ARC CENTRE OF EXCELLENCE FOR CLIMATE SYSTEM SCIENCE

Computational Modelling Systems Wiki

Inicio del wiki
Cambios Recientes
Pages and Files
Miembros
Búsqueda

> CMS News
> Induction
> Data Services
> Storage
> CABLE
> MOM
> Unified Model
> WRF
> NU-WRF
> Coupled Models
> LIS
> FAQs
> Resources

ARCCSS Home Page
Research Opportunities

Python Libraries on Raijin (/Python+Libraries+on+Raijin)

Editar | 0 (/Python+Libraries+on+Raijin#discussion) | 3 (/page/history/Python+Libraries+on+Raijin)
… (/page/menu/Python+Libraries+on+Raijin)

Table of Contents

A number of Python libraries useful for climate research are available for use in the ~access area of Raijin, in addition to Numpy and Scipy which are installed by default at NCI.

To access these you first need to run

```
module use ~access/modules
```

then you can load the individual modules.

To see the full list of available libraries run

```
module avail pythonlib
```

You can request a library be installed by emailling the helpdesk, if it's already available on PyPI then this is quick and easy to do.

CDAT-Lite

```
module load pythonlib/cdat-lite
```

Data analysis tools for working with climate data, NetCDF files and regridding data amongst other things

http://proj.badc.rl.ac.uk/cedaservices/wiki/CdatLite

Example

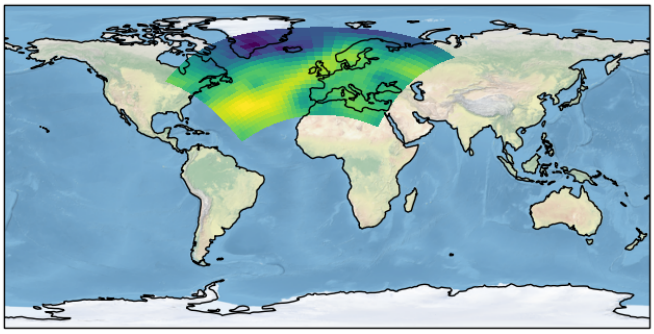
```
#!/usr/bin/env python
import cdms2
from cdms2 import MV
jones = cdms2.open('/pcmdi/cdms/obs/jones_mo.nc')
tasvar = jones['tas']
jans = tasvar[0::12]
julys = tasvar[6::12]
janavg = MV.average(jans)
janavg.id = "tas_jan"
janavg.long_name = "mean January surface temperature"
julyavg = MV.average(julys)
julyavg.id = "tas_jul"
julyavg.long_name = "mean July surface temperature"
out = cdms2.open('janjuly.nc','w')
out.write(janavg)
out.write(julyavg)
out.comment = "Average January/July from Jones dataset"
jones.close()
out.close()
```

Iris

```
module load pythonlib/iris
```

Similar to CDAT, provides tools for working with climate data. Developed by the Met Office to support its data formats, also has integrated plotting tools.

