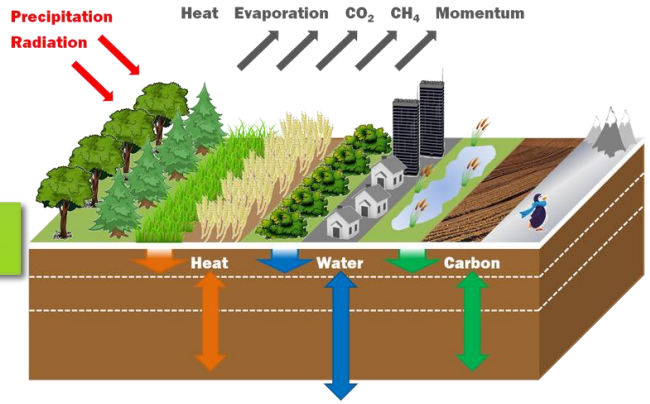
Interpreting JULES output

Emma Robinson, CEH

JULES Short Course

Lancaster, June 29th 2016

Interpreting JULES output



Dump files

Contain enough information to fully describe model state

Used to restart runs

Requested output

Defined in output.nml

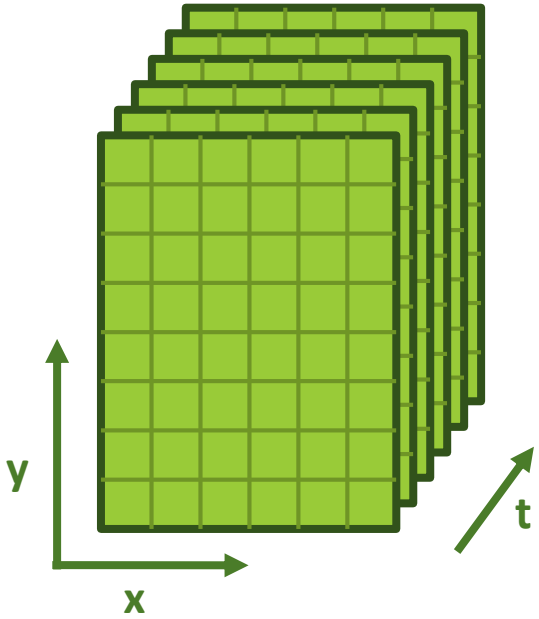
Control over variables, time averaging, spatial and temporal extend

Output format: netCDF

- Widely used way to store gridded data
- Self-describing binary data format
- Portable/machine independent
- Interfaces supported for C, Java, and Fortran
- Interfaces also available for python, R, IDL, MATLAB, C++, Ruby, and Perl.

https://www.unidata.ucar.edu/software/netcdf

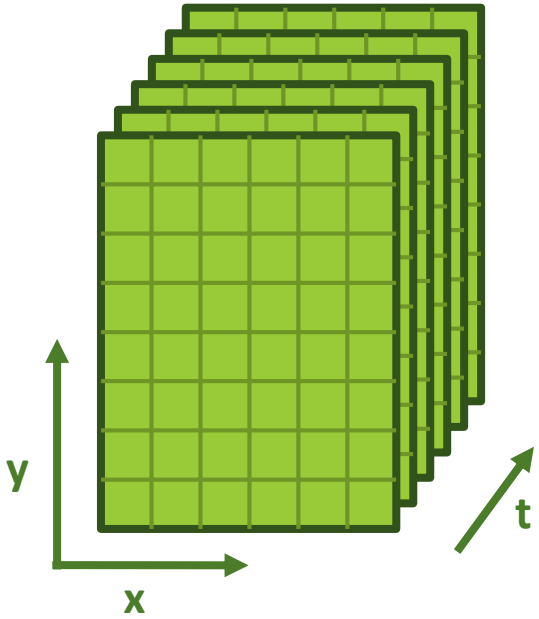
Output format: netCDF




```
wlcmp ~ $ ncdump -h chess_tas_196101.nc
```

