# Use of generate\_days\_netcdf\_metsensors.py

This code generates netCDF files compatible with CF1.6-AMF1.0 from a wide range of Chilbolton Atmospheric Observatory meteorological sensors. Data from many (but not all) are recorded using Campbell Scientific CR1000X dataloggers.

There are several modules to the code:

generate\_days\_netcdf\_metsensors.py which in turn calls:

metsensors\_ncas.py

module\_data\_object\_python3.py

module\_distrometer\_format5.py

metsensors\_ncas.py contains different functions which produce netCDF files for different instruments or groups of instruments. These are selected according to the inputs to generate\_days\_netcdf\_metsensors.py which are described in the following quick start guide.

There are also functions with metsensors\_ncas.py which support the processing of these files. These are described at the end of this document.

The codes run under Python version 2 or 3, but depend on the import of some external libraries such as PyArt which are available at Chilbolton as part of the conda chil\_3\_8 installation. The “#!” first line of the code looks for that installation so the code has been developed using version 3.8.1.

These Python scripts can all be installed in 1 directory, or different directories. If they are in different directories, the command

export PYTHONPATH=/*directory\_where\_code\_stored*/

must be entered at the command line for each directory used.

The user can remove the need to enter this command before running the code by including it in their .bash\_profile file.

## Quick start guide to producing netCDF files from met. data

1. Open a Unix or Linux command window (a run-time environment such as Cygwin can be used if preferred, or a terminal window such as PuTTY).
2. Enter the command **conda activate chil\_3\_8**
3. Connect to the directory where the generate\_days\_netcdf\_metsensors.py code is located (unless it is in your $PATH command, it which can you can run it from any directory).
4. Enter **./generate\_days\_netcdf\_metsensors.py -s *yyyymmdd\_start* -e *yyyymmdd\_end* –x *option*.**

*yyyymmdd\_start* = first date for which you want to produce files

*yyyymmdd\_end* = last date for which you want to produce files

*option* = type of data file you wish to produce, as shown in table 1 below

## Datasets which can be processed

Both codes call a range of functions depending on which met. sensor is to be processed. The purpose of each code is as shown in the table below, together with the meaning of each option, the data file(s) produced, the details of each instrument and the local ID of that instrument. The local ID gives the root name of the corresponding correction file for that instrument in the /data/netCDF/corrections directory (see … section)

