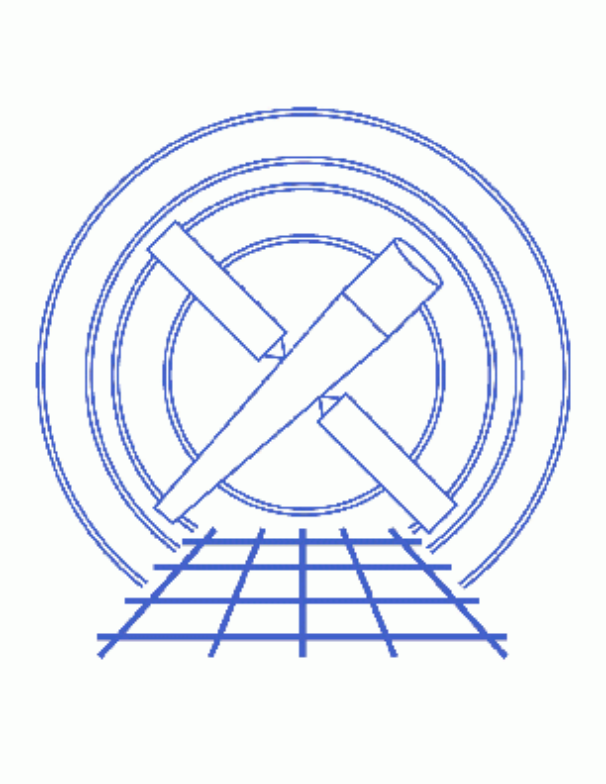
Customizing Sherpa with a Resource File



Sherpa Threads (CIAO 3.4)

Table of Contents

Customizing Sherpa with a Resource File

Sherpa Threads

Overview

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

Synopsis:

Sherpa can be customized by use of a *Sherpa* resource file. This thread explains how to use resource files to execute both *Sherpa* and S−Lang commands at startup.

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

How Sherpa Uses the Resource File

As part of its startup process, *Sherpa* checks whether a resource file exists. If one is found, the contents are processed at the start of the session. This occurs in both interactive and batch modes; however, the resource file is *not* loaded when one imports the Sherpa/S−Lang module into a S−Lang script.

Sherpa checks for a resource file in three places. First, if the environment variable `SHERPARC` contains the name of a resource file, *Sherpa* uses that file. If `SHERPARC` is not set, *Sherpa* then looks for a file named `.sherparc` in the current directory (`$PWD`). Finally, if this file is not found, it looks for a `.sherparc` file in the user's home directory (`$HOME`).

If more than one of the above files exist, *Sherpa* will use the first one it encounters and ignore the rest. Hence, `$PWD/.sherparc` overrides `$HOME/.sherparc`, and a file specified by `$SHERPARC` overrides both. Note that if any command in the resource file produces an error, an error message will be displayed, but *Sherpa* startup will continue.

In addition to its own resource file, *Sherpa* also loads the resource files for Varmm and *ChIPS* at startup. Therefore, any customizations made in these files will also be available in *Sherpa*. The loading order for resource files is Varmm first, *ChIPS* second, and *Sherpa* last. See the ahelp pages for Varmm and *ChIPS* for information on their resource files.

Creating a Resource File

A *Sherpa* resource file is simply a text file that contains *Sherpa* commands. Anything that can be entered at the *Sherpa* prompt may also appear in a resource file. Hence, in addition to *Sherpa* commands, a resource file may contain *ChIPS* commands and single−line S−Lang statements. Also, any line that begins with a # character is interpreted as a comment and not evaluated.

The following is an example resource file that causes *Sherpa* to perform the following actions at startup:

- Print two messages
- Turn off prompting for parameter values when a model is created
- Change the optimization method to `SIMPLEX`
- Define a simple S−Lang function called `q()` that allows you to exit *Sherpa* by entering just "`q`" (or "`q()`") at the prompt

```
unix% more sherparc1.shp
# Example Sherpa resource file
message("Starting to process .sherparc")
paramprompt off
method simplex
define q () { () = sherpa_eval("quit"); }
message("Finished processing .sherparc")
```

Although the first and last lines of the above example create screen output for demonstrative purposes, it is recommended that *Sherpa* resource files not contain any command that produces text or graphical output.

To try this example, download the file `sherparc1.shp` to a working directory, and then use it as follows:

```
unix% mv sherparc1.shp .sherparc
unix% sherpa
...
    Abundances set to Anders & Grevesse
Starting to process .sherparc
Model parameter prompting is off
Finished processing .sherparc

sherpa> show method
Optimization Method: Simplex

      Name       Value         Min         Max                     Description
      ----       -----         ---         ---                     -----------
 1   iters        2000           1       10000    Maximum number of iterations
 2     eps       1e−03       1e−04         100              Absolute accuracy
 3   alpha           1         0.1           2    Algorithm convergence factor
 4    beta         0.5       5e−02           1    Algorithm convergence factor
 5   gamma           2         1.1          20    Algorithm convergence factor

sherpa> q
Goodbye.
```

As expected, two messages about processing `.sherparc` were printed, parameter prompting was turned off, the optimization method was set to SIMPLEX, and typing "`q`" causes *Sherpa* to exit.

Executing S−Lang Scripts at Startup

Unlike the Varmm resource file, the *Sherpa* resource file is *not* a S−Lang script. Rather, it is a *Sherpa* script, which means it may contain only single−line S−Lang statements. However, you can work around this limitation and make *Sherpa* execute S−Lang scripts at startup by including calls to `evalfile` in your *Sherpa* resource file.

The `evalfile` function takes the name of a S−Lang script as its argument and executes the script. The script may contain any valid S−Lang code, including function definitions and multi−line statements.

For example, if `$HOME/myblackbody.sl` contains the `slang_blackbody()` function from the examples section of the S−Lang usermodel help file, then the following resource file will load it and make the usermodel available to *Sherpa*:

```
unix% more sherparc2.shp
# find out the location of $HOME
variable home = getenv("HOME")
# load the S-Lang usermodel
() = evalfile( home + "/myblackbody.sl" )
# register the model (this call must all be on one line)
() = register_model("slang_blackbody",["kT","ampl"],1,[1.0,1.0],[0.001,0.0],[100.0,1.e10],[1,1])
print("Loaded and registered slang_blackbody (S-lang usermodel)")
```

To try this example, download the files `sherparc2.shp` and `myblackbody.sl` to your home directory, and use them as follows:

```
unix% cd $HOME
unix% mv sherparc2.shp .sherparc
unix% sherpa
...
    Abundances set to Anders & Grevesse
Loaded and registered slang_blackbody (S-lang usermodel)

sherpa> paramprompt off
Model parameter prompting is off
sherpa> slang_blackbody[mybb]
sherpa> show mybb
slang_blackbody[mybb]  (integrate: off)
    Param   Type      Value        Min        Max                Units
    -----   ----      -----        ---        ---                -----
 1     kT thawed          1      1e-03        100
 2   ampl thawed          1          0      1e+10
```

The usermodel `slang_blackbody` is now available automatically at startup.

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/sherparc/ Last modified: 1 Dec 2006