URL: http://cxc.harvard.edu/ciao3.4/read.html
Last modified: December 2006

AHELP for CIAO 3.4

read

Context: sherpa

Jump to: Description Examples Bugs See Also

Synopsis

Inputs the contents of one or more files.

Syntax

```
sherpa> READ <arg> [# [ID]] <filespec> [,[# [ID]] <filespec>,...]

where # specifies the number of the dataset to be associated with the
data file (default dataset number is 1). The ID modifier is used only
when background data are input, and then if and only if the Sherpa
state object variable multiback is set to 1, i.e., if more than one
background dataset is to be associated with a single source dataset.
The ID modifier may be any unreserved string (e.g., A, foo, etc.),
i.e., a string that is not a parsable command.
```

Description

<arg> may be:

READ Command Arguments

Argument	To input a file containing:
{DATA | BACK}	Source|background data values
{ERRORS | BERRORS}	Estimated total errors for the source|background data
{SYSERRORS | BSYSERRORS}	Systematic errors for the source|background data
{WEIGHT | BWEIGHT}	Statistic weight values assigned to each source|background data point
{FILTER | BFILTER}	Mask values (0|1) for each source|background data point
{GROUP | BGROUP}	Grouping values (1|−1) associated with each source|background data point
{QUALITY | BQUALITY}	Quality values (0|2|5) associated with each source|background data point
MDL	A model descriptor list file.

Sherpa currently supports the following file types:

READ File Type Arguments

<filetype>	Containing:
ASCII	ASCII data
FITS	FITS image data
FITSIMAGE	FITS image data
FITSBIN	FITS binary table data
IMH	IRAF image data
PHA	Pulse−height amplitude data

For each of these file types, we discuss the allowed <filespec> arguments in turn.

ASCII File Type:

<filename> [{ASCII | HISTOGRAM}] [<colnumbers>]

where

<filename>	The name of the data file (this may also include a path).
<colnumbers>	A list of column numbers.

The modifier ASCII indicates that the input data is to be treated as unbinned (i.e., models are to be evaluated at single points), while HISTOGRAM leads Sherpa to create a binned dataset from the ASCII file. This is done by treating the first data point as the lower bin boundary for the first channel, the second data point as the upper bin boundary for the first channel and the lower bin boundary for the second channel, etc. The last data point (which has no defined upper boundary in this scheme) is dropped. This command is particularly useful when using additive XSPEC models, since these models are always integrated over a bin and so require binned data.

Note that when reading an ASCII file containing more than two columns, only the data in the first two columns are input. To read other columns, or more than two columns, specify the column numbers with <colnumbers>. See the examples below.

Also note that the last column input is considered the dependent coordinate.

FITS, FITSIMAGE, IMH, and QP File Types:

["]<filename>[<virtual_file_syntax>"] [{FITS | FITSIMAGE | IMH | QP}]

where

<filename>	The name of the data file (this may also include a path).
<virtual_file_syntax>	A filtering and/or binning command argument. See the "Using Data Model Filters" section for further information.

If the command argument {FITS | FITSIMAGE} is not included when reading a FITS file, Sherpa will attempt to determine the FITS file type (e.g., 2−D FITS image vs. 1−D FITS binary table) from the FITS header keywords.

Note that whenever <virtual_file_syntax> is specified, <filename><virtual_file_syntax> usually must be surrounded by quotes, " ".

FITSBIN (FITS Binary Table) File Type:

"<filename><virtual_file_syntax>" [FITSBIN]

where

<filename>	The name of the data file (this may also include a path).
<virtual_file_syntax>	A filtering and/or binning command argument. See the "Using Data Model Filters" section for further information.

Note that <virtual_file_syntax> should be included, in order to specify the desired columns. Otherwise, Sherpa will try to input data from all columns, which will lead to an error message if there are more than two columns.

