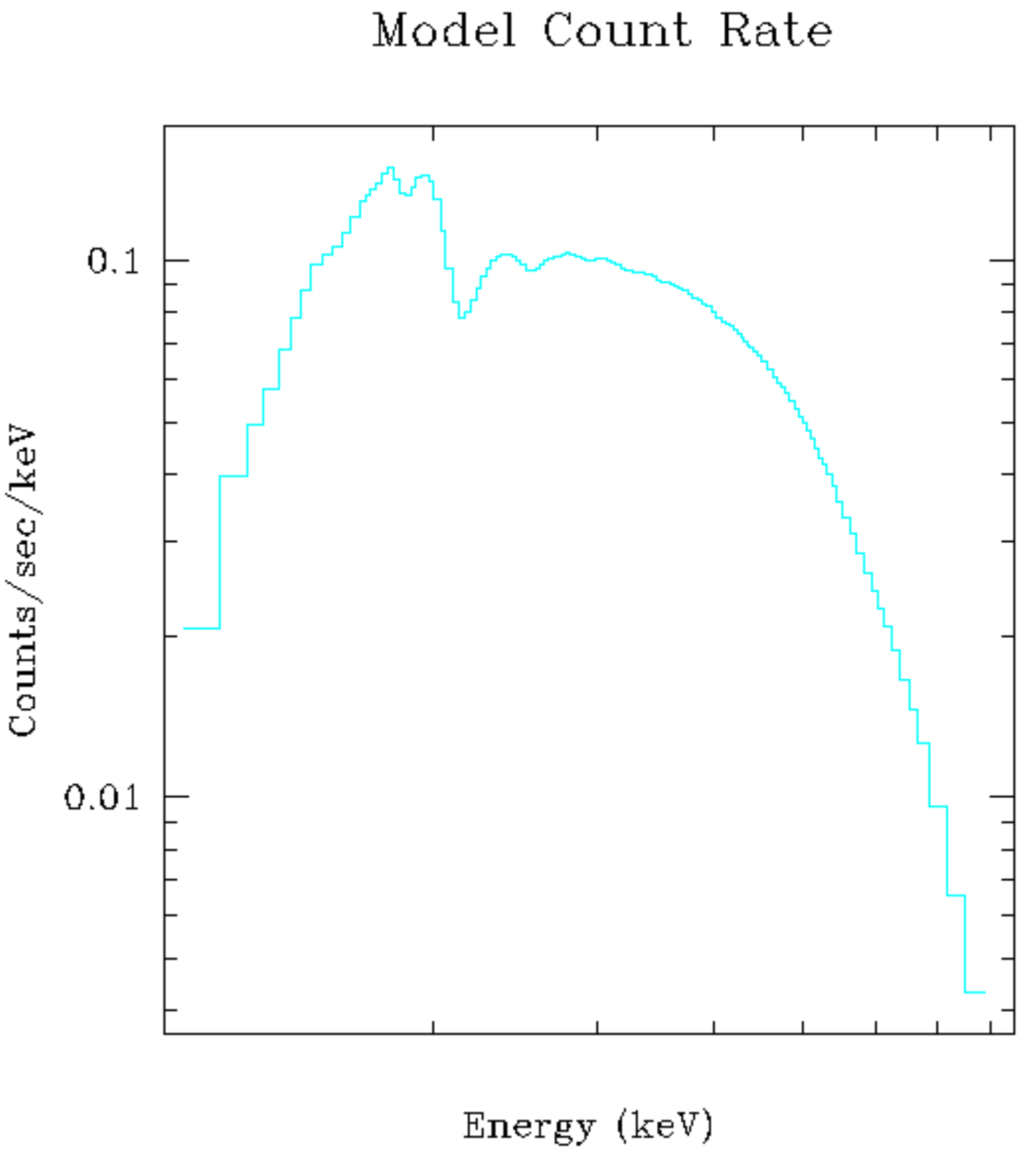
This shows the plot produced by the `LPLOT DATA ARF`. The plot of the DATA section uses the new settings but the ARF plot does not, since it uses the settings defined in the `sherpa.plot` variable.