ncdump

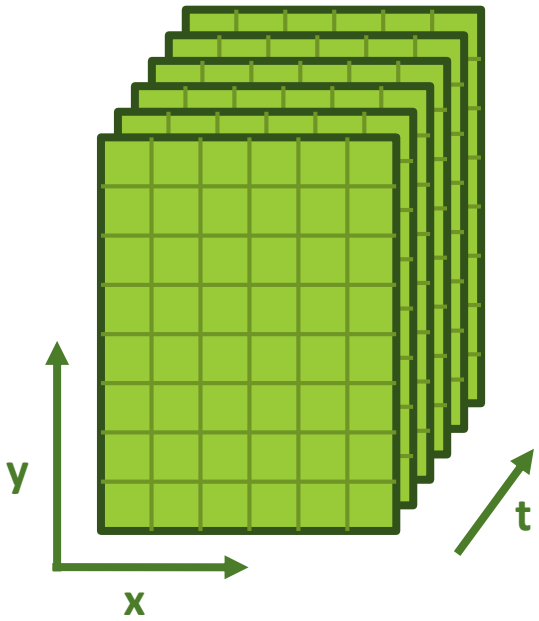
Output format: netCDF




```
wlcmp ~ $ ncdump -h chess_tas_196101.nc
netcdf chess_tas_196101 {
dimensions:
                y = 1057 ;
                x = 656 ;
                time = 31 ;
variables:
                float lat(y, x) ;
                                lat:long_name = "latitude of grid box centre" ;
                                lat:standard_name = "latitude" ;
                                lat:units = "degrees_north" ;
                                lat:_FillValue = -99999.f ;
                float lon(y, x) ;
                                lon:long_name = "longitude of grid box centre" ;
                                lon:standard_name = "longitude" ;
                                lon:units = "degrees_east" ;
                                lon:_FillValue = -99999.f ;
                float time(time) ;
                                time:units = "days since 1961-01-01" ;
                                time:long_name = "Time in days since 1961-01-01" ;
                                time:_FillValue = -99999.f ;
                float tas(time, y, x) ;
                                tas:_FillValue = -99999.f ;
                                tas:standard_name = "air_temperature" ;
                                tas:long_name = "Near-Surface Air Temperature" ;
                                tas:units = "K" ;
                                tas:comment = "1.2m above surface" ;

// global attributes:
                                :title = "Near-Surface Air Temperature" ;
                                :history = "Thu Jan 22 15:58:38 2015: ncks --fix_rec_dmn time
chess_tas_196101.nc" ;
                                :date_created = "2015-01-22" ;
                                :creator_name = "Emma Robinson" ;
                                :time_coverage_start = "1961-01-01" ;
                                :time_coverage_end = "1961-01-31" ;
                                :version = "v1.0" ;
}
```

Output format: netCDF

JULES
Joint UK Land Environment Simulator




```
wlcmp ~ $ ncdump -h chess_tas_196101.nc
netcdf chess_tas_196101 {
dimensions:
                y = 1057 ;
                x = 656 ;
                time = 31 ;
variables:
                float lat(y, x) ;
                        lat:long_name = "latitude of grid box centre" ;
                        lat:standard_name = "latitude" ;
                        lat:units = "degrees_north" ;
                        lat:_FillValue = -99999.f ;
                float lon(y, x) ;
                        lon:long_name = "longitude of grid box centre" ;
                        lon:standard_name = "longitude" ;
                        lon:units = "degrees_east" ;
                        lon:_FillValue = -99999.f ;
                float time(time) ;
                        time:units = "days since 1961-01-01" ;
                        time:long_name = "Time in days since 1961-01-01" ;
                        time:_FillValue = -99999.f ;
                float tas(time, y, x) ;
                        tas:_FillValue = -99999.f ;
                        tas:standard_name = "air_temperature" ;
                        tas:long_name = "Near-Surface Air Temperature" ;
                        tas:units = "K" ;
                        tas:comment = "1.2m above surface" ;

// global attributes:
                        :title = "Near-Surface Air Temperature" ;
                        :history = "Thu Jan 22 15:58:38 2015: ncks --fix_rec_dmn time
chess_tas_196101.nc" ;
                        :date_created = "2015-01-22" ;
                        :creator_name = "Emma Robinson" ;
                        :time_coverage_start = "1961-01-01" ;
                        :time_coverage_end = "1961-01-31" ;
                        :version = "v1.0" ;
}
```

