Manipulating data and metadata in cf-python (version 2)

Homepage: https://cfpython.bitbucket.io (https://cfpython.bitbucket.io)

Online, searchable documenation: https://cfpython.bitbucket.io/docs/latest (https://cfpython.bitbucket.io/docs/latest)

Contents:

Already using cf-python? https://cfpython.bitbucket.io/docs/latest/1_to_2_changes.html (https://cfpython.bitbucket.io/docs/latest/1_to_2_changes.html)

In [1]:

```
import cf
cf.__version__
```

Out[1]:

```
'2.1'
```

1. Read, inspect and write files

https://cfpython.bitbucket.io/docs/latest/generated/cf.read.html (https://cfpython.bitbucket.io/docs/latest/generated/cf.read.html)

https://cfpython.bitbucket.io/docs/latest/generated/cf.read_field.html (https://cfpython.bitbucket.io/docs/latest/generated/cf.read_field.html)

In [2]:

```
f = cf.read_field('ncas_data/IPSL-CM5A-LR_r1i1p1_tas_n96_rcp45_mnth.nc')
```

In [3]:

```
f
```

Out[3]:

```
<CF Field: air_temperature(time(120), latitude(145), longitude(192)) K>
```

In [4]:

```
print f
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(120), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(120) = [1959-12-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

In [5]:

```
f.dump()
```

```
----------------------
Field: air_temperature
----------------------
CDI = 'Climate Data Interface version 1.7.0 (http://mpimet.mpg.de/cdi)'
CDO = 'Climate Data Operators version 1.7.0 (http://mpimet.mpg.de/cdo)'
associated_files = 'baseURL: http://cmip-pcmdi.llnl.gov/CMIP5/dataLocation
                    gridspecFile: gridspec_atmos_fx_IPSL-CM5A-
                    LR_historical_r0i0p0.nc areacella: areacella_fx_IPSL-CM5A-
                    LR_historical_r0i0p0.nc'
branch_time = 1850.0
cmor_version = '2.5.1'
comment = 'This 20th century simulation include natural and anthropogenic
          forcings.'
contact = 'ipsl-cmip5 _at_ ipsl.jussieu.fr Data manager : Sebastien Denvil'
creation_date = '2011-02-23T17:52:35Z'
experiment = 'historical'
experiment_id = 'historical'
forcing = 'Nat,Ant,GHG,SA,Oz,LU,SS,Ds,BC,MD,OC,AA'
frequency = 'mon'
history = "Thu May 26 15:47:13 2016: cdo mergetime /data/cr1/hadlg/helix/IPSL-
          CM5A-LR_rcp45_tmp_output_1_hist.nc /data/cr1/hadlg/helix/IPSL-CM5A-
          LR_rcp45_tmp_output_1_fut.nc /data/cr1/hadlg/helix/IPSL-CM5A-
          LR_r1i1p1_tas_merged_rcp45.nc\n2011-06-24T02:32:44Z altered by CMOR:
          Treated scalar dimension: 'height'. 2011-06-24T02:32:44Z altered by
          CMOR: replaced missing value flag (9.96921e+36) with standard
          missing value (1e+20). 2011-06-24T02:32:45Z altered by CMOR:
          Inverted axis: lat."
initialization_method = 1
institute_id = 'IPSL'
institution = 'IPSL (Institut Pierre Simon Laplace, Paris, France)'
long_name = 'Near-Surface Air Temperature'
model_id = 'IPSL-CM5A-LR'
modeling_realm = 'atmos'
original_name = 't2m'
parent_experiment = 'pre-industrial control'
parent_experiment_id = 'piControl'
parent_experiment_rip = 'r1i1p1'
physics_version = 1
product = 'output'
project_id = 'CMIP5'
realization = 1
references = 'Model documentation and further reference available here :
             http://icmc.ipsl.fr'
source = 'IPSL-CM5A-LR (2010) : atmos : LMDZ4 (LMDZ4_v5, 96x95x39); ocean :
         ORCA2 (NEMOV2_3, 2x2L31); seaIce : LIM2 (NEMOV2_3); ocnBgchem :
         PISCES (NEMOV2_3); land : ORCHIDEE (orchidee_1_9_4_AR5)'
standard_name = 'air_temperature'
table_id = 'Table Amon (31 January 2011) 53b766a395ac41696af40aab76a49ae5'
title = 'IPSL-CM5A-LR model output prepared for CMIP5 historical'
tracking_id = '826ee5e9-3cc9-40a6-a42b-d84c6b4aad97'

