JULES
Joint UK Land Environment Simulator

Example of JULES applications, postprocessing, and visualising outputs

Arthur Argles and Carolina Duran Rojas

Why simulate with JULES?

- JULES is a great resource to study a whole multitude of processes at the Land Surface.
- The model has been applied both in-situ and globally.
- With in-situ observations is often in the form of evaluating a flux: e.g., photosynthesis, soil respiration, hydrology and runoff.
- In larger-scale studies JULES could be evaluated against remote sensing: the biogeophysical response to deforestation or how well we can capture wildfire.
- A very insightful thing to do is to use JULES with different forcings (e.g., present day + 4 degree) and see how vegetation responds.

Evaluating the JULES Land Surface Model Energy Fluxes Using FLUXNET Data

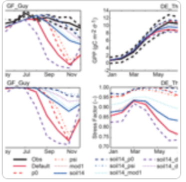
Eleanor Blyth, John Gash, Amanda Lloyd, Matthew Pryor, Graham P. Weedon, and Jim Shuttleworth

Print Publication: 01 Apr 2010

Article | Assets | Peer review | Metrics | Related articles

Model evaluation paper | 03 Jun 2021

Improvement of modeling plant responses to low soil moisture in JULESvn4.9 and evaluation against flux tower measurements



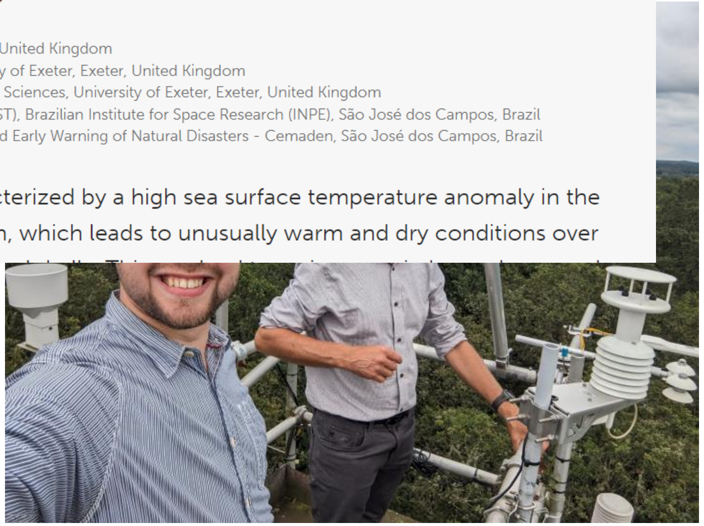
Anna B. Harper, Karina E. Williams, Patrick C. McGuire, Maria Carolina Duran Rojas, Debbie Hemming,

El Niño Driven Changes in Global Fire 2015/16

Chantelle Burton[1*†] Richard A. Betts[1,2†] Chris D. Jones[1†] Ted R. Feldpausch[3†]
Manoel Cardoso[4†] Liana O. Anderson[5†]

[1] Met Office Hadley Centre, Exeter, United Kingdom
[2] Global Systems Institute, University of Exeter, Exeter, United Kingdom
[3] College of Life and Environmental Sciences, University of Exeter, Exeter, United Kingdom
[4] Earth System Science Center (CCST), Brazilian Institute for Space Research (INPE), São José dos Campos, Brazil
[5] National Center for Monitoring and Early Warning of Natural Disasters - Cemaden, São José dos Campos, Brazil

El Niño years are characterized by a high sea surface temperature anomaly in the Equatorial Pacific Ocean, which leads to unusually warm and dry conditions over