dimensions

dimension variables

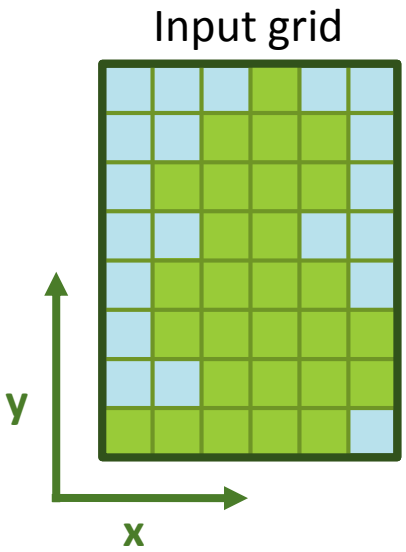
data variables

global information

Model grid

model_grid.nml

```
&JULES_MODEL_GRID

land_only = .false.

/

...
```



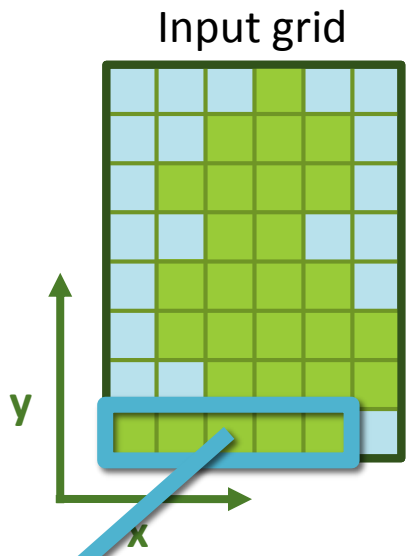
Model grid

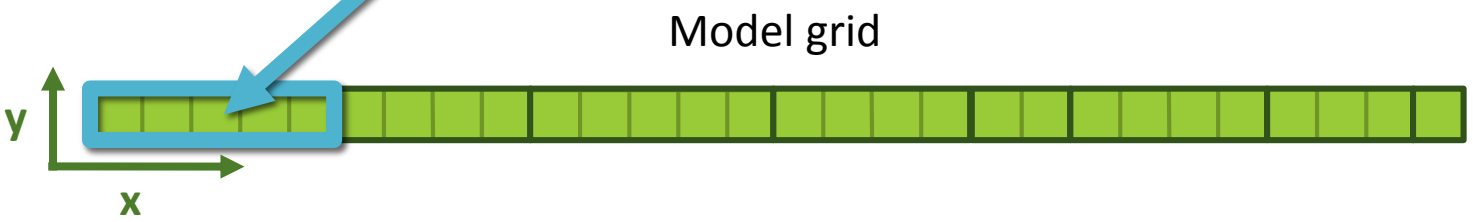
JULES
Joint UK Land Environment Simulator

model_grid.nml

```
&JULES_MODEL_GRID

land_only = .true.

/

...
```

default = .true.