Domain Axis: height(1)
Domain Axis: time(120)
Domain Axis: latitude(145)
Domain Axis: longitude(192)

Data(time(120), latitude(145), longitude(192)) = [[[244.825790405, ..., 244.526885986]]] K

Cell Method: time: mean (interval: 30 minutes)

Dimension Coordinate: time
    axis = 'T'
    long_name = 'time'
    standard_name = 'time'
    Data(time(120)) = [1959-12-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
    Bounds(time(120), 2) = [[1959-12-01T00:00:00Z, ..., 1969-12-01T00:00:00Z]] 365_day

Dimension Coordinate: latitude
    axis = 'Y'
    long_name = 'latitude'
    standard_name = 'latitude'
    Data(latitude(145)) = [-90.0, ..., 90.0] degrees_north
    Bounds(latitude(145), 2) = [[-90.0, ..., 90.0]] degrees_north

Dimension Coordinate: longitude
    axis = 'X'
```

```
    long_name = 'longitude'
    standard_name = 'longitude'
    Data(longitude(192)) = [0.0, ..., 358.125] degrees_east
    Bounds(longitude(192), 2) = [[-0.9375, ..., 359.0625]] degrees_east

Dimension Coordinate: height
    axis = 'Z'
    long_name = 'height'
    positive = 'up'
    standard_name = 'height'
    Data(height(1)) = [2.0] m
```

Properties

In [6]:

```
f.properties()
```

Out[6]:

```
{'CDI': 'Climate Data Interface version 1.7.0 (http://mpimet.mpg.de/cdi)',
 'CDO': 'Climate Data Operators version 1.7.0 (http://mpimet.mpg.de/cdo)',
 'Conventions': 'CF-1.5',
 '_FillValue': 1.0000000200408773e+20,
 'associated_files': 'baseURL: http://cmip-pcmdi.llnl.gov/CMIP5/dataLocation gridspecFile: grid
spec_atmos_fx_IPSL-CM5A-LR_historical_r0i0p0.nc areacella: areacella_fx_IPSL-CM5A-LR_historical
_r0i0p0.nc',
 'branch_time': 1850.0,
 'cmor_version': '2.5.1',
 'comment': 'This 20th century simulation include natural and anthropogenic forcings.',
 'contact': 'ipsl-cmip5 _at_ ipsl.jussieu.fr Data manager : Sebastien Denvil',
 'creation_date': '2011-02-23T17:52:35Z',
 'experiment': 'historical',
 'experiment_id': 'historical',
 'forcing': 'Nat,Ant,GHG,SA,Oz,LU,SS,Ds,BC,MD,OC,AA',
 'frequency': 'mon',
 'history': "Thu May 26 15:47:13 2016: cdo mergetime /data/cr1/hadlg/helix/IPSL-CM5A-LR_rcp45_t
mp_output_1_hist.nc /data/cr1/hadlg/helix/IPSL-CM5A-LR_rcp45_tmp_output_1_fut.nc /data/cr1/hadl
g/helix/IPSL-CM5A-LR_r1i1p1_tas_merged_rcp45.nc\n2011-06-24T02:32:44Z altered by CMOR: Treated
scalar dimension: 'height'. 2011-06-24T02:32:44Z altered by CMOR: replaced missing value flag (
9.96921e+36) with standard missing value (1e+20). 2011-06-24T02:32:45Z altered by CMOR: Inverte
d axis: lat.",
 'initialization_method': 1,
 'institute_id': 'IPSL',
 'institution': 'IPSL (Institut Pierre Simon Laplace, Paris, France)',
 'long_name': 'Near-Surface Air Temperature',
 'missing_value': 1e+20,
 'model_id': 'IPSL-CM5A-LR',
 'modeling_realm': 'atmos',
 'original_name': 't2m',
 'parent_experiment': 'pre-industrial control',
 'parent_experiment_id': 'piControl',
 'parent_experiment_rip': 'r1i1p1',
 'physics_version': 1,
 'product': 'output',
 'project_id': 'CMIP5',
 'realization': 1,
 'references': 'Model documentation and further reference available here : http://icmc.ipsl.fr'
,
 'source': 'IPSL-CM5A-LR (2010) : atmos : LMDZ4 (LMDZ4_v5, 96x95x39); ocean : ORCA2 (NEMOV2_3,
2x2L31); seaIce : LIM2 (NEMOV2_3); ocnBgchem : PISCES (NEMOV2_3); land : ORCHIDEE (orchidee_1_9
_4_AR5)',
 'standard_name': 'air_temperature',
 'table_id': 'Table Amon (31 January 2011) 53b766a395ac41696af40aab76a49ae5',
 'title': 'IPSL-CM5A-LR model output prepared for CMIP5 historical',
 'tracking_id': '826ee5e9-3cc9-40a6-a42b-d84c6b4aad97'}
```

In [7]:

