CEMAC User Documentation: A-CURE

1 Python scripts

1.1 pp2nc_AOD550_CDN.py

1.1.1 Purpose

Extract and condense information from 3-hourly, daily or monthly pp files belonging to the UKCA26AER perturbed parameter ensemble (PPE) set into netCDF format. One set of netCDF files contains aerosol optical depth (AOD) at 550nm, and another set contains column-integrated cloud droplet number concentration (CDNC) fields for all 235 PPE members on a coarsened grid (N96 down to N48). If processing 3-hourly or daily pp files, there is one nc file per day; if processing monthly files, there is one nc file per month.

1.1.2 Usage

The script can be run from the command line as follows:

```
$ ./pp2nc_AOD550_CDN.py <timeRes> <ppRoot> <orogFile> <ncRef> <ncRoot> <startDate> <endDate>
```

where:

- `<timeRes>` is the time resolution of the pp files, either '3hrly', 'daily' or 'monthly'
- `<ppRoot>` is the path (either relative to the current directory, or full) to the root directory containing the pp files. The expected file naming convention under this root directory is described further below.
- `<orogFile>` is the path (relative or full) to the ancillary UM file containing the orography data associated with the pp files (file typically called 'qrparm.orog').
- `<ncRef>` is the path (relative or full) to a reference netCDF file whose coordinate system is at the desired coarsened resolution (N48) onto which the original high-resolution (N96) data should be regridded.
- `<ncRoot>` is the path (relative or full) to the desired output directory.
- `<startDate>` is in the format YYYYMMDD for 3hrly or daily pp files, or YYYYMM for monthly pp files, and refers to the first day/month (inclusive) to be processed.
- `<endDate>` is in the format YYYYMMDD for 3hrly or daily pp files, or YYYYMM for monthly pp files, and refers to the last day/month (inclusive) to be processed.

It is also possible to see the above information in the terminal by typing:

```
$ ./pp2nc_AOD550_CDN.py --help
```

1.1.3 Dependencies

The script has been designed to run on the JASMIN analysis servers and/or LOTUS (which have access to the pp files that reside in the GASSP/UKCA group workspaces). Running the script will automatically choose the python2.7 interpreter, as it is this version of python that has access to all the scientific packages (e.g. IRIS) installed on JASMIN.

For production runs, it is recommended to use LOTUS (batch computing) rather than run interactively on the JASMIN analysis servers, which can become slow if there are many users running interactively at the same time. To run on LOTUS, the following job submission script example can be modified as required (submit using `"bsub < scriptName.sh"`). If processing 3-hourly or daily pp files, users should allow around 20-30 minutes of wall-clock time for each day to be processed; if processing monthly pp files, allow around 2-3 hours of wall-clock time for each month to be processed.

```
#!/bin/bash
#BSUB -q short-serial
#BSUB -J pp2nc_AOD550_CDN
#BSUB -o pp2nc_AOD550_CDN.out
#BSUB -e pp2nc_AOD550_CDN.err
#BSUB -W 05:00
./pp2nc_AOD550_CDN.py 3hrly /ppRoot/path /orogFile/path /ncRef/path ...
   ... /ncRoot/path 20080701 20080710
```

1.1.4 Output

Running the script will generate two netCDF files (one for AOD and one for CDN) per day (for 3-hourly or daily pp files) or month (for monthly pp files) of processed data. The files will be written to the directory specified by the user through the 'ncRoot' command-line argument. The naming convention of the files is '`[aod550/cdn]_tebaa-tebiz_teafw_pbYYYYMMDD_N48.nc`' if processing 3-hourly pp files, '`[aod550/cdn]_tebaa-tebiz_teafw_paYYYYMMDD_N48.nc`' if processing daily pp files, or '`[aod550/cdn]_tebaa-tebiz_teafw_pmYYYYmon_N48.nc`' if processing monthly pp files. The first part refers to the main variable within the file, the next parts refer to the start, end and median job ids, the `'pb'/'pa'/'pm'` part indicates 3-hourly/daily/monthly model data, the date stamp refers to the date of the data within the file, and the '`N48`' part references the grid resolution. A log file ('`logfile.log`') is also generated, which can be used to keep track of the script's progress during execution, as well as to check for any generated error/warning messages.

1.1.5 Further details

The script expects the input pp files to have the following directory structure/filename convention:

`/ppRoot/<jobid>/<jobid>a.pbYYYYDDMM.pp` for 3-hourly pp files,

`/ppRoot/<jobid>/<jobid>a.paYYYYDDMM.pp` for daily pp files, or

`/ppRoot/<jobid>/<jobid>a.pmYYYYmon.pp` for monthly pp files,

where `mon` is in the format 'jan', 'feb', etc. The main root directory 'ppRoot' is provided as a command line argument.

The script also expects the dimensions of the data within each input file to have certain size and order, as detailed below.