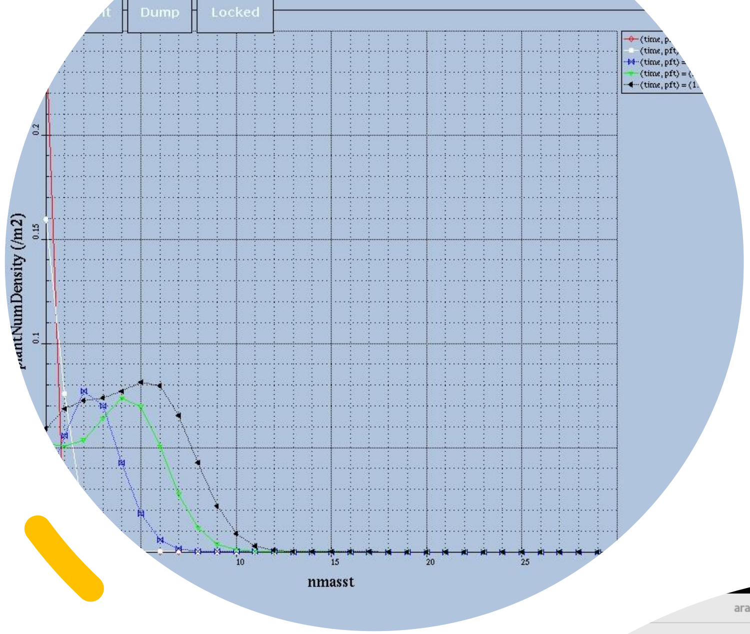


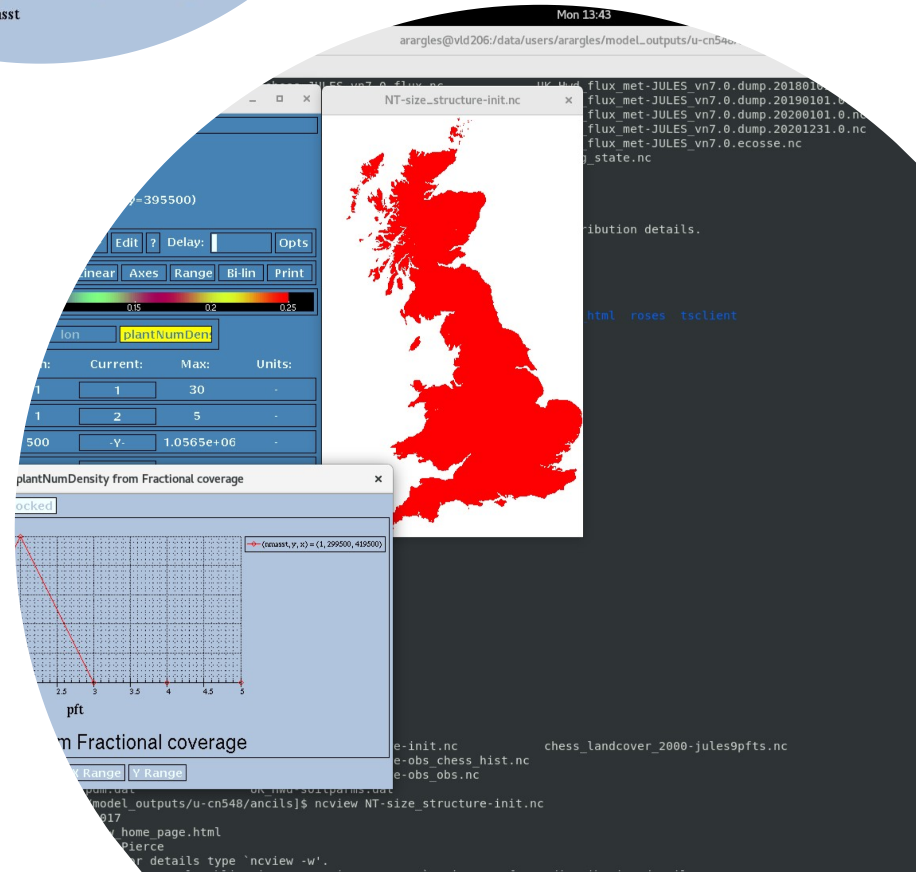
JULES github a great resource:

- Work out what forcing variables you need to run JULES: https://jules-lsm.github.io/latest/namelists/drive.nml.html. Temperature, Pressure, Humidity, Radiative forcing (SW/LW) net or downward, Precipitation, and Wind speed. If greater than 1 day resolution, you must include the daily temperature range.
- To understand how JULES outputs variables: https://jules-lsm.github.io/latest/namelists/output.nml.html#namelist-JULES_OUTPUT_PROFILE
- To see the list of possible output variables along with documentation: https://jules-lsm.github.io/latest/output-variables.html#jules-output-variables

Some useful linux commands to see how your JULES run is getting along

- ncdump *‘filename.nc’*
- ncdump –h (just view meta data of file) *‘filename.nc’*
- ncview *‘filename.nc’*
- xconv *‘filename.nc’*(alternate, sometimes better behaved than ncview)





Some useful Python modules to evaluate JULES

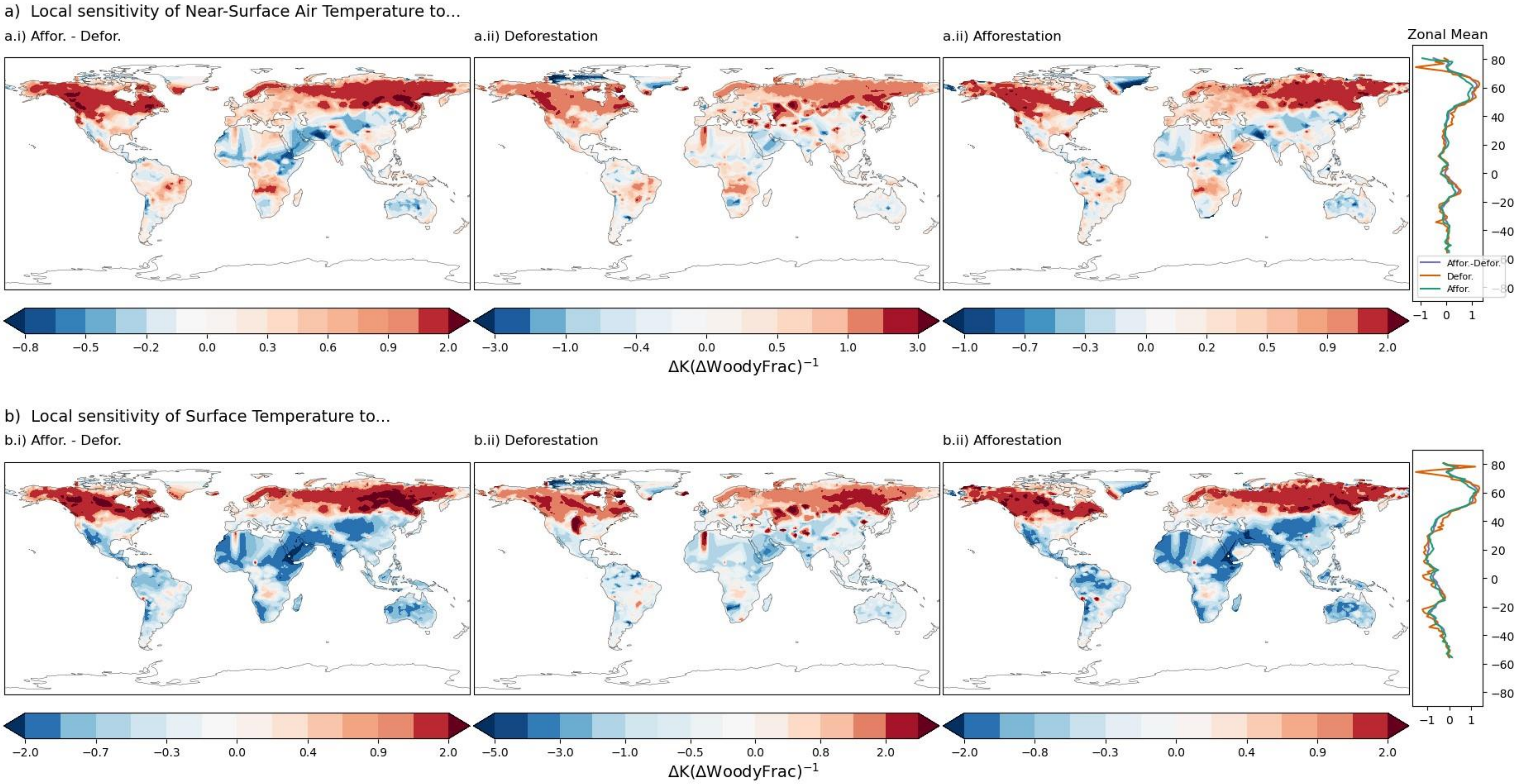
For opening, reading and modifying netcdf files:

netcdf4: https://unidata.github.io/netcdf4-python/

For everything:

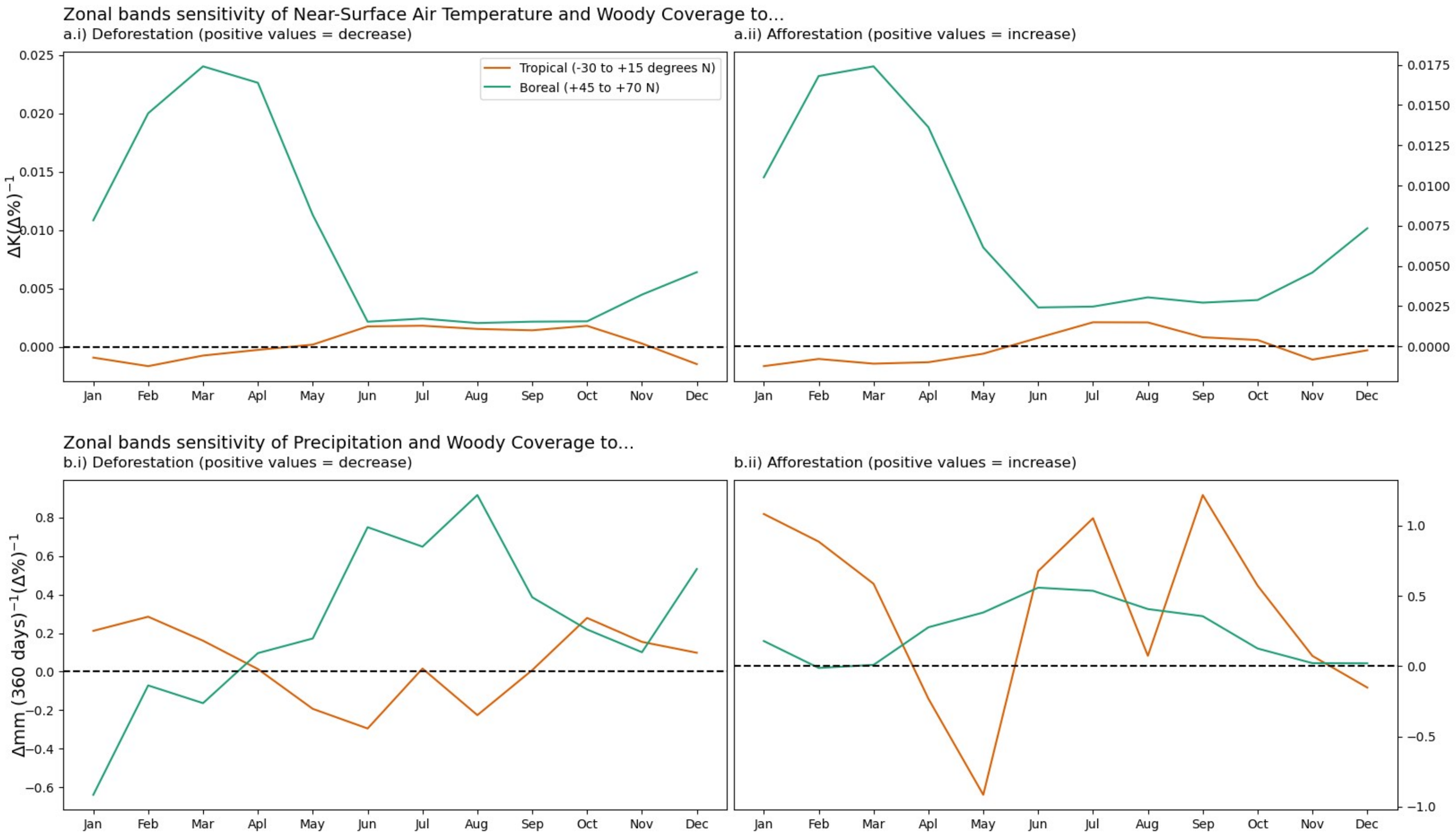
iris: https://scitools-iris.readthedocs.io/en/stable/

Global examples of JULES output



a) Local sensitivity of Near-Surface Air Temperature to...
a.i) Affor. - Defor.
a.ii) Deforestation
a.ii) Afforestation
Zonal Mean
Affor.-Defor.
Defor.
Affor.
−0.8 −0.5 −0.2 0.0 0.3 0.6 0.9 2.0
−3.0 −1.0 −0.4 0.0 0.5 1.0 3.0
−1.0 −0.7 −0.3 0.0 0.2 0.5 0.9 2.0
ΔK(ΔWoodyFrac)$^{-1}$
b) Local sensitivity of Surface Temperature to...
b.i) Affor. - Defor.
b.ii) Deforestation
b.ii) Afforestation
−2.0 −0.7 −0.3 0.0 0.4 0.9 2.0
−5.0 −3.0 −1.0 −0.5 0.0 0.8 2.0
−2.0 −0.8 −0.3 0.0 0.4 0.9 2.0
ΔK(ΔWoodyFrac)$^{-1}$

Averaging for a region across time and space:

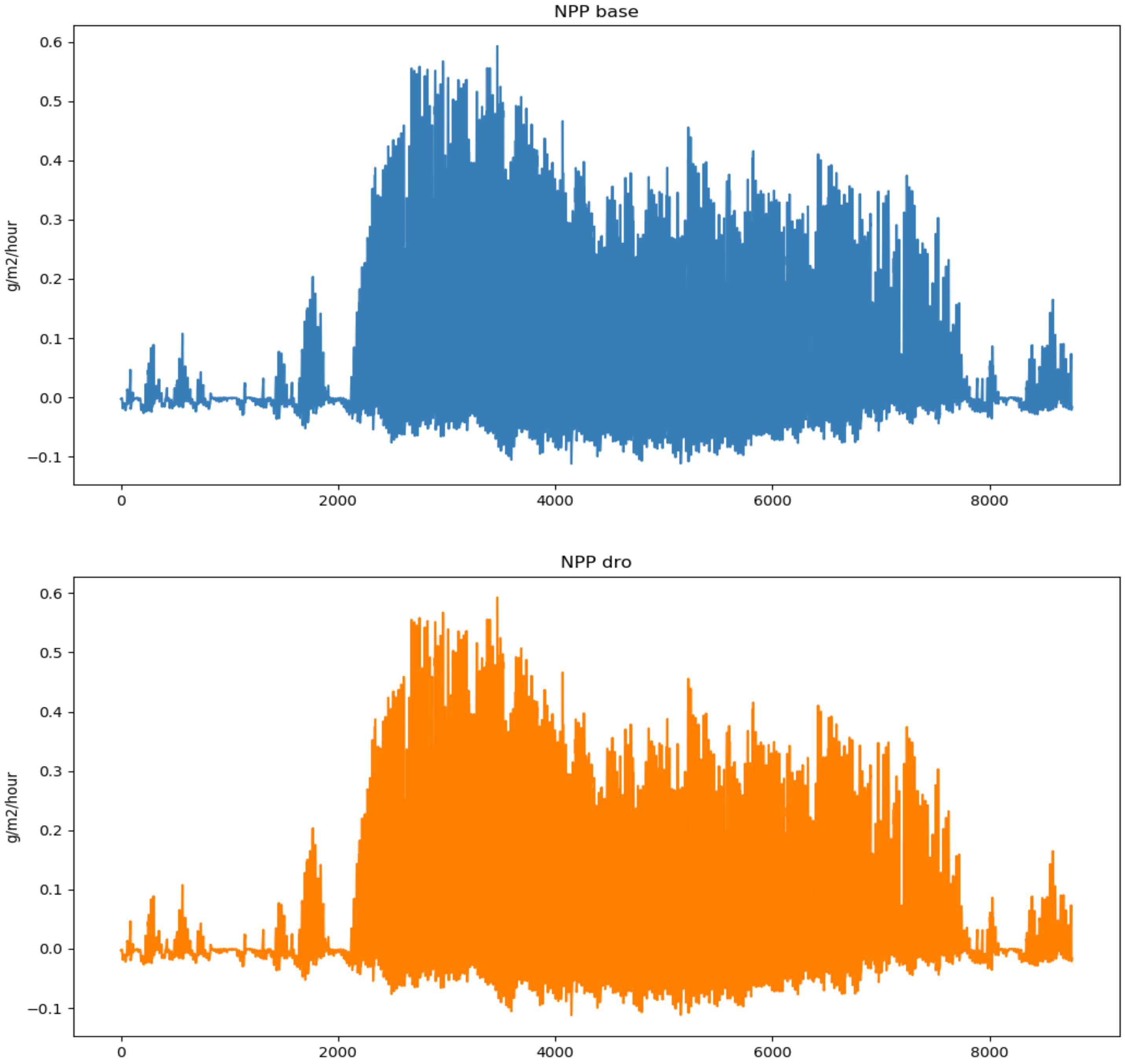


Site examples of JULES output

ncdump –h FILE_NAME

```
netcdf JULES5.2-WB-base.H.2006 {
dimensions:
        time = UNLIMITED ; // (8760 currently)
        y = 1 ;
        x = 1 ;
variables:
        float ANPP(time, y, x) ;
                ANPP:_FillValue = -1.e+20f ;
                ANPP:cell_methods = "time : mean" ;
                ANPP:coordinates = "latitude longitude" ;
                ANPP:long_name = "Aboveground NPP i.e. change to wood and leaf carbon (after tile frac change) on gridbox minus wood a" ;
                ANPP:missing_value = -1.e+20f ;
                ANPP:units = "gC m-2 h-1" ;
        float GPP(time, y, x) ;
                GPP:_FillValue = -1.e+20f ;
                GPP:cell_methods = "time : mean" ;
                GPP:coordinates = "latitude longitude" ;
                GPP:long_name = "Gridbox gross primary productivity" ;
                GPP:missing_value = -1.e+20f ;
                GPP:units = "gC m-2 h-1" ;
        float NPP(time, y, x) ;
                NPP:_FillValue = -1.e+20f ;
                NPP:cell_methods = "time : mean" ;
                NPP:coordinates = "latitude longitude" ;
                NPP:long_name = "Gridbox net primary productivity prior to N limitation" ;
                NPP:missing_value = -1.e+20f ;
                NPP:units = "gC m-2 h-1" ;
        float time(time) ;
                time:standard_name = "time" ;
                time:long_name = "Time of data" ;
                time:units = "seconds since 1995-01-01 00:00:00" ;
                time:bounds = "time_bounds" ;
                time:calendar = "standard" ;
```

NPP output



/work/scratch-pw2/caroduro/training

module load jaspy/2.7/r20190715
python2.7 site_output.py

```
#!/bin/env python3.7
# Aurthor: Dr. Anna Harper
# Used for the JULES training 2023

import os
import numpy as np
import netCDF4 as nc
import matplotlib.pyplot as plt

OUTDIR='OUTPUT'
scenarios=['base','dro']
npoints=14

#This is hourly output. Each annual files has 8760 timesteps.
afilestring='JULES5.2-WB-SCENARIO.H.YYYY.nc'

variables=['NPP']
STARTYEAR=2006
ENDYEAR=2006
YEARS=np.arange(STARTYEAR,ENDYEAR+1)
nyears = ENDYEAR-STARTYEAR+1
ntime = nyears * 8760

nscen = len(scenarios)

model_output = np.empty([ntime,npoints])

for i0 in range(0,nscen):
    ens_string=str(i0).zfill(3)
    print(ens_string)

    afilepath=afilestring.replace('SCENARIO',scenarios[i0])
    d1=0
    for yr in range(STARTYEAR,ENDYEAR+1):
        year_string = str(yr)
        infile=afilepath.replace('YYYY',year_string)
        if os.path.isfile(infile):
            try:
              print('reading '+infile)
              inf=nc.Dataset(infile,'r')
              d2=d1+8760
              model_output[d1:d2,i0] = inf.variables['NPP'][:,0,0]
            except:
              print(infile+' something wrong with this file')
        d1=d2
#this output has been converted to g/m2/hour

fig2,axes2=plt.subplots(ncols=1,nrows=2,figsize=[12,12])
fig2.subplots_adjust(bottom=0.08,left=0.10,top=0.92,right=0.90)

CB_color_cycle = ['#377eb8', '#ff7f00', '#4daf4a',
                  '#f781bf', '#a65628', '#984ea3',
                  '#999999', '#e41a1c', '#dede00',
                  '#377eb8', '#ff7f00', '#4daf4a',
                  '#f781bf', '#a65628']
lstyles=['-','-','-','-','-','-','-','-','-','-.','-.','-.','-.','-.','-.']

ax2=axes2[0]
ax2.plot(model_output[:,i0], color=CB_color_cycle[0],linestyle=lstyles[0])
ax2.set_title('NPP '+scenarios[0])
ax2.set_ylabel('g/m2/hour')

ax2=axes2[1]
ax2.plot(model_output[:,i0], color=CB_color_cycle[1],linestyle=lstyles[1])
ax2.set_title('NPP '+scenarios[1])
ax2.set_ylabel('g/m2/hour')

outfile2='hourly_site_outputs.png'
fig2.savefig(outfile2)

plt.show()
```