| **Option** | **Instruments processed** | **Data file produced** | **Instrument manufacturer, model** | **Chilbolton instrument ID** |
| --- | --- | --- | --- | --- |
| 1 | Pluvio weighing raingauge from dedicated text data file | ncas-rain-gauge-4\_cao\_yyyymmdd\_precipitation\_v1.0.nc | OTT Pluvio 200 | pldc\_ch |
| 2 | Chilbolton met. sensors, drop-counting and tipping bucket raingauges | ncas-temperature-rh-1\_cao\_yyyymmdd\_surface-met\_v1.0.nc  ncas-pressure-1\_cao\_yyyymmdd\_surface-met\_v1.0.nc  ncas-anemometer-2\_cao\_yyyymmdd\_surface-met\_v1.0.nc  ncas-rain-gauge-1\_cao\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-2\_cao\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-3\_cao\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-5\_cao\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-9\_cao\_yyyymmdd\_precipitation\_v1.0.nc | Vaisala HMP155A T/RH  Vaisala PTB110  Vector Instruments cup/vane A100H, W200P  Turf wall drop-counting raingauge  Standard drop-counting raingauge  Large collector drop-counting raingauge  Munro tipping bucket raingauge  Flux compound drop-counting raingauge (not an NCAS instrument, name not approved by them) | oatnew\_ch, rhnew\_ch  QFE\_ch  ws\_ch, wd\_ch  rg001dc\_ch  rg006dc\_ch  rg008dc\_ch  rg004dc\_ch  rg009dc\_ch |
| 3 | Chilbolton raingauges from format5 files (use before 20200728) | ncas-rain-gauge-1\_cao\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-2\_cao\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-3\_cao\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-5\_cao\_yyyymmdd\_precipitation\_v1.0.nc | Turf wall drop-counting raingauge  Standard drop-counting raingauge  Large collector drop-counting raingauge  Munro tipping bucket raingauge | rg001dc\_ch  rg006dc\_ch  rg008dc\_ch  rg004dc-ch |
| 4 | Chilbolton broadband radiometers from format5 files | ncas-radiometer-1\_cao\_yyyymmdd\_radiation\_v1.0.nc  ncas-radiometer-2\_cao\_yyyymmdd\_radiation\_v1.0.nc  ncas-radiometer-3\_cao\_yyyymmdd\_radiation\_v1.0.nc  ncas-radiometer-4\_cao\_yyyymmdd\_radiation\_v1.0.nc | CM21 downwelling pyranometer  CMP21 diffuse downwelling pyranometer  CG4 downwelling longwave pyrgeometer  CHP1 direct pyrheliometer (all Kipp & Zonen) | pyrCM21\_ch  pyr\_CMP21\_ch  [[1]](#footnote-1)pyrCG4\_ch  pyr\_CP1\_ch, pyr\_CP1\_T\_ch |
| 5 | Sparsholt tipping bucket and drop-counting raingauges | ncas-rain-gauge-6\_cao-sparsholt\_yyyymmdd\_precipitation\_v1.0.nc  ncas-rain-gauge-7\_cao-sparsholt\_yyyymmdd\_precipitation\_v1.0.nc | Standard drop-counting raingauge  Munro tipping bucket raingauge | rg001dc\_sp  rg002tb\_sp |
| 6 | Chilbolton RD-80 disdrometer from format5 files | ncas-disdrometer-1\_cao\_yyyymmdd\_precipitation\_v1.0.nc  (up to 20210505)  ncas-disdrometer-2\_cao\_yyyymmdd\_precipitation\_v1.0.nc  (from 20210506) | Distromet RD-80 disdrometer | disdrom\_ch |
| 7 | Sparsholt RD-80 disdrometer from format5 files | ncas-disdrometer-2\_cao-sparsholt\_yyyymmdd\_precipitation\_v1.0.nc  (up to 20190624, currently not operational) | Distromet RD-80 disdrometer | disdrom\_sp |
| 8 | Chilbolton present weather sensor from dedicated text file | ncas-present-weather-1\_cao\_yyyymmdd\_present-weather\_v1.0.nc | Campbell Scientific PWS100 present weather sensor | [[2]](#footnote-2)pws100 (for visibility and synoptic code), pws100\_rain (for hydrometeor size and velocity details, rain rate and accumulation), pws100\_met (for temperature and humidity) |
| 9 | Chilbolton broadband radiometers | ncas-radiometer-1\_cao\_yyyymmdd\_radiation\_v1.0.nc  ncas-radiometer-2\_cao\_yyyymmdd\_radiation\_v1.0.nc  ncas-radiometer-3\_cao\_yyyymmdd\_radiation\_v1.0.nc  ncas-radiometer-4\_cao\_yyyymmdd\_radiation\_v1.0.nc | CM21 downwelling pyranometer  CMP21 diffuse downwelling pyranometer  CG4 downwelling longwave pyrgeometer  CHP1 direct pyrheliometer (all Kipp & Zonen) | pyrCM21\_ch  pyr\_CMP21\_ch  pyrCG4\_ch  pyr\_CP1\_ch, pyr\_CP1\_T\_ch |
| 10 | Chilbolton net flux radiometer | ncas-radiometer-5\_cao\_yyyymmdd\_radiation\_v1.0.nc | CNR4 net flux radiometer (Kipp & Zonen) | cnr4\_dsw  cnr4\_dlw  cnr4\_usw  cnr4\_ulw  (‘d’ = downwelling, ‘u’ = upwelling, ‘sw’ = shortwave, ‘lw’ = longwave |
| 11 | Chilbolton sonic anemometer | ncas-sonic-5\_cao\_yyyymmdd\_mean-winds\_v1.0.nc | Metek USA-1 | sonic |

**Table 1: Dataset details**

The raw data read into the files are from Campbell Scientific CR1000X datalogger files unless they are specified otherwise. Describe further in separate document… The first alternative format is “format5” a format developed at CAO and used for data acquisition from many analog and pulse counting instruments for many years. It is a text format for all except RD-80 disdrometers where it is a binary format. Details (file formats for analog and bbrad) can be found in the files … stored in this repository. The second alternative format is for the Pluvio weighing raingauge, where the format is based on that specified in the instrument manual, located at https://gate.chobs.rl.ac.uk/documentation/calibration\_records/user\_guides/Pluvio2\_E.pdf. The third alternative format is for the Campbell Scientific PWS100 present weather sensor, where the format is based on that specified in the instrument manual, located at …(github address, or copy/paste details into a separate document)

## Dates when sensors were moved to Campbell Scientific dataloggers

|  |  |  |  |
| --- | --- | --- | --- |
| **Instrument** | **Date of move** | **Campbell Scientific datalogger and filename root** | **Previous DAQ and filename root** |
| HMP155A temperature/relative humidity sensor | 10/02/2020 | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | Receive cabin microlink, chan |
| Vector Instruments cup and vane anemometer | 12/04/2019 (failure of wind measurements in microlink precipitated move) | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | Receive cabin microlink, chan |
| Vaisala PTB110 pressure sensor | None – starts 11/04/19. | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | None. Predecessor sensor was logged using receive cabin microlink, chan |
| Turf wall drop-counting raingauge rg001dc\_ch | 28/07/20 | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | Receive cabin microlink, chan |
| Standard drop-counting raingauge rg006dc\_ch | 28/07/20 | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | Receive cabin microlink, chan |
| Low rate (200 mm collector) drop-counting raingauge rg008dc\_ch | 28/07/20 | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | Receive cabin microlink, chan |
| Pluvio weighing raingauge pldc\_ch | None | None | Serial interface to Moe, receive cabin, pldc |
| Tipping bucket raingauge rg004tb\_ch | 28/07/2020 | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | Receive cabin microlink, chan |
| Sparsholt tipping bucket raingauge rg002tb\_sp | 08/08/2019 | S/N 6677  CR1000XSeries\_Sparsholt | Sparsholt DAQ, spww |
| Sparsholt drop-counting raingauge rg001dc\_sp | 08/08/2019 | S/N 6677  CR1000XSeries\_Sparsholt | Sparsholt DAQ, spww |
| Chilbolton RD-80 disdrometer ch\_distrom | None | None | Serial interface to Marvin DAQ computer, receive cabin, chds |
| Sparsholt RD-80 disdrometer sp\_distrom | None | None | Serial interface to Rhubarb DAQ computer, Sparsholt, spds |
| Downwelling pyranometer (visible radiation) pyrCM21\_ch | 20220114 | S/N 6676  CR1000XSeries\_Chilbolton2\_  Rxcabinmet2 | Agilent datalogger connected to Moe, receive cabin, chpy |
| Downwelling diffuse radiation only pyranometer pyr\_CMP21\_ch | 20190304 (as a test in S/N 6678), properly shaded from 20211122 | S/N 6676  CR1000XSeries\_Chilbolton2\_  Rxcabinmet2 | Agilent datalogger connected to Moe, receive cabin, chpy |
| Downwelling pyrgeometer (IR radiation) pyrCG4\_ch | 20220114 | S/N 6676  CR1000XSeries\_Chilbolton2\_  Rxcabinmet2 | Agilent datalogger connected to Moe, receive cabin, chpy |
| Direct visible radiation pyrCP1\_ch | 20211014, not well aligned until 20211104 | S/N 6676  CR1000XSeries\_Chilbolton2\_  Rxcabinmet2 | Agilent datalogger connected to Moe, receive cabin, chpy |
| Flux compound drop-counting raingauge | None | S/N 6678  CR1000XSeries\_Chilbolton\_  Rxcabinmet1 | None  Being collected for project rather than AMOF submission, filename not confirmed with AMOF. |

**Table 2: Data recording details for meteorological sensors now logged using CR1000X dataloggers**

Raw Campbell Scientific datalogger ongoing files are mirrored from the Windows DAQ “Grape” (to which files are regularly download from the dataloggers) to the Chilbolton data server at /data/range/mirror\_grape\_loggernet. Filenames are as indicated in table 2.

Format 5, Pluvio raingauge and PWS100 present weather sensor files are mirrored to the following directories:

|  |  |
| --- | --- |
| **File name and instrument** | **Location** |
| chan Chilbolton met. sensor and raingauge | /data/range/mirror\_marvin\_home2/ranged/analog |
| chds Chilbolton disdrometer | /data/range/mirror\_marvin\_home2/ranged/distrom |
| chpy Chilbolton broadband radiometers | /data/range/mirror\_moe\_home2/ranged/broadband\_radiometer/pyr |
| pldc Pluvio weighing raingauge | data/range/mirror\_moe\_home2/ranged/pluvio |
| pws Campbell Scientific PWS100 | /data/campbell\_PWS/mirror\_cl51sky\_campbell\_data/PWS100 |
| spww Sparsholt raingauges | /data/Sparsholt/mirror\_rhubarb\_home2/ranged/ww\_raingauge |
| spds Sparsholt disdrometer | /data/Sparsholt/mirror\_rhubarb\_home2/ranged/distrom |

**Table 3: Data recording details for meteorological sensors not logged using CR1000X dataloggers**

Campbell Scientific PWS100 data files are recorded directly via Ethernet from the instrument to the above location. Other files are recorded on the DAQ computers Marvin, Moe and Rhubarb.

Each datalogger ongoing file has an equivalent daily files directory as shown below. Daily files are described further in the section “Reading from Campbell datalogger files“.

|  |  |
| --- | --- |
| **Ongoing datalogger file** | **Corresponding daily files directory** |
| CR1000XSeries\_Chilbolton\_Rxcabinmet1.dat | /data/range/daily\_met/cr1000x\_rxcabin\_1 |
| CR1000XSeries\_Chilbolton2\_Rxcabinmet2.dat | /data/range/daily\_met/cr1000x\_rxcabin\_2 |
| CR1000XSeries\_Sparsholt.dat | /data/range/daily\_met/cr1000x\_sparsholt\_1 |

**Table 4: Daily data files for each datalogger**

The Campbell Basic codes which are currently running in each datalogger are shown in the table below. They can be found on Grape in C:\Campbellsci\SCWin

|  |  |
| --- | --- |
| **Ongoing datalogger file** | **Current Campbell Basic data acquisition code** |
| CR1000XSeries\_Chilbolton\_Rxcabinmet1.dat | rxcabin1\_20210203.CR1X |
| CR1000XSeries\_Chilbolton2\_Rxcabinmet2.dat | bbrad\_2.CR1X |
| CR1000XSeries\_Sparsholt.dat | sparsholt\_dc\_tb.CR1X |

**Table 5: Daily data files for each datalogger**

## How the netCDF production codes work

1. The user input of start date, end date and option (which instrument(s) data to process) to generate\_days\_netcdf\_metsensors.py determines which part of the code to call and in turn which raw file is read, as shown in the quick start guide above. This determines the function called by the code.
2. A check is made of whether the relevant data file exists. All files except the Campbell datalogger ones are daily. The Campbell datalogger files are continuous, except that the files as saved on Grape (where they are downloaded to from the dataloggers) must be reduced in size periodically to avoid them getting too large.
3. For all except Campbell dataloggers, the relevant daily file is opened and read. If no daily file is found, the code terminates. A check for relevant data in data files from the previous day and following day is also made.
4. For Campbell dataloggers, the current data file is read and scanned for data which are timestamped with the required date. If that fails, daily datalogger files which were written by a previous run of the code for this day are searched. If no data are found, the code terminates. Further details on how Campbell datalogger files are read is given in the section “Reading from Campbell datalogger files”.
5. In both cases the period to be included in a daily netCDF file is from any time greater than 00:00:00 on that day, up to and including 00:00:00 on the following day. In most cases data are timestamped at the end of each (usually 10 second) time period that they cover.
6. A check is made for corrections in the relevant corrections file for each sensor in the /data/netCDF/corrections directory. Two types of correction are given:
   1. HOLDCAL. This signifies that data are kept in the file, but there is a suspicion that the value may not be genuine. This may occur for example when a rain gauge produces a count when there was not thought to be any rain or when the HMP155A temperature and relative humidity sensor was performing its daily purge cycle.
   2. BADDATA. This signifies a more serious error with the instrument, such as a rain gauge being blocked, or any instrument being broken or away for repair. The data are replaced with the “fill value” for that variable.

If a sensor reports data from more than 1 sensor, there is more than 1 variable written to a netCDF file. In this case, it may be possible for 1 sensor to function correctly while another shows a fault. In this case, table 1 shows more than 1 instrument ID, and therefore there is a separate correction file for each sensor.

The quality flag associated with each variable is changed to reflect these corrections. The meaning of each flag value is described in the netCDF file for each instrument.

1. Data are now ready to be written to netCDF files. Filenames are generated from the information contained in the raw data definition files described in the section “Raw data definition”. The file format and metadata are derived from .yaml files associated with each instrument, stored in the same directory as the Python code files. The .yaml files include the instrument name to which they relate. Those used for Chilbolton drop-counting and tipping bucket raingauges for the period 01/04/2020 to 28/07/2020 (before those gauges were moved to the receive cabin CR1000X datalogger 1) have a “ulink” component to the name. Those used for Chilbolton broadband radiometers for the period 01/04/2020 to 12/10/2021 (before those gauges were moved to the receive cabin CR1000X datalogger 2) have an “agilent” component to the name. The end date for CM21 and CG4 with the Agilent dataloggers is 12/10/2021. The CMP21 and CHP1 had been out of action due to the failure of the solar tracker since December 2018 so for them the restart of good quality data, with the solar tracker correctly aligned, was 22/1/2021, using the CR1000X data logger. The process for writing the data files using module\_data\_object\_python3 was written by David Hooper and is described at <https://github.com/dahooper/metadata-from-template/blob/master/README.md>. Files are currently written as “v1.0”. This version number refers to the version of generate\_days\_netcdf\_metsensors.py and its functions. If the software was to be upgraded and new data files released, they would be named “v1.1” for a minor upgrade or “v2.0” for a more major upgrade. These upgrades are defined by NCAS.
2. If more than 2 variables are required in a netCDF file produced from CR1000X datalogger data, the code will need to be edited to generate more varmin*x*, varmax*x* values to use as substitutions with the yaml files. This would be an easy edit to make for a Python user.
3. Quicklook graphs of the netCDF files are produced and written to directories under /data/amof-netCDF/graphs. These directories are split by year and month. This allows the use of scripts which use Webmagick to produce monthly quicklook pages which can easily be viewed using a web browser. The scripts to produce these quicklooks are described further in a separate document.

## Reading from Campbell datalogger files

The table below shows the typical contents of a CR1000X datalogger file, in this case from receive cabin datalogger no. 1.

|  |  |
| --- | --- |
| **Line** | **Contents** |
| Header 1 | "TOA5","CR1000XSeries\_Chilbolton","CR1000X","6678","CR1000X.Std.02.00","CPU:rxcabin1\_20210203.CR1X","23733","Rxcabinmet1" |
| Header 2 | "TIMESTAMP","RECORD","BattV\_Avg","PTemp\_C\_Avg","Air\_T\_Avg","RH\_Avg","BP\_mbar\_Avg","WS\_Avg","WD\_Avg","SlrW\_Avg","rg001dc\_ch\_Tot","rg006dc\_ch\_Tot","rg008dc\_ch\_Tot","rg009dc\_ch\_Tot","rg004tb\_ch\_Tot" |
| Header 3 | "TS","RN","Volts","DegC","mV","mV","mbar","mV","mV","W/m^2","Counts","Counts","Counts","Counts","mm" |
| Header 4 | "","","Avg","Avg","Avg","Avg","Avg","Avg","Avg","Avg","Tot","Tot","Tot","Tot","Tot" |
| Data 1 | "2021-08-01 00:00:00",1542540,12.13,22.43,2695,4536,1001.113,228.4,1190,"NAN",0,0,0,0,0 |
| Data 2... | … |

**Table 5: format of a data file from a CR1000X datalogger**

So data files are human-readable and comma separated. These are properties of the TOA5 file, the default format. More details can be found in the CR1000X manual.

Header line 1 gives some details of the file including which logger it was collected from and the data acquisition code that was running on it (a \*.CR1X file). The data acquisition code is written in Campbell Scientific’s CR-Basic format. This can be generated using a wizard within the PC200W or Loggernet software or written in a text editor (the one within Loggernet is most convenient). Often it is helpful to produce the initial code using the wizard, then make minor changes such as adding new measurements by manual editing of the code.

Header line 2 describes the columns in the following data. The “RECORD” is a number which increases by 1 for each record in the file. “PTemp\_C\_Avg” is the internal temperature of the datalogger, which reflects the temperature of the local environment quite accurately. The following names indicate which meteorological measurement is in that column.

Header line 3 gives the units of each measurement. Within the datalogger program there is the option to provide calibration factors for each measurement. In general I have not done this, as if they change it makes it necessary to change the netCDF production code to reflect this change. The only measurements shown above which are calibrated are “BP-mbar\_Avg” which is from the PTB110 air pressure sensor and is reported in mbar, SIrW\_Avg which is an irradiance measurement not currently in use and “rg004tb\_ch\_Tot” which is the tipping bucket gauge total, reported in 0.2 mm intervals rather than counts”.