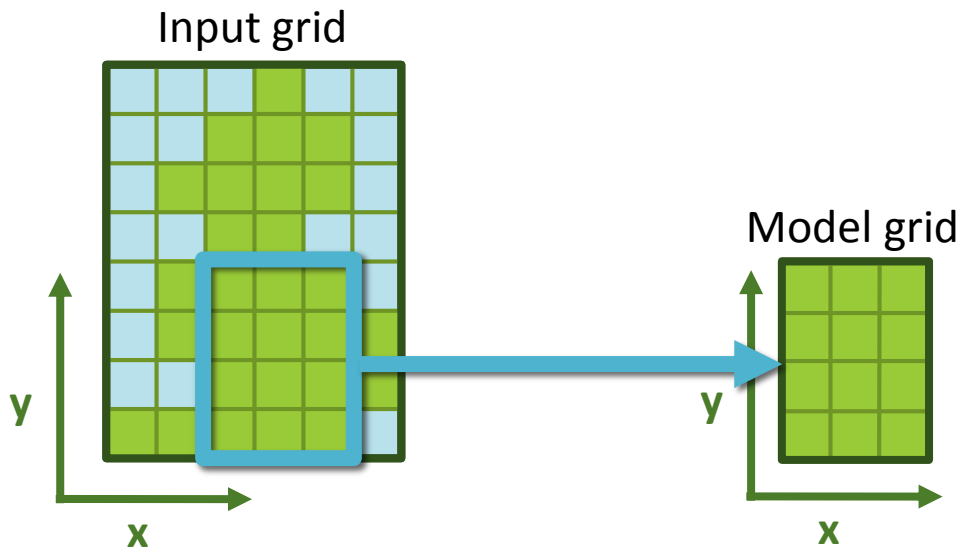
Model grid

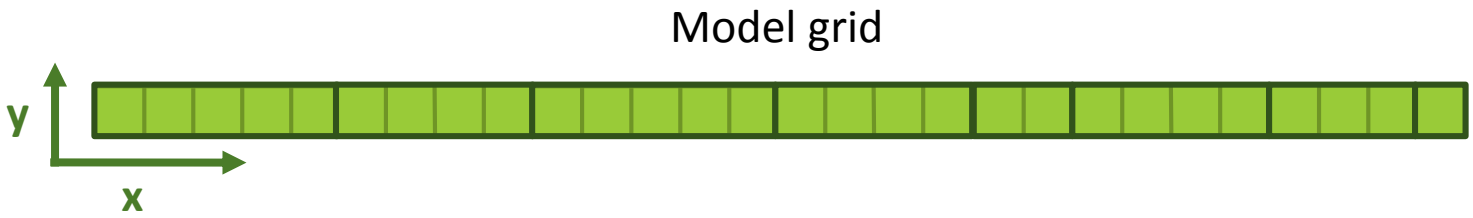
JULES
Joint UK Land Environment Simulator

model_grid.nml

```
&JULES_MODEL_GRID

land_only = .true.

/

...
```

default = .true.





Dump files

- Contain enough information to get the full model state (including ancillary data)
- Output on model grid
- Automatically generated
  - After initialisation is complete, immediately before the start of the run (initial state).
  - Before starting each cycle of spin-up.
  - Before starting the main run.
  - At the end of the run (final state).
  - At the start of each calendar year.
- Use to start/restart runs

initial_conditions.nml

```
&JULES_INITIAL

  dump_file = T

  file = 'run1.dump.19610101.0.nc'

/
```

output.nml

- Output profiles
- Frequency of files
  - Monthly
  - Annual
  - Single
- Can restrict to sub-set of run
  - Spin-up
  - Main run
  - Sub-set of main run
- Choice of variables
- Output frequency

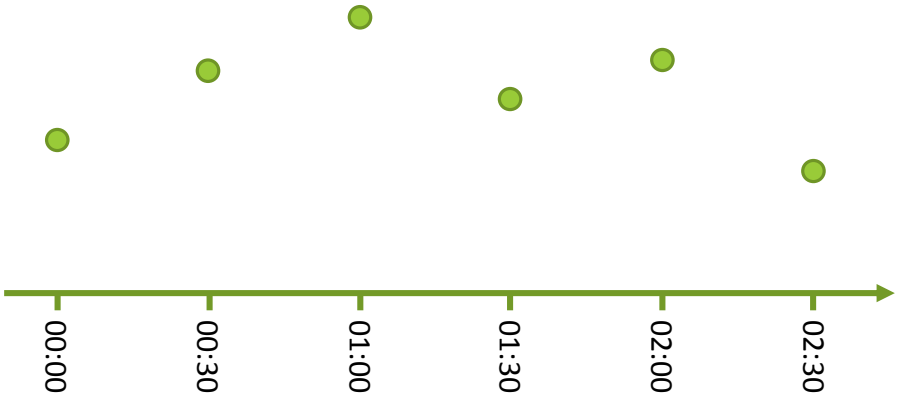
output.nml

```
&JULES_OUTPUT
  run_id = 'run1',
  nprofiles = 2,
  output_dir = './output',
/

&JULES_OUTPUT_PROFILE
  output_main_run = .true.,
  nvars = 2,
  profile_name = 'monthly_fluxes',
  var = 'fqw_gb'  'ftl_gb',
  output_type = 'M'  'M',
  output_period=-1
/
&JULES_OUTPUT_PROFILE
  output_main_run = .true.,
  output_spinup = .true.,
  nvars = 2,
  profile_name = 'daily_t',
  var = 't_soil'  'tstar'
  output_type = 'S'  'S',
  output_period=86400
/
```

http://jules-lsm.github.io/vn4.5/output-variables.html

Output period

- Can be any multiple of timestep period
- Special values
  - **-1** Monthly period
  - **-2** Annual period
- Type of output
  - **S** Instantaneous or snapshot value.
  - **M** Time mean value.
  - **N** Time minimum value.
  - **X** Time maximum value.
  - **A** Accumulation over time.
- two time variables



time

End of the output period.
Time at which snapshot values apply.