Also note that if the command argument FITSBIN is not included, Sherpa will attempt to determine the FITS file type (e.g., 2−D FITS image vs. 1−D FITS binary table) from the FITS header keywords.

Last, note that whenever <virtual_file_syntax> is specified, <filename><virtual_file_syntax> usually must be surrounded by quotes, " ".

PHA File Types (Types I and II):

<filename> [PHA]

where

<filename>	The name of the data file (this may also include a path).

Note that if a FITS binary table is input with no <filetype> modifier [FITSBIN or PHA], the file type will be automatically inferred; if the file contains an extension named SPECTRUM, it is assumed to be a PHA file.

If the input PHA file contains a GROUPING column, the data are automatically grouped. Also, Sherpa retains the contents of the QUALITY columns, allowing the user to, e.g., filter out bad channels by issuing the command IGNORE BAD.

If the input PHA file contains a STAT_ERR column, its contents are ignored (and a message printed to the screen). If you wish to use these statistical error estimates (as opposed to letting Sherpa estimate the errors given a chosen STATISTIC, you should read them in as follows (substituting BERRORS if appropriate):

```
sherpa> READ ERRORS "<filename>[cols CHANNEL,STAT_ERR] FITSBIN
```

On the other hand, if the input PHA file contains a SYS_ERR column, its contents are used; see SYSERRORS for more information.

If the STAT_ERR is read in before a fit, and SYS_ERR was also read in, then the error in a bin is sqrt(STAT_ERR^2+SYS_ERR^2).

If STAT_ERR is not read in, and SYS_ERR is read in, then the error in a bin is sqrt([computed Poisson error]^2+SYS_ERR^2).

If the header of the input PHA file includes keywords that references background data files, or source and/or

background response files, then these files and automatically read in and, if appropriate, RSP instrument models are automatically defined (See INSTRUMENT command for more information). When these files are read in, the following kinds of messages are issued:

```
RMF is being input from:
  <directory_path>/example.rmf
ARF is being input from:
  <directory_path>/example.arf
Background data are being input from:
  <directory_path>/example_bkg.pha
Background RMF is being input from:
  <directory_path>/example_bkg.rmf
Background ARF is being input from:
  <directory_path>/example_bkg.arf
```

Note the following, for all file types:

- Beginning with Sherpa version 3.0.2, it is no longer required that multi−dimensional data be defined on a uniform rectangular grid (e.g., such as is the case in a FITS image). The user can, for instance, input amplitudes at arbitrary coordinates and fit models to these amplitudes. (However, the analysis of arbitrarily placed binned [histogramed] data is not supported.)
- Background datasets and errors, source dataset errors, and filters, read in prior to the entry of source data are erased when the source data are read in.
- The data input from ASCII and FITS files can be of arbitrary dimensionality (e.g., the user can read four columns from an ASCII file to create a 3−D dataset, or can read a 1−D FITS image, etc.).
- One uses READ FILTER to read in filter values for each bin or pixel, and not to read in, for instance, region descriptors. These filter values should be either 1 for inclusion, or 0 for exclusion, of the data point. See the example below, as well as the descriptions of the related commands IGNORE and NOTICE.
- To use input data as a model, do not use the READ command; rather, use the GRIDMODEL model.
- Inputting files of type QP and IMH is not supported for non−Solaris platforms.

Using Data Model Filters

This command is able to take any Data Model virtual file specification (see "ahelp dmsyntax"). If you can do

```
unix% dmcopy "infile.fits[spec 1][spec 2]" outfile.fits
```

you can also do

```
sherpa> read "infile.fits[spec 1][spec 2]"
```

This is especially useful when working with very large files. For example:

```
sherpa> read "evt.fits[bin sky=4][opt mem=100]"
```

bins the event file by a factor of four and allocates additional memory. A similar command (omitting the binning factor) can be used to read in an image.