- For 3-hourly pp files:
  - The CDNC field (expected stash ID: m01s38i479) is expected to have dimensions (nt, nz, nlat, nlon) = (8, 52, 145, 192)
  - The six AOD550 'mode' fields (expected stash IDs: m01s02i500 to m01s02i505) are expected to have dimensions (nt, nlat, nlon)=(8, 145, 192)[1]
- For daily pp files:
  - The CDNC field (expected stash ID: m01s38i479) is expected to have dimensions (nz, nlat, nlon)=(52, 145, 192)
  - The six AOD550 'mode' fields (expected stash IDs: m01s02i500 to m01s02i505) are expected to have dimensions (nλ, nlat, nlon) = (6, 145, 192), with wavelengths λ = 380, 440, 550, 670, 860, and 1020 nm (in that order).
- For monthly pp files:
  - The CDNC field (expected stash ID: m01s38i479) is expected to have dimensions (nz, nlat, nlon)=(85, 145, 192)
  - The six AOD550 'mode' fields (expected stash IDs: m01s02i500 to m01s02i505) are expected to have dimensions (nλ, nlat, nlon) = (6, 145, 192), with wavelengths λ = 380, 440, 550, 670, 860, and 1020 nm (in that order).
- The coarse resolution (N48) reference data file ('ncRef') and orography file ('orogFile') are expected to have dimensions (nlat, nlon) = (73, 96)

The 550nm AOD fields are calculated on the fine domain (N96) as the sum over the six 'modes'. The column-integrated CDNC fields (i.e. CDN per m^2) are calculated on the fine domain by first multiplying each CDNC value by its cell height and then summing over all cells within a given model column. The cell heights are currently obtained using IRIS's 'HybridHeightFactory' function, which takes orography data and the model sigma levels as input and gives the altitude bounds of each grid cell as output. However, it is planned to also implement Masaru's alternative approach of using the pressure and theta fields along with the hydrostatic equation; the method to use at execution could be chosen via an optional command

[1]Although the script expects 8 timesteps per day, it has also been designed to deal with the case where there are only 7 timesteps in the six AOD modes. This is because some of the pp files for the first day of a calendar month have been found to have missing data for first AOD timestep (00:20). When this occurs, the 00:20 field is re-inserted into the output netCDF file and filled with missing values (NaN) so that the grid remains regular (as is necessary). As an example, there are 16 (out of 235) PPE members with a missing first timestep on 2008-07-01

line flag.

The fine domain data is then regridded onto the coarser (N48) domain using IRIS's 'regrid' function, where the coarse grid coordinates are extracted from the 'ncRef' file (supplied via a command-line argument). The 'Linear' regridding method is used and is currently the preferred method (over the alternative 'AreaWeighted' method) as it is believed that the UM output data represents quantities at the grid cell centres rather than a mean over the grid cell volume.

Efforts have been made to capture as many potential errors as possible in a graceful manner (i.e. with descriptive error/warning messages written to the log file). These include:

- Checking that the paths given in the command-line arguments exist
- Checking that the date stamps given in the command-line arguments are in the expected format, with the start date preceding or equalling the end date.
- Checking that the data cubes can be loaded by IRIS (if this fails, it is most likely due to a missing/unexpected STASH codes)
- Checking that all the relevant data cube dimensions are as expected.

1.2 collocate_UM_AERONET.py

1.2.1 Purpose

For a given month, use cis to collocate AOD550 UM data from the 3-hourly UKCA26AER PPE set with AERONET v3 data (all stations) and take monthly averages.

1.2.2 Usage

The script can be run from the command line as follows:

```
$ ./collocate_UM_AERONET.py <mon> <ppRoot> <AERONETRoot> <outDir>
```

where:

- `<mon>` is the month to be processed in the form YYYYMM.
- `<ppRoot>` is the absolute or relative path to root directory containing the UM pp files. The expected file naming convention under this root directory is described further below.
- `<AERONETRoot>` is the absolute/relative path to root directory containing the (processed) AERONET v3 files. The expected format and file naming conventions under this root directory are described further below.
- `<outDir>` is the absolute/relative path to desired output directory.

It is also possible to see the above information in the terminal by typing:

```
$ ./collocate_UM_AERONET.py --help
```

1.2.3 Dependencies

A cis 'plugin' called cis_plugin_AERONETv3nc.py, which is version controlled in the same git repository as the main script, has been written to read in processed AERONET v3 data (at the time of writing, cis can only read v2 data by default). More details about the format of the processed AERONET v3 are given further below. For the plugin to be visible to cis, an environment variable called CIS_PLUGIN_HOME must be set that points to the directory containing the plugin script. To do this, issue the following command in the shell (or add it to your .bashrc file):

```
$ export CIS_PLUGIN_HOME = /path/to/dir/containing/plugin/script/
```

The main script has been designed to run on the JASMIN analysis servers and/or LOTUS (which have access to the pp files that reside in the GASSP/UKCA group workspaces). Running the script will automatically choose the python2.7 interpreter, as it is this version of python that has access to all the scientific packages (e.g. cis) installed on JASMIN.