Header line 4 shows the type of measurement in each column, usually average over the acquisition time period (“Avg”) for analog measurements and total counts over the time period (“Tot”) for pulse counting measurement.

Data lines then follow, following that format. If measurements are outside the measurement range set for the channel, “NAN” is reported. A maximum of +/-5 volts can be recorded in an analog channel.

When data from the required date are found in the ongoing datalogger file they are written to a data array in the code, and also written to a daily text file in /data/range/daily\_met/cr1000x\_rxcabin\_1, /data/range/daily\_met/cr1000x\_rxcabin\_2 or /data/range/daily\_met/cr1000x\_sparsholt\_1 if that file hasn’t already been written (specify what instruments in which). By doing this we ensure that daily files are saved, and thereby the data remain available after the continuous datalogger files are reduced in size. These daily files have the same format as the original continuous file. If data from that date are not found in the continuous datalogger file, a check is made for the availability of a daily file. If that is present, it is read instead. It is most likely that the code will need to search for a daily file if you are processing older data, typically more than a few months old.

When instruments are added to or removed from a Campbell datalogger, the number of columns in each line of the data file will change. We use a text file to specify how the number of channels and which ones should be read in the next section.

## Raw data definition

The files are stored in /data/netCDF/corrections (may change).

### Chilbolton receive cabin 1 definition file

The file for receive cabin datalogger 1 is chilbolton\_rxcabin\_1.txt which has the following contents:

No. measurements: 11

Start column in text file: 5

oatnew\_ch,1,0.02,233.15,ncas-temperature-rh-1,air\_temperature,qc\_flag\_air\_temperature

rhnew\_ch,1,0.02,0.0,ncas-temperature-rh-1,relative\_humidity,qc\_flag\_relative\_humidity

QFE\_ch,1,1.0,0.0,ncas-pressure-1,air\_pressure,qc\_flag

ws\_ch,1,0.01,0.0,ncas-anemometer-2,wind\_speed,qc\_flag\_wind\_speed

wd\_ch,1,0.045,180.0,ncas-anemometer-2,wind\_from\_direction,qc\_flag\_wind\_from\_direction

pyr\_CMP21\_ch,0,1.0,0.0,ncas-radiometer-2,diffuse\_downwelling\_shortwave\_flux\_in\_air,qc\_flag

rg001dc\_ch,1,0.00331,0.0,ncas-rain-gauge-1,thickness\_of\_rainfall\_amount,qc\_flag

rg006dc\_ch,1,0.00329,0.0,ncas-rain-gauge-2,thickness\_of\_rainfall\_amount,qc\_flag

rg008dc\_ch,1,0.00186,0.0,ncas-rain-gauge-3,thickness\_of\_rainfall\_amount,qc\_flag

rg009dc\_ch,1,0.00325,0.0,ncas-rain-gauge-9,thickness\_of\_rainfall\_amount,qc\_flag

rg004tb\_ch,1,1.0,0.0,ncas-rain-gauge-5,thickness\_of\_rainfall\_amount,qc\_flag

This definition file matches the example datalogger file shown in the section “Reading from Campbell datalogger files”.

To make changes to what files are written by generate\_days\_netcdf\_metsensors.py option 2 or 5, edit this file, having saved a copy with a different name first! Note that the format must remain the same, with the same comma delimiters and no extra spaces.

Line 1 gives the number of measurements in the datalogger file, ignoring the first 4 columns (time stamp, record number, battery voltage and internal temperature.

Line 2 shows the start column, in this case 5 (i.e. ignoring the first 4 columns)

Lines 3 onwards show the details for each of the measurements (the number of lines must match the number given in line 1). The contents of each line are:

1. Instrument ID (and name of corrections file)
2. 1/0, where 1 indicates to include the measurement when writing netCDF files, 0 indicates to exclude it.
3. Multiplier to scale the data
4. Offset to apply to the data after the multiplier has been applied
5. Name of netCDF file to write data to
6. Name of variable to write data to within the netCDF file
7. Name of the associated quality control flag of that variable within that data file

### Chilbolton receive cabin 2 definition file

The file for receive cabin datalogger 2 is chilbolton\_rxcabin\_2.txt which has the following contents:

No. measurements: 11

Start column in text file: 3

pyrCM21\_ch,1,92.166,0.0,ncas-radiometer-1,downwelling\_shortwave\_flux\_in\_air,qc\_flag

pyr\_CMP21\_ch,1,115.075,0.0,ncas-radiometer-2,diffuse\_downwelling\_shortwave\_flux\_in\_air,qc\_flag

pyrCP1\_ch,1,126.103,0.0,ncas-radiometer-4,direct\_downwelling\_shortwave\_flux\_in\_air,qc\_flag\_direct\_downwelling\_shortwave\_flux\_in\_air

pyrCG4\_ch,1,72.78,0.0,ncas-radiometer-3,downwelling\_longwave\_flux\_in\_air,qc\_flag\_downwelling\_longwave\_flux\_in\_air

pyrCG4\_ch,1,1.0,0.0,ncas-radiometer-3,body\_temperature,qc\_flag\_body\_temperature

pyrCG4\_ch,0,1.0,0.0,ncas-radiometer-3,body\_temperature\_intermediate,qc\_flag\_body\_temperature

pyrCG4\_ch,0,1.0,0.0,ncas-radiometer-3,body\_temperature\_intermediate,qc\_flag\_body\_temperature

pyrCG4\_ch,0,1.0,0.0,ncas-radiometer-3,body\_temperature\_intermediate,qc\_flag\_body\_temperature

pyrCP1\_T\_ch,1,1.0,0.0,ncas-radiometer-4,body\_temperature,qc\_flag\_body\_temperature

pyrCP1\_T\_ch,0,1.0,0.0,ncas-radiometer-4,body\_temperature\_intermediate,qc\_flag\_body\_temperature

pyrCP1\_T\_ch,0,1.0,0.0,ncas-radiometer-4,body\_temperature\_intermediate,qc\_flag\_body\_temperature

This file includes the calibration factors for each radiometer. These should be changed each time a radiometer is re-calibrated by the manufacturer. When these radiometers were logged using the Agilent datalogger, writing files using the “format5” in-house format, the calibration values were stored in files named /data/netCDF/corrections/*instrument-name*\_calibration.txt, but these files are not used with CR1000X data files as the calibration values are more conveniently provided in the raw definition files. For the temperature sensors on the CG4 and CHP1, the voltage measured across the standard 10k voltage bridge resistor is read from the raw data file. This is converted to thermistor resistance and then temperature using the resistance-temperature relationship derived in-house from the manual T-R data (for the CG4) or the relationship given in the manual (CHP1). This is done in the generate\_netcdf\_met function of metsensors\_ncas.py.

### Sparsholt definition file

The file for the Sparsholt datalogger is sparsholt\_1.txt which has the following contents:

No. measurements: 2

Start column in text file: 6

rg001dc\_sp,1,0.00335,0.0,ncas-rain-gauge-7,thickness\_of\_rainfall\_amount,qc\_flag

rg002tb\_sp,1,1.0,0.0,ncas-rain-gauge-6,thickness\_of\_rainfall\_amount,qc\_flag

## Reducing the size of ongoing Campbell datalogger files

The CR1000X dataloggers produce ongoing text files which gradually increase in size. This makes generate\_days\_netcdf\_metsensors.py progressively run more slowly, as there are more lines to search in the file for the required date string. The interval at which you reduce the size of the files is arbitrary, but I recommend at least every 3 months.

1. Check that daily met files have been written and that they are not zero bytes in size. They can be empty files if no data was found in the ongoing file. If they are empty, run generate\_days\_netcdf\_metsensors.py again, ensuring first that data are available in the ongoing file for that day.
2. The ongoing file must be edited on Grape, the Windows machine located in the control room to which data are mirrored from the dataloggers. Login to Grape as range.
3. Choose the correct data file to edit, as specified in table 2. If it’s a file which is still being written, it should have been updated already during the current day. Ongoing data files are located in C:\Campbellsci\LoggerNet.
4. Copy that file to a new name, for example adding today’s date to the end of the name. This ensures nothing is lost.
5. Edit the file in Notepad. Take care not to do this when data are about to be automatically collected from the datalogger, or those most recent data may be lost. Leave the header lines in place and remove the lines that you wish. I recommend keeping any data that you still wish to convert to netCDF files again. This implies keeping all of the current month, and possibly the previous month. Ensure that the first line in the file is from a time of 00:00:10 on the required day.
6. Ensure you have left no blank lines or incomplete lines in the file.
7. Save the file.
8. When this directory on Grape is next mirrored to /data/range/mirror\_grape\_loggernet, these changes will be reflected in the files there.