Example 1

Input an ASCII data file having a .dat extension name:

```
sherpa> READ DATA 1 example.dat ASCII 1 2
```

Reads the first two columns of the ASCII data file example.dat, as dataset number 1. The following commands are each equivalent to the above command:

```
sherpa> READ DATA 1 example.dat ASCII
sherpa> READ DATA 1 example.dat
sherpa> READ DATA example.dat
sherpa> DATA example.dat
```

Note that, if not specified, only the first two columns are read. Also, the dataset number is assumed to be 1 if it is not specified.

Example 2

Input ASCII data and error files not having a .dat extension name:

```
sherpa> READ DATA 1 example.qdp ASCII 1 2
sherpa> READ ERRORS 1 example.qdp ASCII 1 3
```

The first command reads columns 1 and 2 of the ASCII data file example.qdp, as dataset number 1. Then, columns 1 and 3 of the same ASCII data file are read, as the measurement errors of this dataset. Note that using the ASCII argument is no longer necessary for input of files not having a .dat extension. Thus, the following READ DATA commands are each equivalent to the above READ command:

```
sherpa> READ DATA 1 example.qdp ASCII
sherpa> READ DATA 1 example.qdp
sherpa> READ DATA example.qdp
sherpa> DATA example.qdp
```

Example 3

Input various data columns from ASCII data and error files:

```
sherpa> READ DATA 1 example.dat 3 8
sherpa> READ ERRORS 1 example.dat 3 5
sherpa> READ DATA 2 example.dat
sherpa> READ ERRORS 2 example.dat 1 4
```

The first command reads columns 3 and 8 of the ASCII data file example.dat, as dataset number 1. Next, the measurement errors for dataset 1, from columns 3 and 5 of example.dat are read. Then, the first and second columns of the ASCII data file example.dat, as dataset number 2, are read. The last command reads the measurement errors for dataset 2, from columns 1 and 4 of example.dat.

Example 4

Overwrite dataset number 1:

```
sherpa> READ DATA 1 example1.dat 3 8
sherpa> READ DATA example2.dat
```

Note that the command READ DATA example2.dat overwrites the data that had been input from example1.dat.

Example 5

Input multiple ASCII datasets, using a single command:

```
sherpa> READ DATA 1 example1.dat, 2 example2.dat, 3 example3.dat 2 3
```

This example illustrates the input of multiple data files simultaneously. The command reads example1.dat as dataset number 1, and example2.dat as dataset number 2. Columns 2 and 3 of example3.dat are read as dataset number 3. The following command is equivalent:

```
sherpa> READ DATA example1.dat, example2.dat, example3.dat 2 3
```

Example 6

Input ASCII data, and weight assignments from a file:

```
sherpa> READ DATA example1.dat 1 2
sherpa> READ WEIGHT example1.dat 1 3
sherpa> SHOW WEIGHTS
```

The first command reads columns 1 and 2 of the ASCII data file example1.dat, as dataset number 1. Column 3 of example1.dat contains a weight assignment for each of the data points. These weight assignments are input with the second command. Current weight assignments for each data point can be reported with the command SHOW WEIGHTS.

Example 7

Input ASCII data, and filter assignments from a file:

```
sherpa> READ DATA example1.dat 1 2
sherpa> READ FILTER example1.dat 1 3
sherpa> SHOW FILTER
```

The first command reads columns 1 and 2 of the ASCII data file example1.dat, as dataset number 1. Column 3 of example1.dat contains a filter assignment for each of the data points (1 for the data point to be included; 0 for the data point to be excluded). Current filter assignments for each data point can be reported with the command SHOW FILTER.