http://scitools.org.uk/iris/



Example

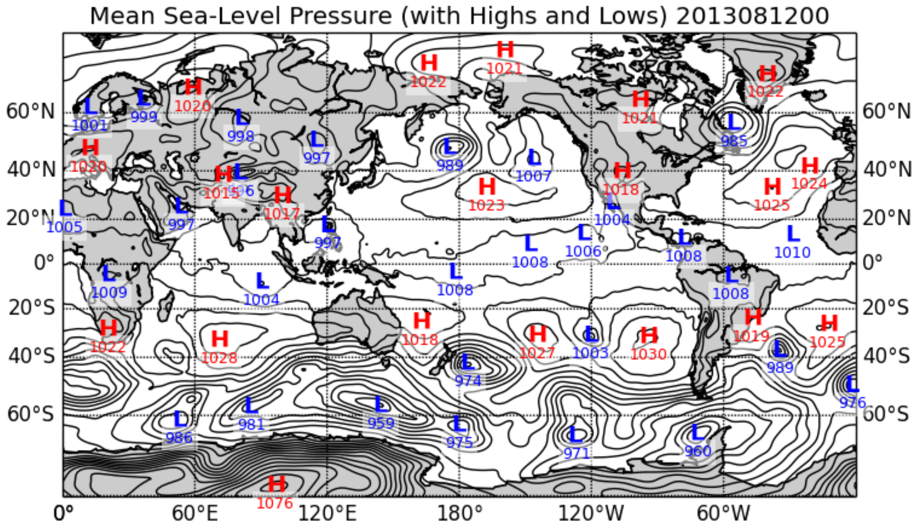
```
import cartopy.crs as ccrs
import matplotlib.pyplot as plt

import iris
import iris.plot as iplt
import iris.quickplot as qplt
import iris.analysis.cartography


def main():
    fname = iris.sample_data_path('rotated_pole.nc')
    air_pressure = iris.load_cube(fname)

    # Plot #1: Point plot showing data values & a colorbar
    plt.figure()
    points = qplt.points(air_pressure, c=air_pressure.data)
    cb = plt.colorbar(points, orientation='horizontal')
    cb.set_label(air_pressure.units)
    plt.gca().coastlines()
    plt.show()

if __name__ == '__main__':
    main()
```

Basemap

```
module load pythonlib/basemap
```

Python cartography library

http://matplotlib.org/basemap/



Example

```
from mpl_toolkits.basemap import Basemap
import matplotlib.pyplot as plt
import numpy as np
# set up orthographic map projection with
# perspective of satellite looking down at 50N, 100W.
# use low resolution coastlines.
map = Basemap(projection='ortho',lat_0=45,lon_0=-100,resolution='l')
# draw coastlines, country boundaries, fill continents.
map.drawcoastlines(linewidth=0.25)
map.drawcountries(linewidth=0.25)
map.fillcontinents(color='coral',lake_color='aqua')
# draw the edge of the map projection region (the projection limb)
map.drawmapboundary(fill_color='aqua')
# draw lat/lon grid lines every 30 degrees.
map.drawmeridians(np.arange(0,360,30))
map.drawparallels(np.arange(-90,90,30))
# make up some data on a regular lat/lon grid.
nlats = 73; nlons = 145; delta = 2.*np.pi/(nlons-1)
lats = (0.5*np.pi-delta*np.indices((nlats,nlons))[0,:,:])
lons = (delta*np.indices((nlats,nlons))[1,:,:])
wave = 0.75*(np.sin(2.*lats)**8*np.cos(4.*lons))
mean = 0.5*np.cos(2.*lats)*((np.sin(2.*lats))**2 + 2.)
# compute native map projection coordinates of lat/lon grid.
x, y = map(lons*180./np.pi, lats*180./np.pi)
# contour data over the map.
cs = map.contour(x,y,wave+mean,15,linewidths=1.5)
plt.title('contour lines over filled continent background')
plt.show()
```

netCDF4

```
module load pythonlib/netCDF4
```

Manipulates NetCDF format files

http://code.google.com/p/netcdf4-python/

Example

```
from netCDF4 import Dataset
rootgrp = Dataset('test.nc', 'w', format='NETCDF4')

level = rootgrp.createDimension('level', None)
time = rootgrp.createDimension('time', None)
lat = rootgrp.createDimension('lat', 73)
lon = rootgrp.createDimension('lon', 144)

times = rootgrp.createVariable('time','f8',('time',))
levels = rootgrp.createVariable('level','i4',('level',))
latitudes = rootgrp.createVariable('latitude','f4',('lat',))
longitudes = rootgrp.createVariable('longitude','f4',('lon',))
# two dimensions unlimited.
```

```
temp = rootgrp.createVariable('temp','f4',('time','level','lat','lon',))

print rootgrp.variables['temp']
```