Science & Technology Facilities Council | UK Research and Innovation | NERC SCIENCE OF THE ENVIRONMENT

Python data analysis

cf-python and cf-plot

What is cf-python?

cf-python is an implementation of the CF data model that:

- Reads CF-netCDF and PP format files, aggregating contents into as few multi-dimensional fields as possible.
- Writes fields to CF-netCDF files on disk.
- Creates, deletes and modifies field data and metadata.
- Subsets fields by conditions on their metadata.
- Subspaces a field to create a new field.
- Enables arithmetic/comparison operations with fields.
- Calculates statistics on field data.

Documentation

cf-python 2.2.8 documentation

Note: For version 1.x documentation, see the documentation archive.

Warning: Incompatible differences between versions 1.x and 2.x

https://cfpython.bitbucket.io/docs/latest/index.html

Main concept - the "field"

The `cf` package allows a data array and its associated metadata to be contained and manipulated as a single entity called a *field*, which is stored in a `cf.Field` object.

Some example functionality

Here we will highlight some example cf-python functionality that goes beyond that provided by lower level packages:

Reading data from multiple files:

```
>>> import cf
>>> f = cf.read('~/file.nc')
>>> f = cf.read('file[1-9a-c].nc')
>>> f = cf.read('dir*/*.pp')
>>> f = cf.read(['file1.nc', 'file2.nc',
                 'file3*.nc'])
```

Selecting from a field

Fields may be selected with the `match` and `select` methods. These methods take conditions on field CF properties, attributes and coordinates as inputs:

```
>>> f
[<CF Field: x_wind(grid_latitude(110), grid_longitude(106)) m
s-1>,
 <CF Field: air_temperature(time(12), grid_latitude(73),
grid_longitude(96)) K>]

>>> f.match('temperature', regex=True)
[False, True]
>>> g = f.select('air_temperature', cvalue={'longitude': 0})
>>> g
[<CF Field: air_temperature(time(12), grid_latitude(73),
grid_longitude(96)) K>]
```

Functions of the cf module

The cf module provides a variety of functions, including:

- **I/O:** `read, write, open_files`
- **Aggregation:** `aggregate`
- **Statistics:** `collapse`
- **Comparison:** `eq, gt, lt, ...`
  For climatologies: `djf, mam, jja, son`
- **Date-time:** `dt, Y, M, D`

Command-line tools

cfplot provides some useful command-line utilities:

The **`cfdump`** tool generates text representations on standard output of the CF fields contained in the input files.

The **`cfa`** tool creates aggregated CF datasets - it creates and writes to disk the CF fields contained in the input files.

For usage instructions, use the -h option to display the manual pages:

cfa example

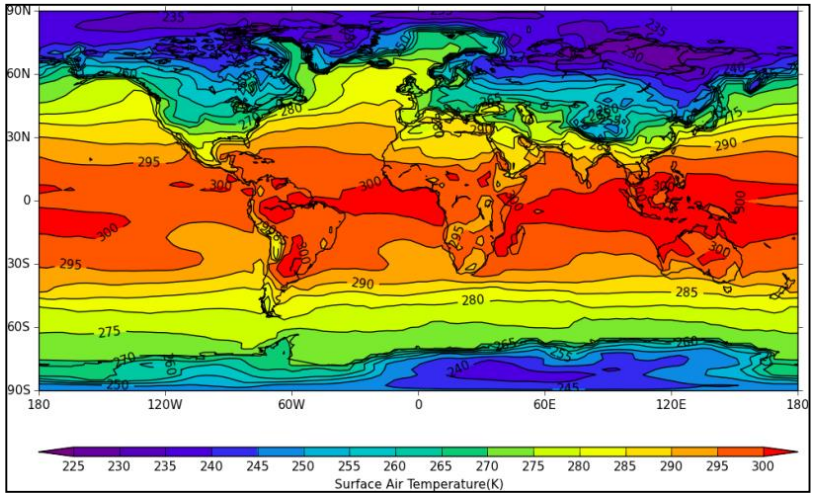
cfa can read multiple files and aggregate the contents into a single output file, e.g.:

```
$ cfa -o out.nc file1.nc file2.nc

$ cfa -o out.nc file[1-9].nc

$ cfa -f NETCDF3_CLASSIC -o out.nc data1/*.nc
    data2/*.nc

$ cfa -o out.nc
     http://test.opendap.org/dap/coads_climat
     ology.nc file*.nc    # remote file(s)
```

Plotting with cfplot

cfplot is a set of Python routines for making the common contour and vector plots that climate researchers use. The data to make a contour plot can be passed to **cfplot** using **cf-python** as per the following example.