time_bounds

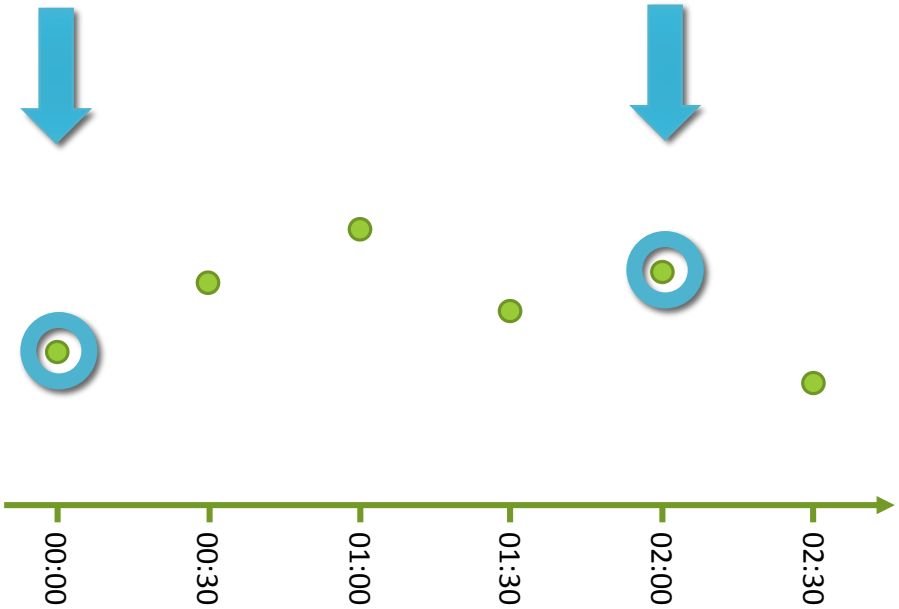
Start and end of the output period

time_bounds(1) < time <= time_bounds(2)

Used for means, minima, maxima, accumulations

Output period

- Can be any multiple of timestep period
- Special values
  - **-1** Monthly period
  - **-2** Annual period
- Type of output
  - **S** Instantaneous or snapshot value.
  - **M** Time mean value.
  - **N** Time minimum value.
  - **X** Time maximum value.
  - **A** Accumulation over time.
- two time variables



time

End of the output period.

Time at which snapshot values apply.

time_bounds

Start and end of the output period

time_bounds(1) < time <= time_bounds(2)

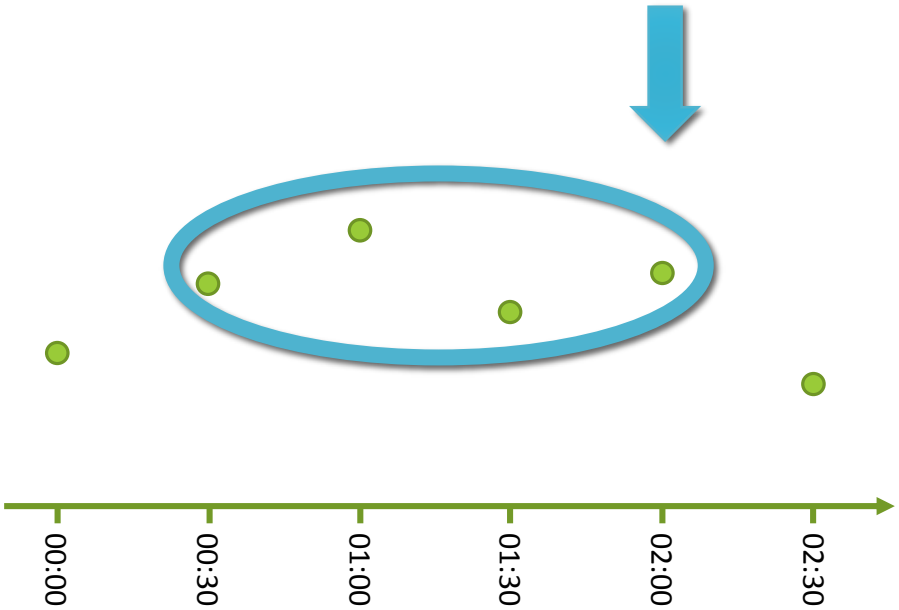
Used for means, minima, maxima, accumulations

Output period

- Can be any multiple of timestep period
- Special values
  - **-1** Monthly period
  - **-2** Annual period
- Type of output
  - **S** Instantaneous or snapshot value.
  - **M** Time mean value.
  - **N** Time minimum value.
  - **X** Time maximum value.
  - **A** Accumulation over time.
- two time variables

time

End of the output period.
Time at which snapshot values apply.



time_bounds

Start and end of the output period

time_bounds(1) < time <= time_bounds(2)

Used for means, minima, maxima, accumulations

How to read output

- GrADS (Alberto)
- R
- Python
  - Iris
- Viewers
- more...

