Supplement of Geosci. Model Dev., 9, 3093–3110, 2016
http://www.geosci-model-dev.net/9/3093/2016/
doi:10.5194/gmd-9-3093-2016-supplement

Geoscientific
Model Development
Open Access

Supplement of

Community Intercomparison Suite (CIS) v1.4.0: a tool for intercomparing models and observations

Duncan Watson-Parris et al.

Correspondence to: Duncan Watson-Parris (duncan.watson-parris@physics.ox.ac.uk)

CIS Reference Card

Community Intercomparison Suite v1.4.0
August 2016 - www.cistools.net

Global options

Some options apply to all commands:

-o	Specify an output filename.
?	Print help about the command
-q	Quiet – suppress all output to screen
-v	Verbose – include more detailed screen output

Commands

The basic structure of a CIS command is:

```
$ cis <command> <variable>:<file>:<options> <options>
```

The <data variable>:<data file> construct is common to all commands and is generally referred to as a `datagroup'. A datagroup can contain multiple, comma separated, variables, and filenames, and both can include wildcards. All datagroups accept the **product=`…'** option to specify the plugin to read the data with.

All currently available commands are summarised below.

version

Will print the version of CIS currently being used.

subset <data variable>:<data file> <subset limits>

Subset any number of datasets to the specified spatio-temporal extent specified by a comma separated sequence of one or more coordinate ranges of the form **variable=[start,end]**. The variable can be its name as it is in the data file, its CF standard name, or an axes name shorthand such as x, y, t, z and p.

aggregate <data variable>:<data file>:<options> <aggregation grid>

Aggregate a dataset over one or more coordinates, either completely, or to a user defined grid. Multiple coordinates should be comma separated, grids are defined using **coordinate=[start,end,step]**.

col <data variable>:<data file> <sample file>:<colocation method>

Collocate the data variable onto the spatio-temporal sampling provided by the sample file. Available collocation methods depend on the data-types:

Sample

Source	GRIDDED	UNGRIDDED
GRIDDED	**lin**, nn, box	**lin**, nn
UNGRIDDED	**box**, bin	**box**

Where the default options are shown in bold and described below.

lin	linear interpolation in space and time
nn	Nearest Neighbour
bin	operates *kernel* on all *source* data in *sample* bounds
box	operates *kernel* on all *source* data in user-defined box on *sample*

eval <data variable>:<data file> <expression> <units>

Perform an evaluation over correlated data variables of any valid Python expression. Aliases can be used to simplify evaluation expressions as shown in the recipes.

stats <data variable>:<data file>

Calculate statistics and correlation factors between exactly two correlated datasets.

info <data file> [-v <data variable>]

Print the variables present in a file, or set of files, and optionally summary information fo one or more specific variables.

plot <data variable>:<data file>:<layer options> [plot options]

Plot any CIS compatible data using a wide variety of plot types and options. Each layer corresponds to a single variable, so only one variable is allowed per datagroup.

Layer options

Multiple layer options should be specified in a comma separated list.

color	Colour of markers, e.g. for scatter plot points or contour lines
cmap	Colour map to use, e.g. for contour lines or heatmap
cmin	The minimum value for the colourmap
cmax	The maximum value for the colourmap
edgecolor	Colour of scatter marker edges
itemstyle	Shape of scatter marker
label	Name of datagroup for the legend
contnlevels	Additional datagroup options for contour plots only:
contlevels	The number of levels for the contour plot

contlabel	A list of levels for the contour plot, e.g. contlevels=[0,1,3,10]
contwidth	Options are true or false, if true then contour labels are shown
contfontsize	Width of the contour lines
type	The type of plot for this particular layer (only used for overlay plots)

Plot options

--type	The plot type, one of: **line**, **scatter**, **heatmap**, **contour**, **contourf**, **histogram2d**, **histogram3d**, **comparativescatter** and **overlay**
--xlabel	The label for the x axis
--ylabel	The label for the y axis
--cbarlabel	The label for the colour bar
--xtickangle	The angle for the ticks on the x axis
--ytickangle	The angle for the ticks on the y axis
--title	The title of the plot
--itemwidth	The width of an item. Unit are points in the case of a line, and points squared in the case of a scatter point
--fontsize	The size of the font in points
--cmap	The colour map to be used when plotting a 3D plot
--height	The height of the plot, in inches
--width	The width of the plot, in inches
--xbinwidth	The width of the histogram bins on the x axis
--ybinwidth	The width of the histogram bins on the y axis
--cbarorient	The orientation of the colour bar, either horizontal or vertical
--nocolourbar	Hides the colour bar on a 3D plot
--grid	Shows grid lines
--plotwidth	Width of the plot in inches
--plotheight	Height of the plot in inches
--cbarscale	Used to change the size of the colour bar when plotting, defaults to 0.55 for vertical colour bars, 1.0 for horizontal.
--coastlinescolour	The colour of the coastlines on a map
--nasabluemarble	Use NASA Blue Marble for the background
--logx	The x axis will be plotted using a log scale of base 10
--logy	The y axis will be plotted using a log scale of base 10
--logv	The values (colours) will be plotted using a log scale of base 10

Also, the arguments **--xmin**, **--xmax**, **--xstep**, **--ymin**, **--ymax**, **--ystep**, **--vmin**, **--vmax** and **--vstep** can be used to specify the range of values to plot on each axis.

Recipes

There are some useful command structures which may not be immediately obvious, some of those are listed here.

Reading model fields with ancillary orography files

Many model outputs include an orography in a separate ancillary file, assuming CF-compliant standard names, these can easily be included to produce correct altitude fields

```
$ cis plot aot500aer:aot500aer.nc,orography.nc
```

Plotting multiple variables

Different plot types handle plotting multiple variables slightly differently, which may lead to some confusion.
For plotting multiple variables in line and scatter plots, simply use as many datagroups as is necessary, applying layer options as needed:

```
$ cis plot CCN1:file.nc:label='CCN at 1%' CCN2:file.nc:label='CCN at 2%'
```

For plotting multiple variables *over* each other, on their own scales, use an overlay plot:

```
$ cis plot CCN1:file.nc:type=heatmap CCN2:file.nc:type=contour
        --type=overlay
```

Calculating the difference between two datasets

A useful demonstration of the use of eval is in calculating the difference between two collocated files:

```
$ cis  eval  var1=a:obs.nc  var2=b:collocated_model.nc "a-b" 1
        -o var_diff:obs_minus_model.nc
```

Notice here we have used aliases for the variable names, and also specified the output variable name (this only works for the eval command currently)

Masking values based on a second variable

A more advanced example of eval is masking values from one variable based on the value of a second variable. For example only including the mean values if the number of points used for that mean was more than 20:

```
$ cis eval ccn,ccn_num_points=num:collocated_ccn.nc
       "numpy.ma.masked_where(num < 20, ccn)" 1 -o nc:masked.nc
```

CIS as a Python Library

Using CIS to read datasets into Python, and even convert to Pandas, is straightforward:

```
from cis import read_data
data = read_data('my_file.nc','my_var')
df = data.as_data_frame()
```