Image 4: Applying a similar customisation to ARF and DATA plots



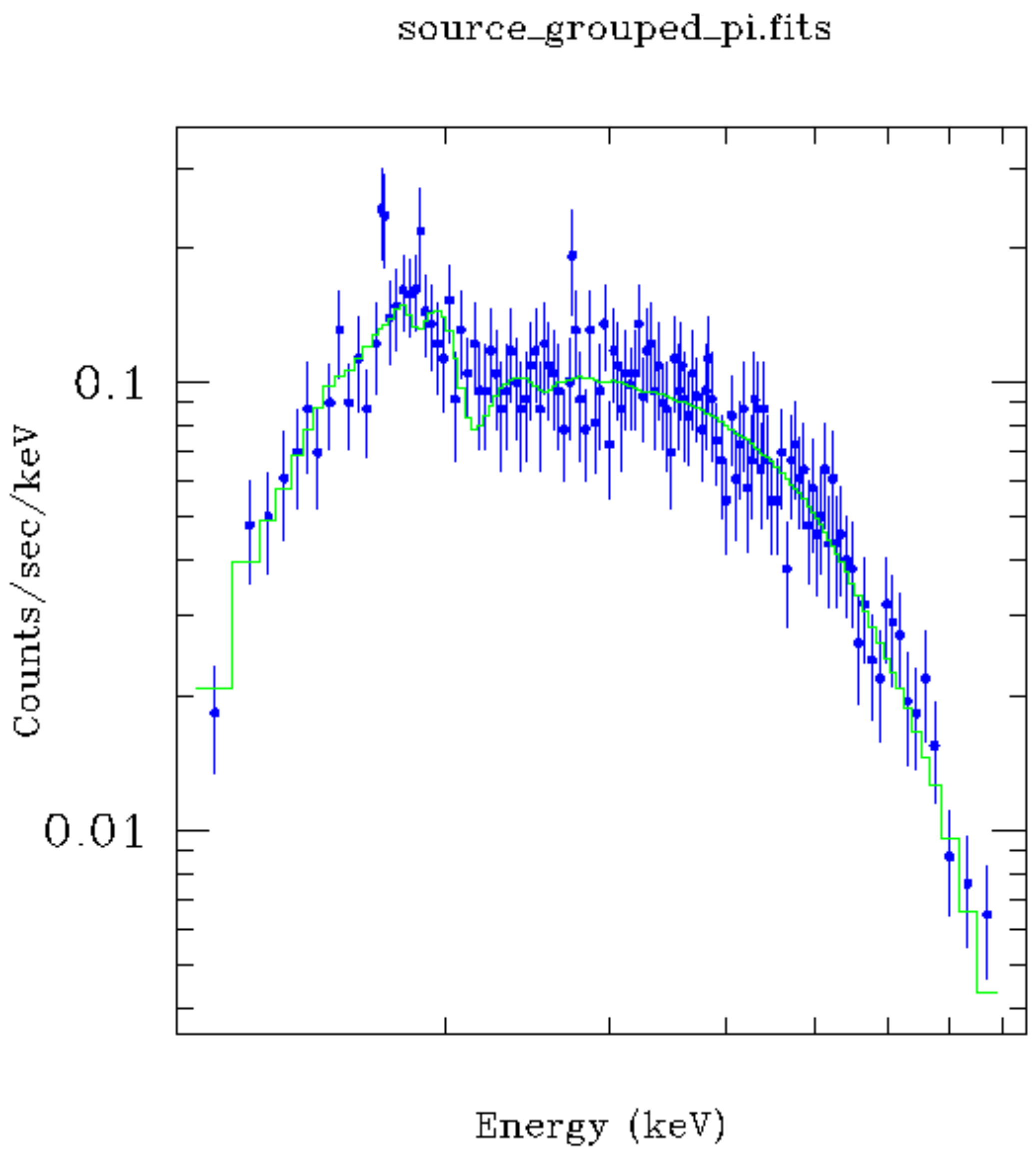
Here we have changed the look of both the DATA and ARF plots to be similar (compare to the previous version of the plot). Note that the X−axes − though in a logarithmic scale for both plots − do not cover the same range.

Image 5: LP MODEL is also changed



Since the same plot variable is used to configure model and ARF plots − as well as many other plots − the model plot matches the style set up for the ARF plot. Since we now have a title we can see that the size of the title has been increased.

Image 6: The plot produced by LP FIT


source_grouped_pi.fits
Counts/sec/keV
Energy (keV)