Example 8

Compare input ASCII data to input HISTOGRAM data:

```
sherpa> READ DATA 1 data/example.dat
sherpa> SHOW DATA 1
Y Column: Counts
  Dimensions: 1
  Total Size: 4 bins (or pixels)
  Axis: 0; Name: Bin
  Length: 4 bins (or pixels)
File Name: data/example.dat
SubSection (if any):
File Type: ASCII
```

```
[1] = 1
[2] = 5
[3] = 8
[4] = 17

sherpa> READ DATA 2 data/example.dat HISTOGRAM
sherpa> SHOW DATA 2
Y Column: Counts
  Dimensions: 1
  Total Size: 3 bins (or pixels)
  Axis: 0; Name: Bin
  Length: 3 bins (or pixels)
File Name: data/example.dat
SubSection (if any):
File Type:
[1.500000] = 1
[2.500000] = 5
[3.500000] = 8
```

Example 9

Utilize the HISTOGRAM argument to input binned data:

```
sherpa> READ DATA 1 data/spectrum_notintegrated.dat ASCII
```

The above command inputs data from an ASCII file that has two columns: energy (in keV), and flux (in photons/cm^2/sec/keV). Note that this dataset cannot be used with additive XSPEC models, since they require binned data.

```
sherpa> READ DATA 2 data/spectrum_integrated.dat HISTOGRAM
```

The above command inputs and bins data from an ASCII file that has two columns: energy (in keV), and flux (in photons/cm^2/sec). (i.e., where the second column contains data of units photons/cm^2/sec/keV multiplied by the bin width in keV). Note that this dataset can be used with XSPEC models, since the input data are binned.

Example 10

Input a 2−D FITS image data file:

```
sherpa> READ DATA 1 example_img.fits FITS
```

This command reads the 2−D FITS image example_img.fits as dataset number 1. The following command is equivalent:

```
sherpa> READ DATA 1 example_img.fits FITSIMAGE
```

The following commands are also equivalent to the above, if the example_img.fits file contains the proper header kewords identifying the file as a 2−D FITS image:

```
sherpa> READ DATA example_img.fits
sherpa> DATA example_img.fits
```

Example 11

Input 2−D FITS image data and background files:

```
sherpa> READ DATA 3 example_img.fits FITSIMAGE
sherpa> READ BACK 3 example_img_bkg.fits FITSIMAGE
```

First, the FITS image example_img.fits, as dataset number 3, is read. Then, the background FITS image for this dataset is read. Note that the command SUBTRACT must be issued in order to actually have the background subtracted from the data.

Example 12

Input a portion of a 2−D FITS image data file:

```
sherpa> READ DATA "example_img.fits[#1=100:200, #2=100:400]" FITS
```

This command reads the specified portion of the 2−D FITS image example_img.fits. Note that, by default, the data is taken from the first FITS block for which NAXIS is nonzero. The following command is equivalent:

```
sherpa> READ DATA "example_img.fits[100:200,100:400]" FITS
```

Example 13

Input various data columns from a FITS binary data file:

```
sherpa> READ DATA 1 "example_bin.fits[2][columns #1, #2]" FITSBIN
```

This command reads the first two columns from the second extension of the FITS binary table file example_bin.fits, as dataset number 1. Note that column numbers or names must always be specified when reading FITS binary table files. The following commands are each equivalent to the above command:

```
sherpa> READ DATA 1 "example_bin.fits[2][cols #1, #2]" FITSBIN
sherpa> DATA "example_bin.fits[2][cols #1, #2]" FITSBIN
```

Example 14

Input various data columns from FITS binary data files:

```
sherpa> READ DATA 2 "example_bin.fits[2][cols TIME, EXPNO]" FITSBIN
sherpa> READ DATA 3 "example_bin.fits[EVENTS][cols time, expno]" FITSBIN
```

The first command reads columns time and expno, from the second extension of the FITS binary table file example_bin.fits, as dataset number 2. The second command reads columns time and expno, from the EVENTS extension of the FITS binary table file example_bin.fits, as dataset number 3. Note that columns may be specified by case insensitive name. Also, the FITS extension can be specified by either the number or the name of the extension (in this example, the name of the second extension is EVENTS).

