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

AHELP for CIAO 3.4

get_par

Context: sherpa

Jump to: Description Examples Bugs See Also

Synopsis

Module function for getting model parameter values, etc.

Syntax

```
Array_Type get_par([String_Type])

Error Return Values: NULL

Argument:

(1) Model name, or model parameter name (default all model parameters)
```

Description

This function retrieves an array of structures, each of which contains information about a defined model parameter. A typical structure looks as follows:

```
sherpa> GAUSS[g]
sherpa> foo = get_par("g")
sherpa> print(foo[0])
name             =  g.fwhm
model            =  gauss1d
type             =  src
value            =  10
min              =  2.22507e-308
max              =  1.79769e+308
delta            =  -1
units            =  NULL
frozen           =  0
linked           =  0
linkexpr         =  NULL
```

The fields of the structure are:

get_par Structure Fields

Argument	Description
name	the name of the parameter
model	the model that the parameter belongs to
type	source/background−type (src) or instrument−type (inst) model
value	the parameter value (either a number, or a string filename)
min	the current (soft) lower bound on allowed parameter values
max	the current (soft) upper bound on allowed parameter values
delta	the initial step size for the parameter in fits (or −1 to use the default step size)
units	parameter units, if known/appropriate
frozen	if 1, the parameter value is frozen; if 0, it is thawed
link	if 1, the parameter's value is linked to that of other parameter(s); if 0, it is not linked
linkexpr	if the parameter is linked to other parameters, the expression showing how it is linked

See the Sherpa command CREATE for more information.

Example 1

Get a parameter structure; change two fields; set back into Sherpa:

```
sherpa> GAUSS[g]
sherpa> foo = get_par("g.pos")
sherpa> print(foo)
name             =  g.pos
model            =  gauss1d
type             =  src
value            =  0
min              =  −3.40282e+38
max              =  3.40282e+38
delta            =  −1
units            =  NULL
frozen           =  0
linked           =  0
linkexpr         =  NULL
sherpa> foo.value = 15.5
sherpa> foo.min = 0.0
sherpa> () = set_par(foo)
sherpa> SHOW g
gauss1d[g]  (integrate: on)
    Param   Type      Value        Min        Max                Units
    -----   ----      -----        ---        ---                -----
 1   fwhm thawed         10 1.1755e−38 3.4028e+38
 2    pos thawed       15.5          0 3.4028e+38
 3   ampl thawed          1 −3.403e+38 3.4028e+38
```

Example 2

Here we loop through all the defined parameters and display their name and current value:

```
sherpa> erase all
sherpa> paramprompt off
```

```
sherpa> xsmekal[gal]
sherpa> xsphabs[abs]
sherpa> ps = get_par()
sherpa> ps
Struct_Type[7]
sherpa> foreach(ps){p=();vmessage("Par %-14s = %g",p.name,p.value);}
Par gal.kT         = 1
Par gal.nH         = 1
Par gal.Abund      = 1
Par gal.Redshift   = 0
Par gal.Switch     = 1
Par gal.norm       = 1
Par abs.nH         = 0.1
```

The initial set of lines are to set up two models ("gal", which is an XSMEKAL model, and "abs", which is an XSPHABS model) with default parameter values. The ps variable is set here to an array of 7 structures, which we loop through using the S−Lang foreach function. Each member of the array is looped through, and stored in the variable p. This variable is then used to get the parameters name and value (p.name and p.value respectively). The whole 'foreach' statement must be on one line since Sherpa does not allow multi−line S−Lang statements. This restriction does not apply to code in a file executed via evalfile().

Example 3

Here we use a small S−Lang function we have written to display parameter values. If the file print_pars.sl contains:

```
define print_pars() {
  variable pars = get_par();
  if ( NULL == pars ) {
    print( "No parameter values found!" );
    return;
  }

  vmessage( "# Name                model  frozen  value" );
  foreach ( pars ) {
    variable par = ();
    vmessage( "%-20s  %-10s  %d  %g",
      par.name, par.model, par.frozen, par.value
    );
  }
}
```

then you can say:

```
sherpa> () = evalfile("print_pars.sl")
sherpa> erase all
sherpa> paramprompt off
sherpa> xsmekal[gal]
sherpa> xsphabs[abs]
sherpa> print_pars
# Name                model  frozen  value
gal.kT                xsmekal     0  1
gal.nH                xsmekal     1  1
gal.Abund             xsmekal     1  1
gal.Redshift          xsmekal     1  0
gal.Switch            xsmekal     1  1
gal.norm              xsmekal     0  1
abs.nH                xsphabs     0  0.1
```

although you may find either list_par() or SHOW more useful.

Bugs

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

See Also

sherpa

autoest, background, create, create_model, createparamset, fit, freeze, get_defined_models, get_model_params, get_models, get_num_par, get_stackexpr, getx, gety, guess, instrument, integrate, is_paramset, jointmode, kernel, lineid, linkparam, mdl, modelexpr, modelstack, nestedmodel, noise, paramprompt, paramset, pileup, rename, run_fit, set_par, set_paramset, set_stackexpr, source, thaw, truncate, unlink

---

The Chandra X−Ray Center (CXC) is operated for NASA by the Smithsonian Astrophysical Observatory.
60 Garden Street, Cambridge, MA 02138 USA.
Smithsonian Institution, Copyright © 1998−2006. All rights reserved.

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