Astronomical Data Analysis Software and Systems XVI
ASP Conference Series, Vol. 376, 2007
R. A. Shaw, F. Hill and D. J. Bell, eds.

CIAO 4 Preview: Sherpa and ChIPS Demo

Brian Refsdal, Gregg Germain, Douglas Burke, Stephen Doe, Ian Evans, Janet Evans, Jonathan McDowell, Warren McLaughlin, Robert Milaszewski, Joseph Miller, Dan Nguyen, Christopher Stawarz, Aneta Siemiginowska

Smithsonian Astrophysical Observatory, Cambridge, MA, USA

John Houck, Michael Nowak

MIT Kavli Institute for Astrophysics and Space Research, Cambridge, MA, USA

Abstract. Two prominent applications in the *Chandra* Interactive Analysis of Observations (CIAO) software package are Sherpa, a general purpose fitting and modeling application, and ChIPS, the *Chandra* Imaging and Plotting System. We demonstrate new versions of both applications, which will be part of the upcoming CIAO 4.0 release and also be available as standalone packages independent of CIAO.

1. Introduction

Sherpa has been restructured into a set of Python modules (with computationally-intensive portions written in C++, C, or Fortran and wrapped as Python extensions). It consists of a set of low-level subpackages (model functions, statistics, optimizers, etc.), as well as high-level functions that integrate the subpackages and provide a convenient user interface (UI) for fitting. With the PySL[1] package, S-Lang[2] users can also access the same high-level UI functions to perform fits from S-Lang. Sherpa can be used to analyze images and spectra from a variety of sources (e.g. *Chandra, ROSAT, Hubble*). Sherpa is able to interoperate with other packages (some based in Python, such as PyFITS and Matplotlib, and others, such as DS9 and the XSPEC 12 model library) due to our new, modular design and implementation in Python.

ChIPS, the *Chandra* Imaging and Plotting System, is a general purpose plotting application. ChIPS provides both a high-level UI for quickly visualizing FITS files and other data sources and an extensive low-level command set that allows users to modify and refine their visualizations in a wide variety of ways. Python users can now import ChIPS as a module that provides access to all ChIPS plotting functions; S-Lang users can call any of these Python functions via PySL. ChIPS has been upgraded to a multiple client server model so that

[1]http://software.pseudogreen.org/pysl

[2]http://www.s-lang.org

multiple applications, such as Sherpa, Prism, Python, or S-Lang applications (e.g. slsh, ISIS) can all connect to and control one instance of ChIPS. We show that ChIPS can produce plots both for interactive analysis and for publication. A prime example of this is how Sherpa uses ChIPS, both to quickly produce basic plots to visualize results during fitting, and also to produce more complex, publication-quality plots once satisfactory fit results have been obtained.

2. Sherpa Example

This example details the commands of Sherpa's high-level UI used to fit a *Chandra* X-ray FITS image. A Sherpa session consists of various stages: load in the data, assign a model, fit, and display the results. A user begins by importing the Sherpa package at the Python prompt.

```
>>> from sherpa.astro.ui import *
```

Having imported the Sherpa commands into the current namespace, a user can load in data from a FITS image file, select the Cash statistic (X-ray data tend to contain low numbers of counts) and set the optimization method to Nelder-Mead (our implementation of the Simplex algorithm).

```
>>> load_image('image2.fits')
>>> set_stat('cash')
>>> set_method('neldermead')
```

A user can set a Gaussian model for the source, plus a constant model to account for the background. Selecting reasonable initial parameter values for the constant and the source's position, amplitude, and full-width half max will allow the fit to proceed more quickly if the initial parameter values are already close to the final best-fit values. The ellipticity and theta parameters can be set to remain static during fitting.

```
>>> set_model(gauss2d.g1 + const2d.bgnd)
>>> g1.xpos = 135.5
>>> g1.ypos = 115.5
>>> g1.ampl=20
>>> g1.fwhm=60
>>> bgnd.c0 = 0.2
>>> freeze(g1.ellip)
>>> freeze(g1.theta)
```

With the setup complete, Sherpa fits the data to the model and displays images of the data, model values, and residuals (see Figure 1).

```
>>> fit()
>>> image_fit()
```

3. ChIPS

ChIPS provides a high-level UI for quickly visualizing FITS files and other data sources, an extensive low-level command set that allows users to modify their

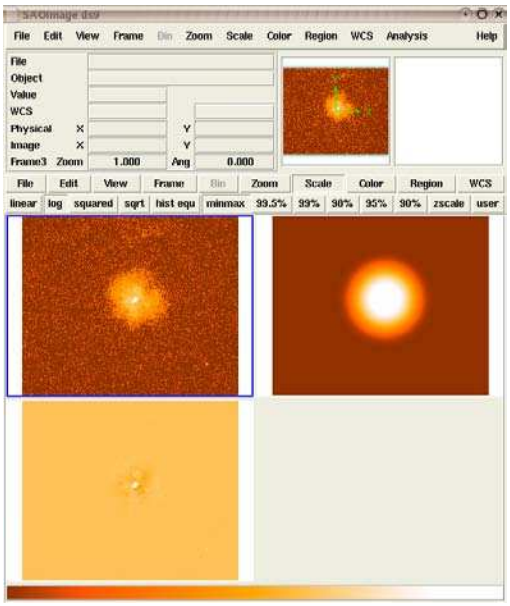

Figure 1. A Sherpa 2D image fit with residuals in DS9

plots and generate publication quality visualizations. To provide this flexibility and ease of use, ChIPS has been changed in the following ways:

- ChIPS is an importable module in either Python or S-Lang.
- ChIPS operates on the Client-Server model as shown in Figure 2.
- Complex but common visualizations have been wrapped in high level calls.
- ChIPS users can import the Crates (new CIAO Data Model interface) module for access to capable and easy to use file I/O.
- ChIPS incorporates Tangent Plane Coordinate Systems.

Scripting languages provide users with the ability to manipulate and analyze their data in scripts or at the prompt (Python or slsh). The ChIPS API has been wrapped in Python, and built as an importable module. Using PySL, a module that translates S-Lang calls to Python, S-Lang users can also use ChIPS. So now, Python and S-Lang users can visualize their data and computations on that data using ChIPS, right in line with their computations. Data for plotting can come from a variety of sources. Sherpa users will want to view the results of their fitting operations; scripting language users will need to plot data that they may have created through a series of operations - and the initial input of that data might come from PRISM (CIAO data file browser). ChIPS was designed to receive arrays for plotting and does not know or care where the arrays came from. In addition, ChIPS is built on a client server model so that multiple clients can communicate with one ChIPS server to create and/or enhance a visualization. For example, a user might have a PRISM session running and create a ChIPS plot of two columns in a FITS file. The user can then run Sherpa, produce a fit, connect to the same ChIPS server PRISM created, and overplot the fit results. The user then could open a Python session in another xterm, read in data, manipulate it, connect to the same ChIPS session and request a plot be created in another window, or in another frame in the same window.

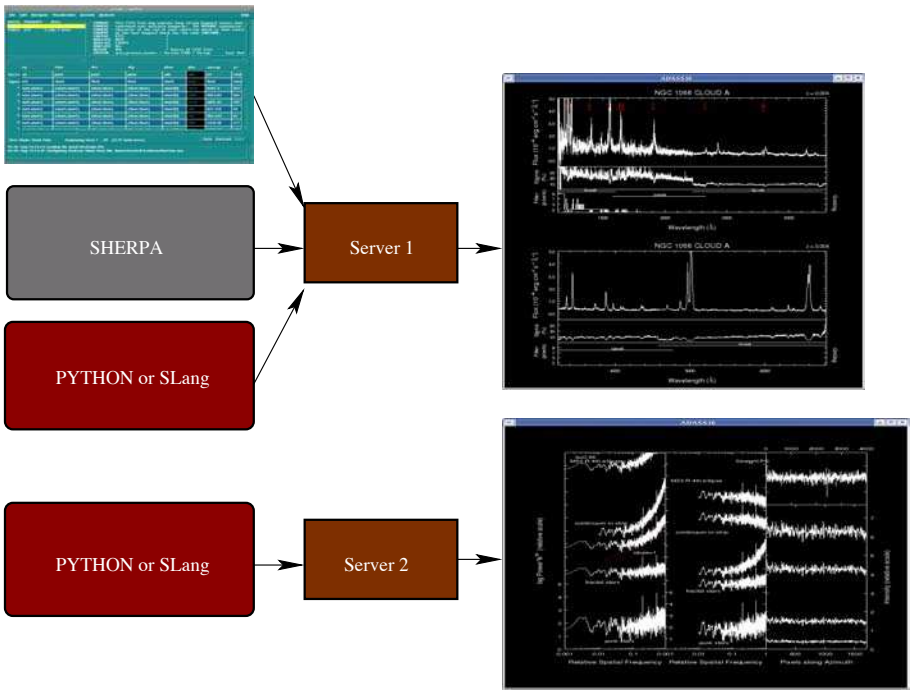

Figure 2. ChIPS client server.

Many common visualizations such as multiple panel strip plots, with shared axes, and m x n grids of plots, have been wrapped in high level function calls so that users can rapidly create these complex plots with ease. Scripting language users can import Crates for a flexible, high level I/O interface to their FITS or ASCII files, manipulate the data, and then send the arrays to the ChIPS server for visualization.

Any time a user creates an x, y axis pair, a data coordinate system is created in ChIPS. Users can place items in the visualization using that data coordinate system or various normalized coordinate systems. In addition, ChIPS has the ability to accept coordinate system transforms from the user, apply them to visualization data, or use them in the placement of visualization entities such as annotations.

4. Future ChIPS Enhancements

Future enhancements of ChIPS will include WCSLIB functionality in the form of an importable module. In addition, a Graphical User Interface will be supplied.

Acknowledgments. This project is supported by the *Chandra* X-ray Center under NASA contract NAS8-03060.