Example 15

Input a 2−D image by binning columns from a FITS binary data file:

```
sherpa> DATA "example_bin.fits[bin chipx, chipy]" FITSIMAGE
Warning: Could not retrieve WCS coord descriptor
```

This command inputs into Sherpa the FITS binary table example_bin.fits, but bins the table to create an image using the chipx and chipy columns. Note that the FITSIMAGE argument is required since it is ultimately a FITS image that is being input to Sherpa.

Example 16

Input a 2−D image by binning and filtering columns from a FITS binary data file:

```
sherpa> DATA "example_bin.fits[bin chipx=200:400:4, chipy=300:400:4]" FITSIMAGE
```

This command creates and inputs into Sherpa, an image using the chipx and chipy columns of the binary FITS table file example_bin.fits. In this example, ranges for the axes, and bin sizes, are given. Note that the FITSIMAGE argument is required.

Example 17

Input a 2−D image data file:

```
sherpa> READ DATA 1 example.imh IMH
```

This command reads the image example.imh, as dataset number 1. The following commands are each equivalent:

```
sherpa> READ DATA example.imh IMH
sherpa> READ DATA example.imh
sherpa> DATA example.imh
```

Example 18

Input a portion of a 2−D image data file:

```
sherpa> READ DATA 2 "example.imh[#1=100:200, #2=100:400]"
```

This command reads a portion of the image example.imh, from Axis 0 coordinates 100 to 200 and from Axis 1 coordinates 100 to 400. The data are read as dataset number 2. The following command is equivalent:

```
sherpa> READ DATA 2 "example.imh[100:200, 100:400]"
```

Example 19

Input PHA data and background files:

```
sherpa> READ DATA 4 example.pha PHA
The inferred file type is PHA.  If this is not what you want, please
specify the type explicitly in the data command.
WARNING: using systematic errors specified in the PHA file.
RMF is being input from:
  <directory_path>/example2.rmf
ARF is being input from:
  <directory_path>/example2.arf
Background data are being input from:
  <directory_path>/example2_bkg.pha
sherpa> READ BACK 4 example_bkg.pha
The inferred file type is PHA.  If this is not what you want, please
specify the type explicitly in the data command.
```

First, the PHA data file example.pha is read as dataset number 4. Note that the systematic errors contained in the PHA data file are input. These input systematic errors are added in quadrature with the statistical errors (which are automatically computed using the currently defined STATISTIC). Note also that since the header of the PHA data file contains the proper keywords, instrument and background data files are automatically loaded. The READ

BACK 4 example_bkg.pha command inputs the background PHA file is read for dataset number 4.

Example 20

Input multiple PHA data files:

```
sherpa> READ DATA example1.pha PHA, example2.pha
The inferred file type is PHA.  If this is not what you want, please
specify the type explicitly in the data command.
WARNING: using systematic errors specified in the PHA file.
The inferred file type is PHA.  If this is not what you want, please
specify the type explicitly in the data command.
WARNING: using systematic errors specified in the PHA file.
```

This example illustrates the input of multiple data files simultaneously. The command reads example1.dat as dataset number 1, and example2.dat as dataset number 2. Again, note that the statistical errors in the PHA data files are not input, but the systematic errors are input.

Bugs

See the Sherpa bug pages online for an up−to−date listing of known bugs.

See Also

chandra

guide

sherpa

autoest, back, berrors, bsyserrors, coord, data, dataspace, fakeit, feffile, group, guess, is_subtracted, load, load_arf, load_ascii, load_back_from, load_backset, load_dataset, load_fitsbin, load_image, load_inst, load_inst_from, load_pha, load_pha2, load_rmf, set_analysis, set_axes, set_backscale, set_coord, set_data, set_exptime, set_subtract, set_weights, setback, setdata, subtract, ungroup, unsubtract, use

---

URL: http://cxc.harvard.edu/ciao3.4/read.html
Last modified: December 2006