R

JULES
Joint UK Land Environment Simulator

Package required to read netCDF files

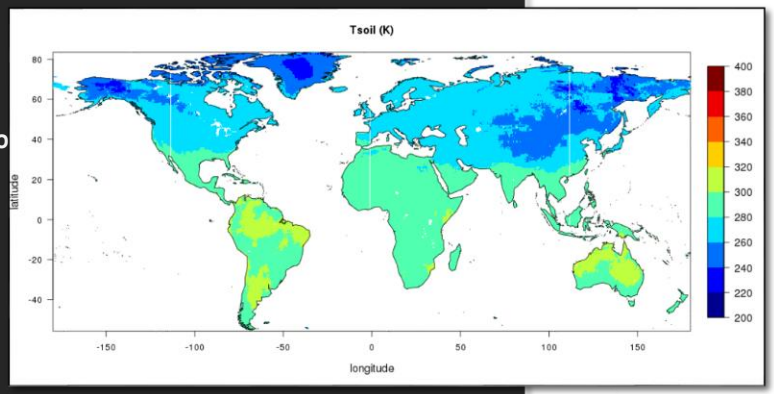
- ncdf4
  - https://cran.r-project.org/web/packages/ncdf4/index.html
- RNetCDF
  - https://cran.r-project.org/web/packages/RNetCDF/index.html

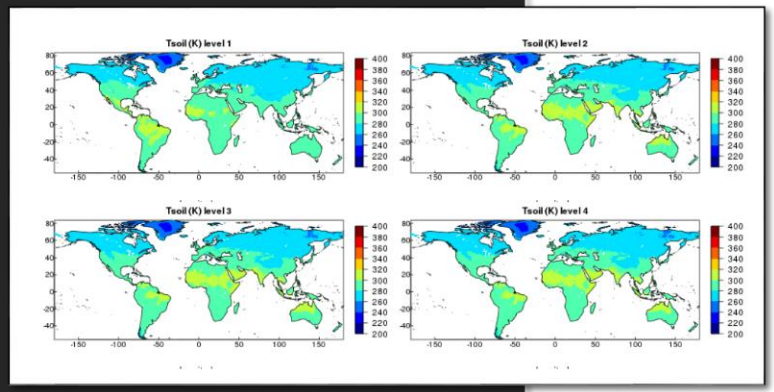
R example (ncdf4)

JULES
Joint UK Land Environment Simulator

jules_short_course_R.r

```
### 22.06.16 rfu
###Short R script to read JULES NetCDF output and plot


###Packages required
###If not already downloaded use install.packages(), select somewhere to
###downlad from, then select required package to install
require(ncdf4)
require(fields)
require(maps)
require(RColorBrewer)
require(reshape)


  ##File name
ncname <- ./runs/test1/output/run1.daily_t.1980.nc"

  ## open netCDF file and print header information
ncin <- nc_open(ncname)
print(ncin)

  ## get variables:
lon <- ncvar_get(ncin,"longitude")
nlon <- dim(lon)
head(lon)
lat <- ncvar_get(ncin,"latitude")
nlat <- dim(lat)
head(lat)
```





R example (ncdf4)

JULES
Joint UK Land Environment Simulator

- Load library:

```
require(ncdf4)
```

- Open file

```
ncin <- nc_open(ncname)
```

- Read variable, then subset

```
t_soil <- ncvar_get(ncin,"t_soil")
t_soil.1 <- t_soil[,1,2:4]
```

- Redistribute from land vector to grid using reshape

Python

Module required to interface with the files

- netCDF4
  - https://pypi.python.org/pypi/netCDF4

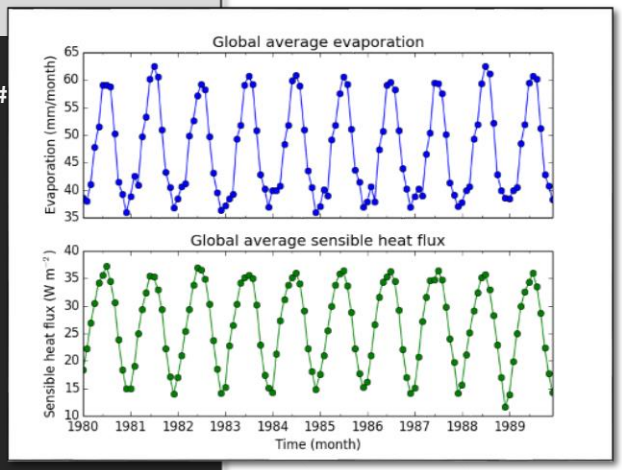
- netcdf4-python from Unidata
  - https://github.com/Unidata/netcdf4-python