```
f.getprop('project_id')
```

Out[7]:

```
'CMIP5'
```

In [8]:

```
f.setprop('project_id', 'banana')
f.getprop('project_id')
```

Out[8]:

```
'banana'
```

In [9]:

```
f.delprop('project_id')
f.getprop('project_id')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-9-4caf226c3e8e> in <module>()
      1 f.delprop('project_id')
----> 2 f.getprop('project_id')

/home/david/cf-python/cf/variable.pyc in getprop(self, prop, *default)
   6586
   6587         raise AttributeError("%s doesn't have CF property %r" %
-> 6588                              (self.__class__.__name__, prop))
   6589     #--- End: def
   6590

AttributeError: Field doesn't have CF property 'project_id'
```

Shorthand for named CF properties

http://cfconventions.org/Data/cf-conventions/cf-conventions-1.7/cf-conventions.html#attribute-appendix (http://cfconventions.org/Data/cf-conventions/cf-conventions-1.7/cf-conventions.html#attribute-appendix)

In [10]:

```
print f.standard_name
f.standard_name = 'banana'
print f.standard_name
del f.standard_name
f.standard_name = 'air_temperature'
print f.standard_name
```

```
air_temperature
banana
air_temperature
```

Reading many files

In [11]:

```
fl = cf.read('ncas_data/data[2-7].nc')
fl
```

Out[11]:

```
[<CF Field: air_temperature(long_name:t(1), long_name:p(1), long_name:latitude(256), long_name:
longitude(512)) K>,
 <CF Field: air_temperature(long_name:t(1), long_name:p(1), latitude(160), longitude(320)) K>,
 <CF Field: air_temperature(time(1680), latitude(73), longitude(96)) K>,
 <CF Field: eastward_wind(time(1), pressure(37), latitude(256), longitude(512)) m s**-1>,
 <CF Field: eastward_wind(time(1), pressure(23), latitude(160), longitude(320)) m s**-1>,
 <CF Field: eastward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>,
 <CF Field: northward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>]
```

In [12]:

```
for x in fl:
    print 'NAME:', x.name(), 'SHAPE:', x.shape, 'UNITS:', x.units
```

```
NAME: air_temperature SHAPE: (1, 1, 256, 512) UNITS: K
NAME: air_temperature SHAPE: (1, 1, 160, 320) UNITS: K
NAME: air_temperature SHAPE: (1680, 73, 96) UNITS: K
NAME: eastward_wind SHAPE: (1, 37, 256, 512) UNITS: m s**-1
NAME: eastward_wind SHAPE: (1, 23, 160, 320) UNITS: m s**-1
NAME: eastward_wind SHAPE: (1, 23, 36, 48) UNITS: m s**-1
NAME: northward_wind SHAPE: (1, 23, 36, 48) UNITS: m s**-1
```

Select by list position

In [13]:

```
g = fl[0]
g
```

Out[13]:

```
<CF Field: air_temperature(long_name:t(1), long_name:p(1), long_name:latitude(256), long_name:l
ongitude(512)) K>
```

In [14]:

```
fl[4:]
```

Out[14]:

```
[<CF Field: eastward_wind(time(1), pressure(23), latitude(160), longitude(320)) m s**-1>,
 <CF Field: eastward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>,
 <CF Field: northward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>]
```

Select by metadata

https://cfpython.bitbucket.io/docs/latest/generated/cf.FieldList.select.html (https://cfpython.bitbucket.io/docs/latest/generated/cf.FieldList.select.html)

In [15]:

```
fl.select('air_temperature')
```

Out[15]:

```
[<CF Field: air_temperature(long_name:t(1), long_name:p(1), long_name:latitude(256), long_name:
longitude(512)) K>,
 <CF Field: air_temperature(long_name:t(1), long_name:p(1), latitude(160), longitude(320)) K>,
 <CF Field: air_temperature(time(1680), latitude(73), longitude(96)) K>]
```

In [16]:

```
#help(fl.select)
```

In [17]:

```
fl.select('northward_wind')
```

Out[17]:

```
[<CF Field: northward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>]
```

In [18]:

```
fl.select({'units': 'km h-1'})
```

Out[18]:

```
[<CF Field: eastward_wind(time(1), pressure(37), latitude(256), longitude(512)) m s**-1>,
 <CF Field: eastward_wind(time(1), pressure(23), latitude(160), longitude(320)) m s**-1>,
 <CF Field: eastward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>,
 <CF Field: northward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>]
```

In [19]:

```
fl.select('(east|north)ward_wind')
```

Out[19]:

```
[<CF Field: eastward_wind(time(1), pressure(37), latitude(256), longitude(512)) m s**-1>,
 <CF Field: eastward_wind(time(1), pressure(23), latitude(160), longitude(320)) m s**-1>,
 <CF Field: eastward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>,
 <CF Field: northward_wind(time(1), pressure(23), latitude(36), longitude(48)) m s**-1>]
```

Write fields to a netCDF file

https://cfpython.bitbucket.io/docs/latest/generated/cf.write.html (https://cfpython.bitbucket.io/docs/latest/generated/cf.write.html)

In [20]:

```
cf.write(f, 'new_file.nc')
```

In [21]:

```
g = cf.read_field('new_file.nc')
f.equals(g)
```

Out[21]:

```
True
```

2. Subspace a field

https://cfpython.bitbucket.io/docs/latest/generated/cf.Field.subspace.html
(https://cfpython.bitbucket.io/docs/latest/generated/cf.Field.subspace.html)

Index-space: [square brackets]

In [22]:

```
f.subspace[0, 0, 0]
```

Out[22]:

```
<CF Field: air_temperature(time(1), latitude(1), longitude(1)) K>
```

In [23]:

```
f[0, 0, 0] # shorthand method - leave out the .subspace
```

Out[23]:

```
<CF Field: air_temperature(time(1), latitude(1), longitude(1)) K>
```

In [24]:

```
f.subspace[0:6, :, :]
```

Out[24]:

```
<CF Field: air_temperature(time(6), latitude(145), longitude(192)) K>
```

Metadata-space: (round brackets)

In [25]:

```
print f
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(120), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(120) = [1959-12-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

In [26]:

```
f.subspace(longitude=180) # No shorthand for the "round brackets" form
```

Out[26]:

```
<CF Field: air_temperature(time(120), latitude(145), longitude(1)) K>
```

`cf.lt(30)` is a "query" that means *less than 30*

https://cfpython.bitbucket.io/docs/latest/function.html#comparison
(https://cfpython.bitbucket.io/docs/latest/function.html#comparison)

In [27]:

```
print f.subspace(latitude=cf.lt(30))
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(120), latitude(96), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(120) = [1959-12-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
               : latitude(96) = [-90.0, ..., 28.75] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

`cf.wi(90, 270)` is a query that means *within the range [90, 270]*

In [28]:

```
print f.subspace(longitude=cf.wi(90, 270))
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(120), latitude(145), longitude(97)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(120) = [1959-12-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(97) = [90.0, ..., 270.0] degrees_east
```

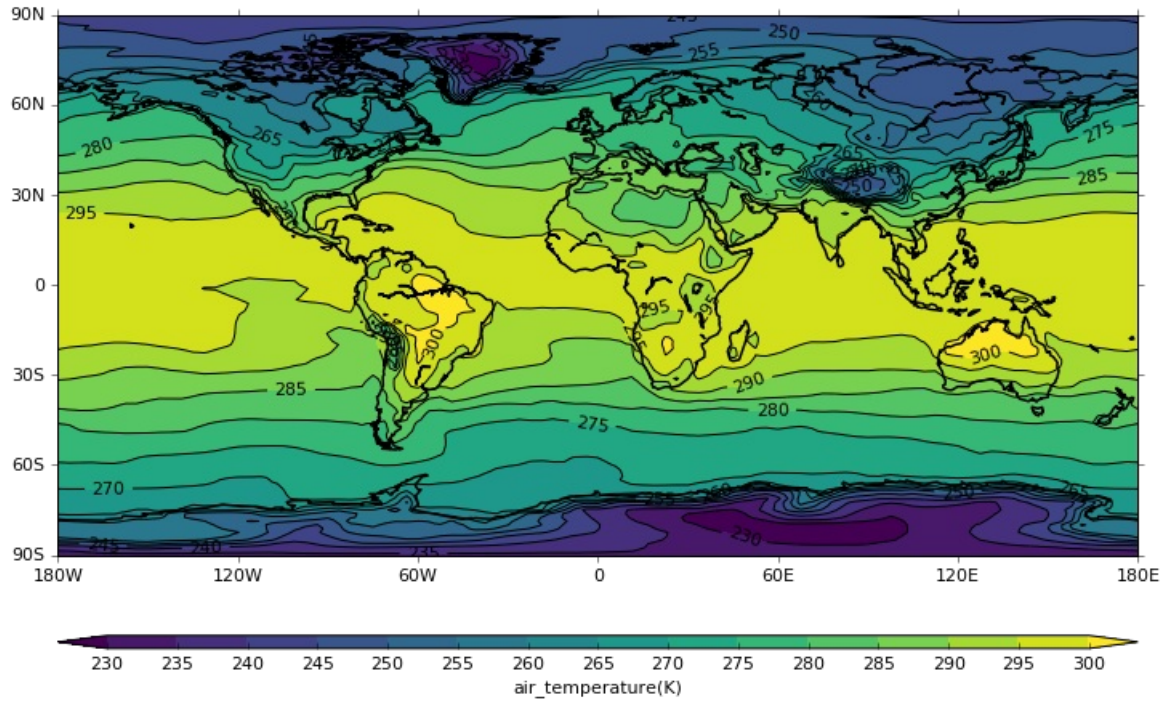
In [29]:

```
g = f.subspace(time=cf.dt('1965-11-16'))
print g
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(1), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(1) = [1965-11-16T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

In [30]:

```
%matplotlib inline
import cfplot as cfp
cfp.con(g)
```



T is shorthand for *time*

In [31]:

```
print f.subspace(T=cf.ge(cf.dt('1967-2-18')))
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(33), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(33) = [1967-03-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

In [32]:

```
print f.subspace(T=cf.month(4))
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(10), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(10) = [1960-04-16T00:00:00Z, ..., 1969-04-16T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

In [33]:

```
print f.subspace(time=cf.dt('1965-11-16'), Y=cf.gt(30))
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(1), latitude(48), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(1) = [1965-11-16T00:00:00Z] 365_day
               : latitude(48) = [31.25, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

3. The field's data

In [34]:

```
f.data
```

Out[34]:

```
<CF Data: [[[244.825790405, ..., 244.526885986]]] K>
```

Get the data as a `numpy` array

In [35]:

```
#f.array # This is numpy array
```

In [36]:

```
f.array[-1, 2, -3]
```

Out[36]:

```
237.00251856000077
```

In [37]:

```
f.subspace[-1, 2, -3].array
```

Out[37]:

```
array([[[ 237.00251856]]])
```

In [38]:

```
x = f.copy()
x.subspace[-1, -1, -1] = -999
x.subspace[-1, -1, -1].array
```

Out[38]:

```
array([[[-999.]]])
```

In [39]:

```
x.subspace[-1, ...] = 888
x.subspace[-1, ...].array
```

Out[39]:

```
array([[[ 888.,  888.,  888., ...,  888.,  888.,  888.],
        [ 888.,  888.,  888., ...,  888.,  888.,  888.],
        [ 888.,  888.,  888., ...,  888.,  888.,  888.],
        ...,
        [ 888.,  888.,  888., ...,  888.,  888.,  888.],
        [ 888.,  888.,  888., ...,  888.,  888.,  888.],
        [ 888.,  888.,  888., ...,  888.,  888.,  888.]]])
```

In [40]:

```
x.subspace[0, ...] = x.subspace[-1, ...] - 111
x.subspace[0, ...].array
```

Out[40]:

```
array([[[ 777.,  777.,  777., ...,  777.,  777.,  777.],
        [ 777.,  777.,  777., ...,  777.,  777.,  777.],
        [ 777.,  777.,  777., ...,  777.,  777.,  777.],
        ...,
        [ 777.,  777.,  777., ...,  777.,  777.,  777.],
        [ 777.,  777.,  777., ...,  777.,  777.,  777.],
        [ 777.,  777.,  777., ...,  777.,  777.,  777.]]])
```

Modify the data where a condition is met

https://cfpython.bitbucket.io/docs/latest/generated/cf.Field.where.html (https://cfpython.bitbucket.io/docs/latest/generated/cf.Field.where.html)

In [41]:

```
f.min(), f.mean(), f.max()
```

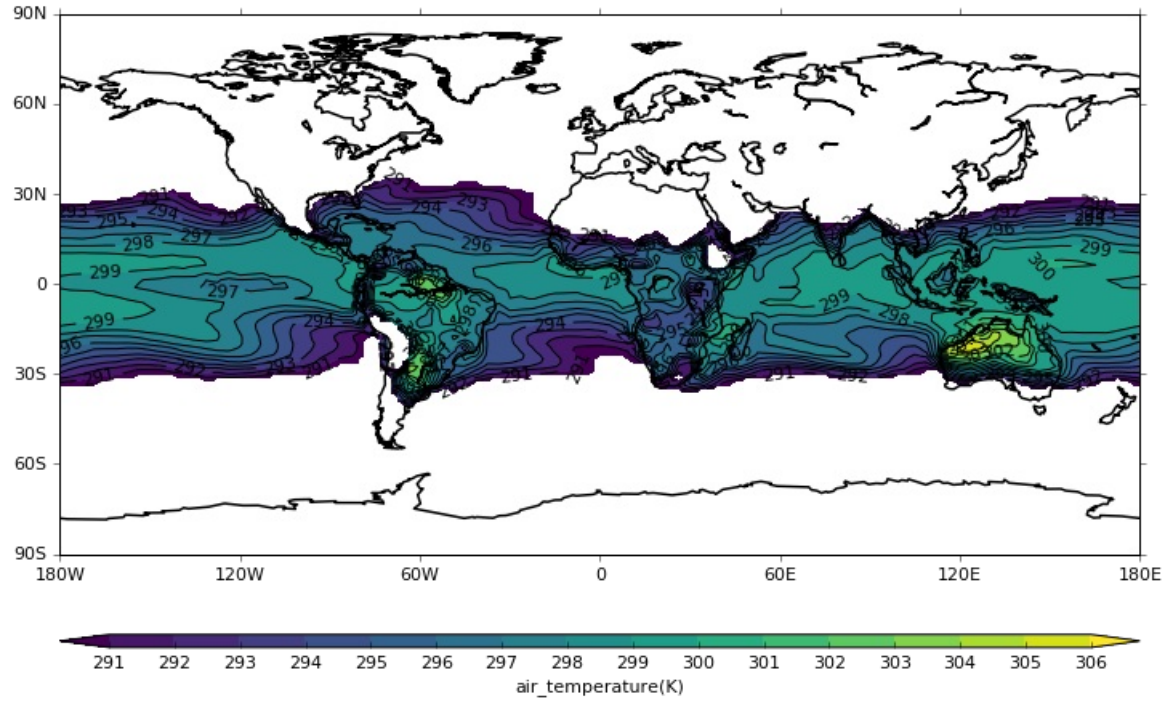
Out[41]:

```
(<CF Data: 203.624511719 K>,
 <CF Data: 276.584738291 K>,
 <CF Data: 311.895974978 K>)
```

Set values below 290 to missing data

In [42]:

```
x = f.where(cf.lt(290), cf.masked)
x.min(), x.mean(), x.max()
cfp.con(x.subspace[0])
```



Manipulate the axes

In [43]:

```
f.transpose(['X', 'T', 'Y'])
```

Out[43]:

```
<CF Field: air_temperature(longitude(192), time(120), latitude(145)) K>
```

Modifying the units

In [44]:

```
f = cf.read_field('ncas_data/IPSL-CM5A-LR_r1i1p1_tas_n96_rcp45_mnth.nc')
f.units, f.mean()
```

Out[44]:

```
('K', <CF Data: 276.584738291 K>)
```

In [45]:

```
f.units = 'degC'
f.units, f.mean()
```

Out[45]:

```
('degC', <CF Data: 3.43473829149 degC>)
```

In [46]:

```
f.Units # Upper case "U" gives a units object that we can manipulate
```

Out[46]:

```
<CF Units: degC>
```

In [47]:

```
f.Units += 273.15
f.Units, f.units, f.mean()
```

Out[47]:

```
(<CF Units: K>, 'K', <CF Data: 276.584738291 K>)
```

Field arithmetic

In [48]:

```
f
```

Out[48]:

```
<CF Field: air_temperature(time(120), latitude(145), longitude(192)) K>
```

In [49]:

```
f.min(), f.mean(), f.max()
```

Out[49]:

```
(<CF Data: 203.624511719 K>,
 <CF Data: 276.584738291 K>,
 <CF Data: 311.895974978 K>)
```

In [50]:

```
g = f + 2
g
```

Out[50]:

```
<CF Field: air_temperature(time(120), latitude(145), longitude(192)) K>
```

In [51]:

```
g.min(), g.mean(), g.max()
```

Out[51]:

```
(<CF Data: 205.624511719 K>,
 <CF Data: 278.584738291 K>,
 <CF Data: 313.895974978 K>)
```

In [52]:

```
g = f - f
g
```

Out[52]:

```
<CF Field: air_temperature(time(120), latitude(145), longitude(192)) K>
```

In [53]:

```
g.min(), g.mean(), g.max()
```

Out[53]:

```
(<CF Data: 0.0 K>, <CF Data: 0.0 K>, <CF Data: 0.0 K>)
```

In [54]:

```
x = f.copy()
x.units = 'degC'
x.data
```

Out[54]:

```
<CF Data: [[[-28.3242095947, ..., -28.6231140137]]] degC>
```

Subtract the celcius field from the Kelvin field and check that the result is zero

In [55]:

```
(f - x).mean()
```

Out[55]:

```
<CF Data: 0.0 K>
```

In [56]:

```
g = f * f
g
```

Out[56]:

```
<CF Field: (time(120), latitude(145), longitude(192)) K2>
```

Find the anomalies relaitive to the first time

In [57]:

```
first_time = f.subspace[0]
first_time = first_time.transpose(['Y', 'T', 'X'])
first_time
```

Out[57]:

```
<CF Field: air_temperature(latitude(145), time(1), longitude(192)) K>
```

In [58]:

```
g  = f - first_time
g
```

Out[58]:

```
<CF Field: air_temperature(time(120), latitude(145), longitude(192)) K>
```

In [59]:

```
g.min(), g.mean(), g.max()
```

Out[59]:

```
(<CF Data: -32.6200714111 K>,
 <CF Data: 1.44168717443 K>,
 <CF Data: 53.5055999756 K>)
```

4. Statistical operations

https://cfpython.bitbucket.io/docs/latest/generated/cf.Field.collapse.html
(https://cfpython.bitbucket.io/docs/latest/generated/cf.Field.collapse.html)

In [60]:

```
g = f.collapse('max')
g
```

Out[60]:

```
<CF Field: air_temperature(time(1), latitude(1), longitude(1)) K>
```

In [61]:

```
g.data
```

Out[61]:

```
<CF Data: [[[311.895974978]]] K>
```

In [62]:

```
g = f.collapse('T: mean')
print g
print 'data values:\n', g.data
print '\ntime bounds:\n', g.coord('T').bounds.dtarray
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(1), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes) time: mean
Axes           : height(1) = [2.0] m
               : time(1) = [1964-12-01T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east

data values:
[[[227.63307279, ..., 254.509607188]]] K

time bounds:
[[<CF Datetime: 1959-12-01T00:00:00Z 365_day>
  <CF Datetime: 1969-12-01T00:00:00Z 365_day>]]
```

Collapse multiple axes simultaneously

In [63]:

```
g = f.collapse('X: Y: sd')
g
```

Out[63]:

```
<CF Field: air_temperature(time(120), latitude(1), longitude(1)) K>
```

Collapse an axis into groups, rather than a single value

In [64]:

```
g = f.collapse('T: mean', group=cf.seasons())
print g
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(40), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes) time: mean
Axes           : height(1) = [2.0] m
               : time(40) = [1960-01-15T00:00:00Z, ..., 1969-10-16T12:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east
```

`cf.seasons()` is a list of queries, each of which defines a range of months

In [65]:

```
cf.seasons()
```

Out[65]:

```
[<CF Query: month[(ge 12) | (le 2)]>,
 <CF Query: month(wi (3, 5))>,
 <CF Query: month(wi (6, 8))>,
 <CF Query: month(wi (9, 11))>]
```

By default, collpases are not weighted

In [66]:

```
g = f.collapse('area: mean', weights='area') # Area mean for each time
g = g.collapse('T: max')                     # Time maxiumum of the area means
g.data
print g
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(1), latitude(1), longitude(1)) K
Cell methods   : time: mean (interval: 30 minutes) latitude: longitude: mean time: maximum
Axes           : height(1) = [2.0] m
               : time(1) = [1964-12-01T00:00:00Z] 365_day
               : latitude(1) = [0.0] degrees_north
               : longitude(1) = [179.0625] degrees_east
```

File aggregation

Create a sequence of files on disk, each of which contains one year

In [67]:

```
f = cf.read_field('ncas_data/IPSL-CM5A-LR_r1i1p1_tas_n96_rcp45_mnth.nc')
print f
for i in range(10):
    g = f.subspace[12*i:12*(i+1)]
    year = str(g.coord('T').year.array[0])
    new_file = 'air_temperature_'+year+'.nc'
    cf.write(g, new_file)
    print '    ',new_file
```

```
air_temperature field summary
-----------------------------
Data           : air_temperature(time(120), latitude(145), longitude(192)) K
Cell methods   : time: mean (interval: 30 minutes)
Axes           : height(1) = [2.0] m
               : time(120) = [1959-12-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
               : latitude(145) = [-90.0, ..., 90.0] degrees_north
               : longitude(192) = [0.0, ..., 358.125] degrees_east

     air_temperature_1959.nc
     air_temperature_1960.nc
     air_temperature_1961.nc
     air_temperature_1962.nc
     air_temperature_1963.nc
     air_temperature_1964.nc
     air_temperature_1965.nc
     air_temperature_1966.nc
     air_temperature_1967.nc
     air_temperature_1968.nc
```

In ipython ! preceeeds a shell command

In [68]:

```
!ls -o air_temperature_*.nc
```

```
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1959.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1960.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1961.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1962.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1963.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1964.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1965.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1966.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1967.nc
-rw-r--r-- 1 david 2725071 Oct 27 09:04 air_temperature_1968.nc
```

In [69]:

```
f2 = cf.read('air_temperature_*.nc')
print f2
```

```
air_temperature field summary
-----------------------------
Data            : air_temperature(time(120), latitude(145), longitude(192)) K
Cell methods    : time: mean (interval: 30 minutes)
Axes            : height(1) = [2.0] m
                : time(120) = [1959-12-16T12:00:00Z, ..., 1969-11-16T00:00:00Z] 365_day
                : latitude(145) = [-90.0, ..., 90.0] degrees_north
                : longitude(192) = [0.0, ..., 358.125] degrees_east
```

In [70]:

```
f.equals(f2[0])
```

Out[70]:

```
True
```

In [71]:

```
f3 = cf.read('air_temperature_*.nc', aggregate=False)
f3
```

Out[71]:

```
[<CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>,
 <CF Field: air_temperature(time(12), latitude(145), longitude(192)) K>]
```

5. PP and UM files

In [72]:

```
x = cf.read('ncas_data/aaaaoa.pmh8dec.pp')
x
```

Out[72]:

```
[<CF Field: long_name:CANOPY THROUGHFALL RATE      KG/M2/S(grid_latitude(30), grid_longitude(24
)) kg m-2 s-1>,
 <CF Field: relative_humidity(air_pressure(17), grid_latitude(30), grid_longitude(24)) %>,
 <CF Field: relative_humidity(grid_latitude(30), grid_longitude(24)) %>]
```

In [73]:

```
cf.write(x, 'aaaaoa.pmh8dec.nc')
```

In [ ]: