Scientific Data Types I: NetCDF, HDF

Programming for Environmental Sciences

What is NetCDF?

- Network Common Data Form
- A set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data.
- Developed and maintained at Unidata.
- Unidata is part of the University Corporation for Atmospheric Research (UCAR)
- Unidata is funded primarily by the National Science Foundation (NSF)

The Purpose of Netcdf

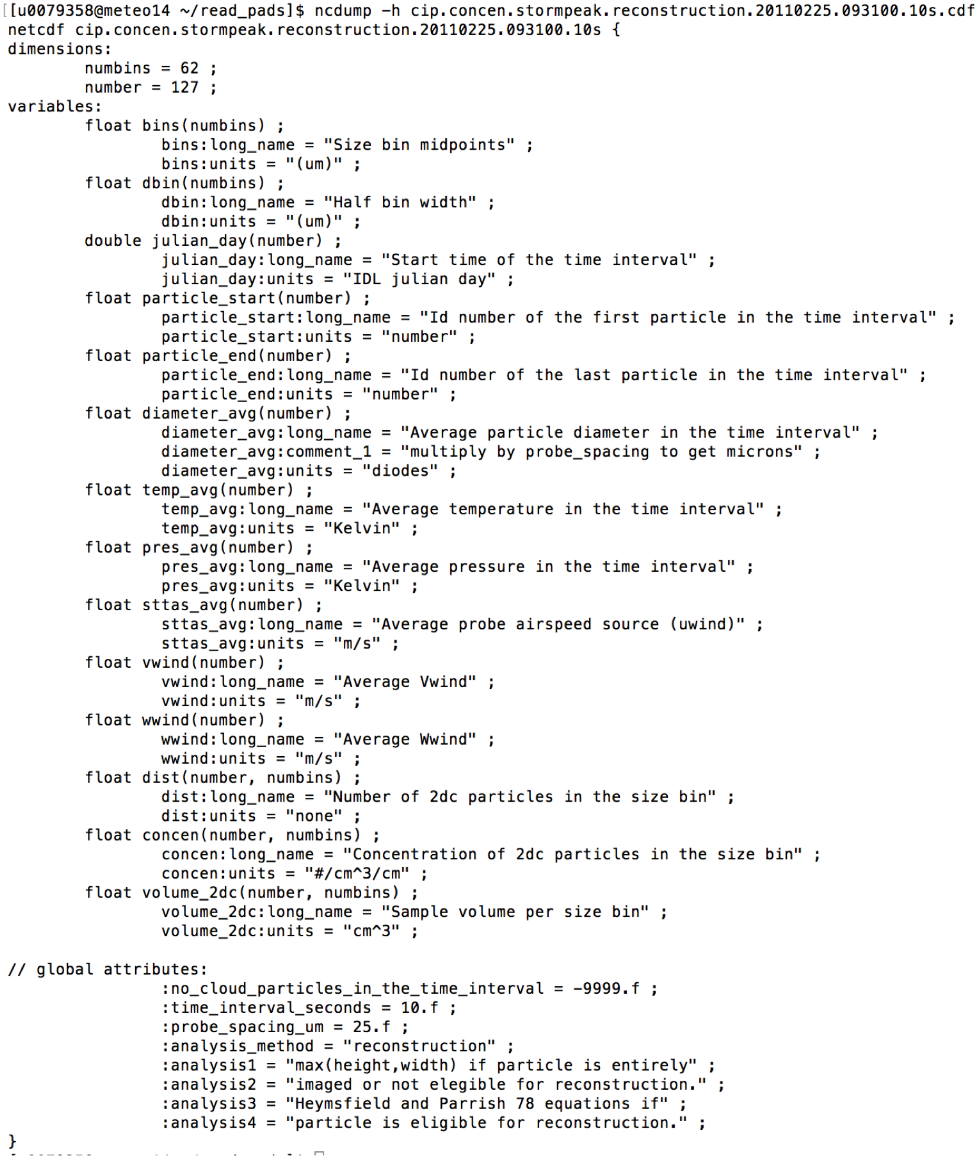
- Easily create, access and share data
- **Self-describing** A dataset includes information defining the data it contains.
- **Portable** The data in the dataset is represented in a form that can be accessed by computers with different ways of storing integers, characters, and floating-point numbers.
- C, Fortran77, Fortran90, and C++ interfaces for accessing netCDF data.
- Interfaces for other programming languages.
- Source code is freely available.
- One writer, multiple readers

The netCDF Interface

- Library of functions for storing and retrieving data in the form of arrays.
- Array values may be accessed directly, without knowing details of how the data are stored.
- Auxiliary information about the data, such as what units are used, may be stored with the data.
- Must use the only the set of functions provided by the interface.
- Files less than 1 gigabyte.