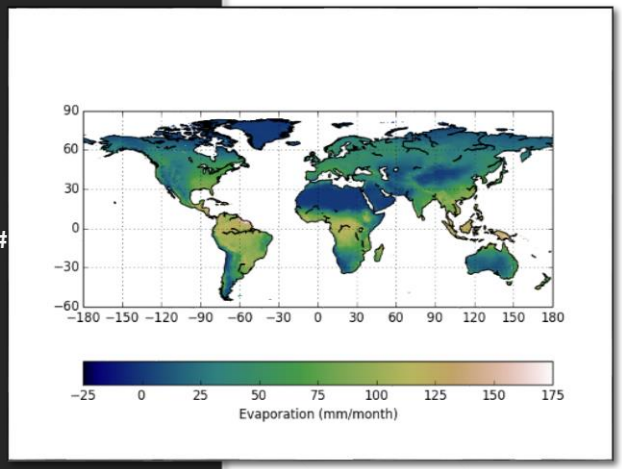
Python example (netCDF4)

JULES
Joint UK Land Environment Simulator

jules_short_course_python.py

```
#!/usr/bin/env python
##################################################################################

# Import netCDF library
import netCDF4 as nc

# Import numpy and matplotlib
import numpy as np
import matplotlib.pyplot as plt

# Import Basemap to make maps
from mpl_toolkits.basemap import Basemap

# Import libraries useful for times
import datetime as dt
import calendar as cal

# Import locators for fancying up plots
from matplotlib.dates import YearLocator, MonthLocator
from matplotlib.ticker import MultipleLocator

##################################################################################
# 1. Read the data

# File name
fname = 'runs/test1/output/run1.monthly_fluxes.nc'

# Open the file
f = nc.Dataset(fname,'r')
```





Python example (netCDF4)

JULES
Joint UK Land Environment Simulator

- Import module:

```
import netCDF4
```

- Open file (use ‘r’ for read and ‘w’ for write)

```
f = nc.Dataset(fname,’r’)
```

- Read variable (can subset at io)

```
fqw_gb  = f.variables['fqw_gb'][:]
ftl_sub = f.variables['ftl'][:10,:,0,2:4]
```

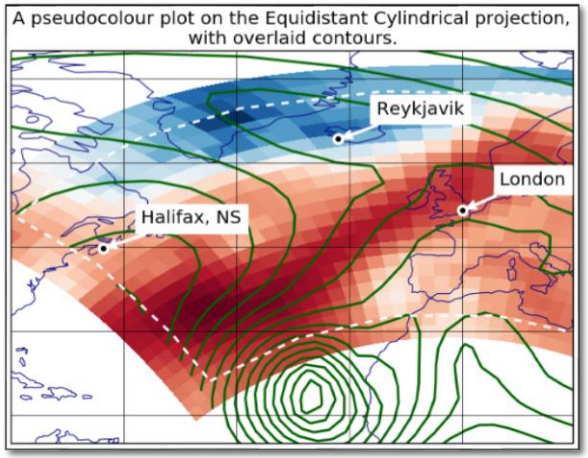
- Redistribute from land vector to grid using indices