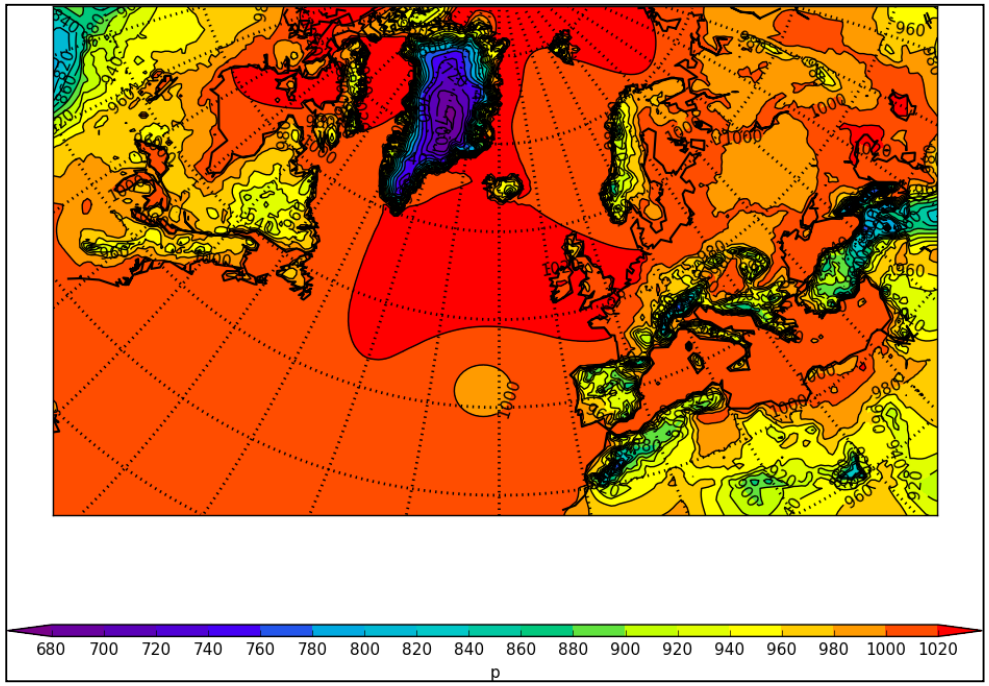
```
import cf, cfplot as cfp
f = cf.read('/opt/graphics/cfplot_data/tas_A1.nc')[0]
cfp.con(f.subspace(time=15))
```



Plotting with cfplot

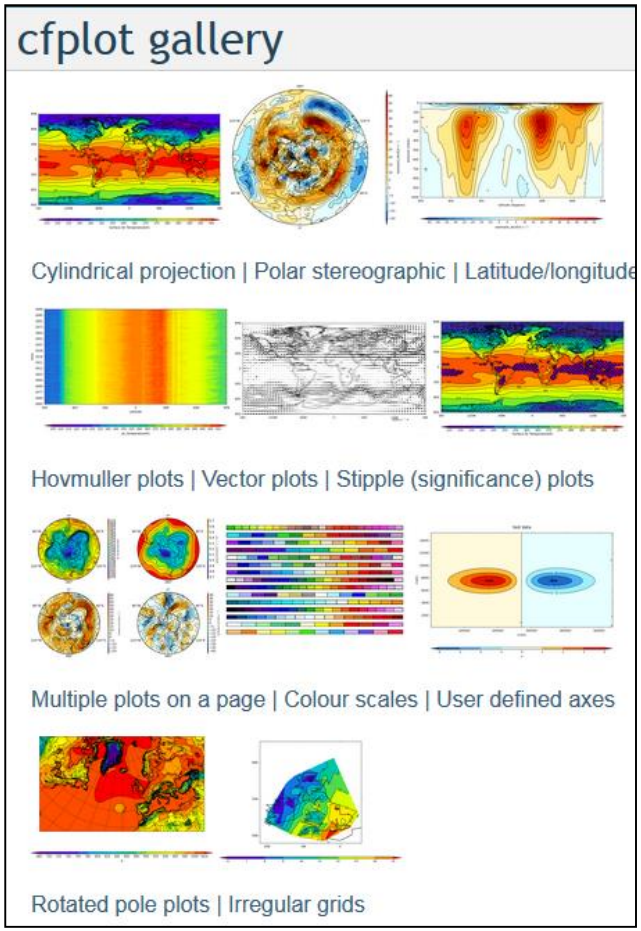
Plotting rotated pole data.

```
import cf, cfplot as cfp
f = cf.read('/opt/graphics/cfplot_data/rgp.nc')[0]
cfp.con(f)
```



And more

See: http://ajheaps.github.io/cf-plot/gallery.html



Further reading

cf-python documentation (current version):
https://cfpython.bitbucket.io/docs/latest/index.html

cf tools:
http://cms.ncas.ac.uk/wiki/ToolsAndUtilities

cfplot:
http://ajheaps.github.io/cf-plot/