For production runs, it is recommended to use LOTUS (batch computing) rather than run interactively on the JASMIN analysis servers, which can become slow if there are many users running interactively at the same time. To run on LOTUS, the following job submission script example can be modified as required (submit using `"bsub < scriptName.sh"`). Users should allow around 8-9 hours of wall-clock time for the entire PPE set (235 jobs) to be processed.

```
#!/bin/bash
#BSUB -q short-serial
#BSUB -J collocate_UM_AERONET
#BSUB -o collocate_UM_AERONET.out
#BSUB -e collocate_UM_AERONET.err
#BSUB -W 09:00
./collocate_UM_AERONET.py 200807 ./ppFiles/ ./AERONETFiles/ ./output/
```

1.2.4 Output

Running the script will generate two sets of netCDF files:

- `<outDir>/UM_nc_files/aod550_total_<jobid>_<YYYYMM>.nc` – These files contain the AOD550 fields extracted from the raw UM pp files (after summing over six 'modes'), for the given month, in gridded nc format.
- `<outDir>/col_mav_files/aod550_total_<jobid>_pb<YYYYMM>_col_mav.nc` – These files contain the result of collocating the UM AOD550 fields onto the AERONET dataset (in time and space) and subsequently taking the monthly-average at each station, in ungridded nc format.

A log file ('`logfile.log`') is also generated, which can be used to keep track of the script's progress during execution, as well as to check for any generated error/warning messages.

1.2.5 Further details

The script expects the input pp files to have the following directory structure/filename convention:

`<ppRoot>/<jobid>/<jobid>a.pb<YYYYDDMM>.pp`

where the main root directory 'ppRoot' is provided as a command line argument. Presently, the relevant pp files reside on JASMIN in the following directory (i.e. 'ppRoot' can be set to this):

`/group_workspaces/jasmin2/ukca/vol1/myoshioka/um/Dumps/`.

The six 'mode' fields within these pp files that are summed to give total AOD550 are expected to have stash IDs m01s02i500 to m01s02i505, and it must be possible to concatenate these summed fields (in time) over the entire month using iris's concatenate_cube function, i.e. each file should have the same dimensions for latitude and longitude.

The processed AERONET files are expected to have the following naming convention:

`<AERONETRoot>/AOD_440_<station_name>_<YYYY>-<MM>_v3.nc`

where the main root directory 'AERONETRoot' is provided as a command line argument. Presently, the relevant AERONET files reside on JASMIN in the following directory (i.e. 'AERONETRoot' can be set to this):

`/group_workspaces/jasmin2/crescendo/Data/AERONET/AOT/ver3/LEV20/Monthly/`.

Each file is expected to contain ungridded netCDF data with a single dimension (typically called 'obs') with (at-least) the following variables included and sized on that dimension: latitude, longitude, time, AOD_440. Note that since there is one file per station, the latitudes and longitudes will all be the same repeated value. The time variable is expected to be in units of "days since 1990-01-01 00:00:00" on a Gregorian calendar. Masaru has created files in this format from the original AERONET v3 dataset (one large file over all stations/times/variables).

The main body of work in this script is performed using cis functions. The cis Python library (API) has been used, rather than the command-line utilities, so that multiple operations can be performed in one single script and without having to write to disk in between commands. The specific cis functions used are as follows:

- `cis.read_data(`*`filename,variable,product=None`*`)` – Used to read both the UM nc files and the AERONET nc files. 'product' is set to "AERONETv3nc" in the case of AERONET files, which ensures use of the new plugin.
- `AERONETData.subset(`*`time=[start,end]`*`)` – Used to subset the data in each AERONET file in time (i.e. extract just the given month).
- `UMdata.collocated_onto(`*`AERONETData,how="lin"`*`)` – Used to collocate the UM data onto the subsetted AERONET data (in space and time) using a linear interpolation method (as is suitable for collocating gridded data onto ungridded data)
- `data.aggregate(`*`how="moments",t=[start,end,step]`*`)` – Used to calculate the monthly mean

(and standard deviation) of the collocated dataset.

For each job (member of the PPE set), collocated monthly-averaged UM data are calculated for each AERONET station location separately and stored in a pandas data-frame over all stations. The data in this data-frame is then written to a netCDF file, and the script then moves on to the next job. There are therefore as many collocated monthly-averaged output nc files as there are PPE members (235). As well as the monthly average ("aod550"), these nc files also include the standard deviation ("aod550_std_dev") and the number of points used in the calculation of the monthly average ("aod550_num_pts").