## Other functions within metsensors\_ncas.py

### generate\_netcdf\_common

The numeric version of the day is passed to this function. It returns the date in various formats which are used later in the code.

### generate\_netcdf\_datetimeinfo

The numeric versions of day, month and year, the number of data points and the time in seconds are passed to this function. It returns various date and time arrays in the format required for the output netCDF file.

### load\_netcdf\_corrections

The numeric day value, “chids” instrument identifier, number of variables to be written in this netCDF file, the data time interval, the time array in seconds and the data values array that has been read are passed to this array. It reads the corrections file in /data/netCDF/corrections defined by the chids identifier and loads any corrections that apply to the current day. It sets up a quality flag array of the same dimensions as the data values array with initial values of 1 (which indicates good data) then uses the corrections in the file (BADDATA, or sometimes HOLDCAL for raingauges) to change the value of the quality flag. BADDATA means that the data values are set to the fill value. HOLDCAL is only used for raingauges and flags points with a positive rain measurement that were unlikely to be rain as suspect. When the raingauge\_click\_plots.py code is used to highlight parts of a day where rain counts were unlikely to be rain, they are assigned a HOLDCAL correction which is then read by this function.

If the data being processed are from the Vaisala HMP155 temperature and relative humidity sensor, this function also calls the hmp\_purge\_times function, described below, so that a quality flag can also be applied to times when that sensor was going through a purge cycle.

It returns the quality flag array and an array of the valid minimum and maximum values for each instrument in the data values array. These valid minimum and maximum values exclude data values where the quality flag is not equal to one. They are required for the netCDF file.

### hmp\_purge\_times

The input to this function is the numeric day. It reads the /data/netCDF/corrections/hmp155\_purgetime.txt file to determine the range of times for that day when the instrument data should be considered unreliable to the purge cycle and passes those start and end times to the load\_netcdf\_corrections function.

### load\_bbrad\_calibrations

The input to this function is the numeric day. It reads the 4 broadband radiometer calibration files in /data/netCDF/corrections, named *xxx*\_calibration.txt where *xxx* is the name of each radiometer (CM21, CMP21, CG4 or CHP1). Depending on the numeric day, the correct calibration factor is read for each radiometer. This function is only called by the generate\_netcdf\_bbrad\_f5 function, i.e. the function used to read format5 broadband radiometer files prior to October 2021. Since broadband radiometer data have been recorded using a CR1000X datalogger, the calibration factors are included in the CR1000X definition file.

### generate\_netcdf\_graphs

The inputs to this function are the netCDF file to be plotted, the directory where the plots should be saved, the netCDF variables to plot and the current data as a string. It produces plots of the required variables in subdirectories under the /data/amof-netCDF/graphs directory. When new directories are created, permissions are changed to allow read, write and execute access for the user creating the files and the netCDF group, and read and execute access for other users.

The PyART module, produced by ARM, is imported at the start of the metsensors\_ncas.py code and is used to provide some functions and colormaps.

Both time-series and time-height plots can be produced using this function, using the Matplotlib plot and pcolormesh commands respectively.

The code has to read the quality flag for the variables to be plotted. The default if there is only 1 variable in the netCDF files is that this quality flag variable will be called “qc\_flag” but if there is more than 1 variable the quality flag associated with each variable has a different name to identify it. These qc\_flag names are included in the code. If a new netCDF file was to be written with a quality flag name other than qc\_flag, the function would need to be modified to include this name.

## Production of webpages to display quicklooks

Scripts using Webmagick are used to produce a directory structure of quicklooks of all plots. These are described in a separate document.

1. Both the CG4 and CHP1 incorporate a temperature sensor. For the CG4 the temperature data must be available in order to produce a valid irradiance value, so there is only 1 correction file for that instrument. Failure of either the radiation measurement or the temperature sensor invalidates the measurement. For the CHP1, the temperature measurement is not required to produce the irradiance value, so a separation correction file is needed for the radiometer and the temperature sensor. [↑](#footnote-ref-1)
2. The use of 3 separate correction files for the PWS100 reflects the different parts of the instrument which can be affected by a fault. Not all data are invalid when there is a fault in 1 part of the instrument. [↑](#footnote-ref-2)