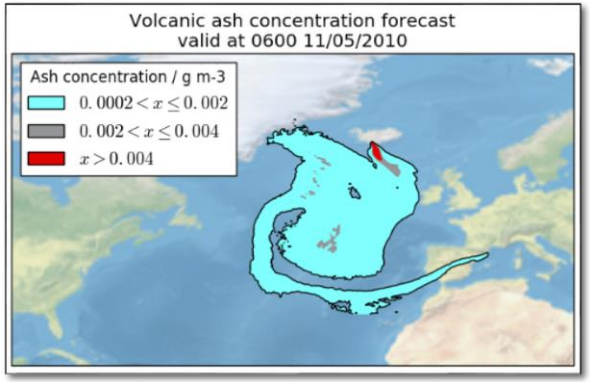
Iris

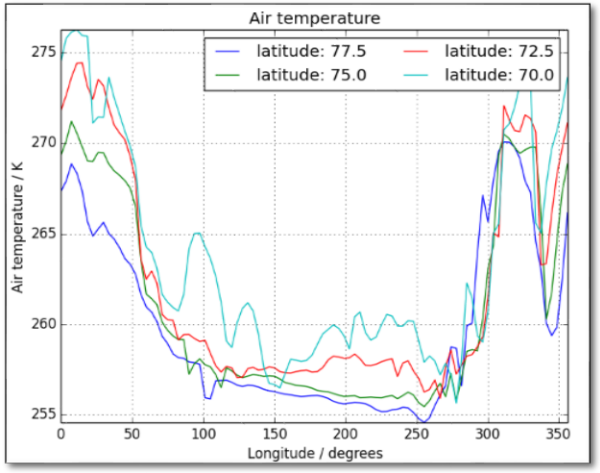
JULES
Joint UK Land Environment Simulator

- Python library for analysing and visualising gridded data sets
- Community-driven, open source
- Tools for reprojection, averaging, statistics…

http://scitools.org.uk/iris/docs/latest/index.html

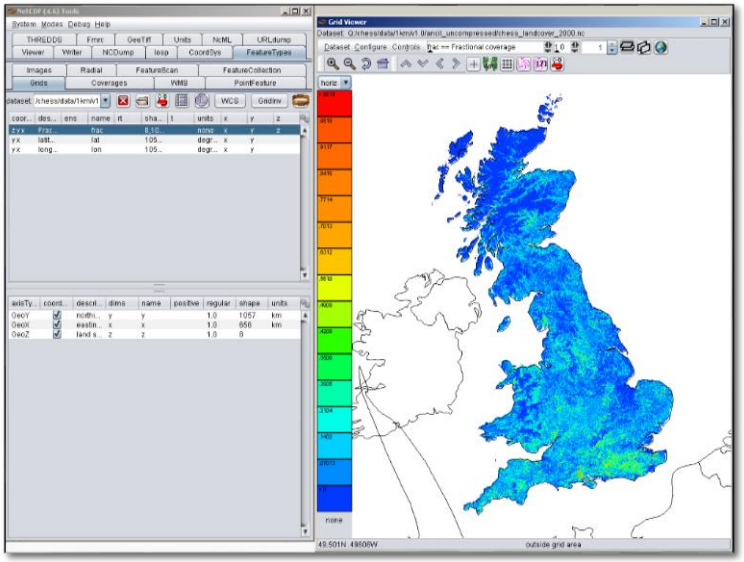


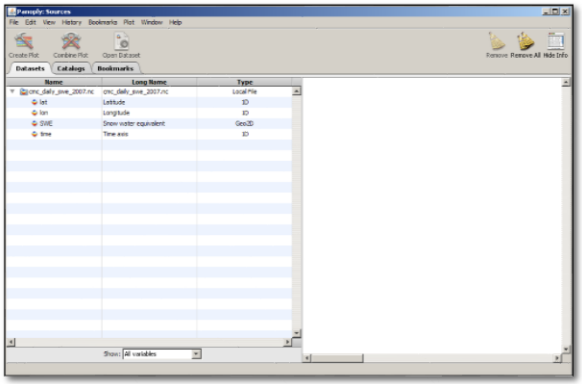


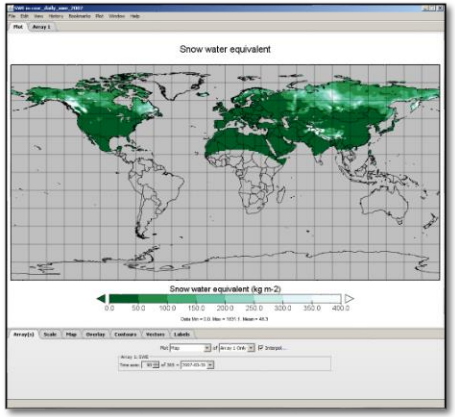


netCDF viewers

- ToolsUI:
  - https://www.unidata.ucar.edu/software/thredds/v4.5/netcdf-java/ToolsUI.html
  - Java-based
  - Requires gridded CF-compliant files
- Panoply:
  - http://www.giss.nasa.gov/tools/panoply/
  - Java-based
  - Can read and plot any files, but needs gridded data for mapping







JULES

Joint UK Land
Environment Simulator