Parts of a NetCDF File

- Dataset is stored as a single file comprising two parts, header and data.



```
[u0079358@meteo14 ~/read_pads]$ ncdump -h cip.concen.stormpeak.reconstruction.20110225.093100.10s.cdf
netcdf cip.concen.stormpeak.reconstruction.20110225.093100.10s {
dimensions:
        numbins = 62 ;
        number = 127 ;
variables:
        float bins(numbins) ;
                bins:long_name = "Size bin midpoints" ;
                bins:units = "(um)" ;
        float dbin(numbins) ;
                dbin:long_name = "Half bin width" ;
                dbin:units = "(um)" ;
        double julian_day(number) ;
                julian_day:long_name = "Start time of the time interval" ;
                julian_day:units = "IDL julian day" ;
        float particle_start(number) ;
                particle_start:long_name = "Id number of the first particle in the time interval" ;
                particle_start:units = "number" ;
        float particle_end(number) ;
                particle_end:long_name = "Id number of the last particle in the time interval" ;
                particle_end:units = "number" ;
        float diameter_avg(number) ;
                diameter_avg:long_name = "Average particle diameter in the time interval" ;
                diameter_avg:comment_1 = "multiply by probe_spacing to get microns" ;
                diameter_avg:units = "diodes" ;
        float temp_avg(number) ;
                temp_avg:long_name = "Average temperature in the time interval" ;
                temp_avg:units = "Kelvin" ;
        float pres_avg(number) ;
                pres_avg:long_name = "Average pressure in the time interval" ;
                pres_avg:units = "Kelvin" ;
        float sttas_avg(number) ;
                sttas_avg:long_name = "Average probe airspeed source (uwind)" ;
                sttas_avg:units = "m/s" ;
        float vwind(number) ;
                vwind:long_name = "Average Vwind" ;
                vwind:units = "m/s" ;
        float wwind(number) ;
                wwind:long_name = "Average Wwind" ;
                wwind:units = "m/s" ;
        float dist(number, numbins) ;
                dist:long_name = "Number of 2dc particles in the size bin" ;
                dist:units = "none" ;
        float concen(number, numbins) ;
                concen:long_name = "Concentration of 2dc particles in the size bin" ;
                concen:units = "#/cm^3/cm" ;
        float volume_2dc(number, numbins) ;
                volume_2dc:long_name = "Sample volume per size bin" ;
                volume_2dc:units = "cm^3" ;

// global attributes:
                :no_cloud_particles_in_the_time_interval = -9999.f ;
                :time_interval_seconds = 10.f ;
                :probe_spacing_um = 25.f ;
                :analysis_method = "reconstruction" ;
                :analysis1 = "max(height,width) if particle is entirely" ;
                :analysis2 = "imaged or not elegible for reconstruction." ;
                :analysis3 = "Heymsfield and Parrish 78 equations if" ;
                :analysis4 = "particle is eligible for reconstruction." ;
}
```

Create a NetCDF file

- Typical sequence of netCDF calls used to create a new netCDF dataset.
  - NF90_CREATE ! Create netcdf dataset: enter define mode
  - NF90_DEF_DIM ! Define dimensions
  - NF90_DEF_VAR ! Define variables
  - NF90_PUT_ATT ! Assign values
  - NF90_ENDDEF ! End definitions: leave define mode
  - NF90_PUT_VAR ! Provide vales for variable
  - NF90_CLOSE ! Close: save new netCDF dataset

  - Two modes, define mode or data mode
  - Go to IDL program

NetCDF

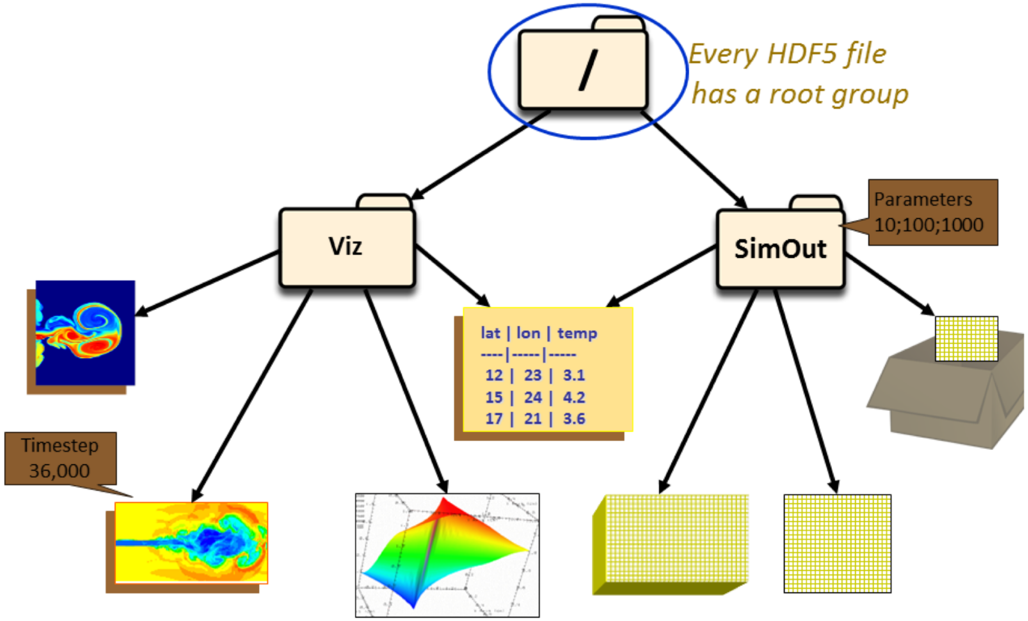
- Variable names through dumping the header
- Variable names using “Inquire” functions
- You can overwrite data values in an existing variable
- Append more data variables
- Append more data to variables along the unlimited dimension

HDF Hierarchial Data Format

- Designed to store and organize large amounts of data
- Originally developed at the National Center for Supercomputing Applications
- Supported by the HDF Group, a non-profit corporation
- Hdf is supported by many platforms
- Freely available
- Java-based HDF Viewer   HDFView
- Self-describing, Metadata is stored in the form of user-defined, named attributes attached to groups and datasets.
- Datasets can be images, tables, graphs, and even documents.

Groups

HDF groups organize data objects. Every HDF file contains a root group that can contain other groups. There are two groups, viz, simout.



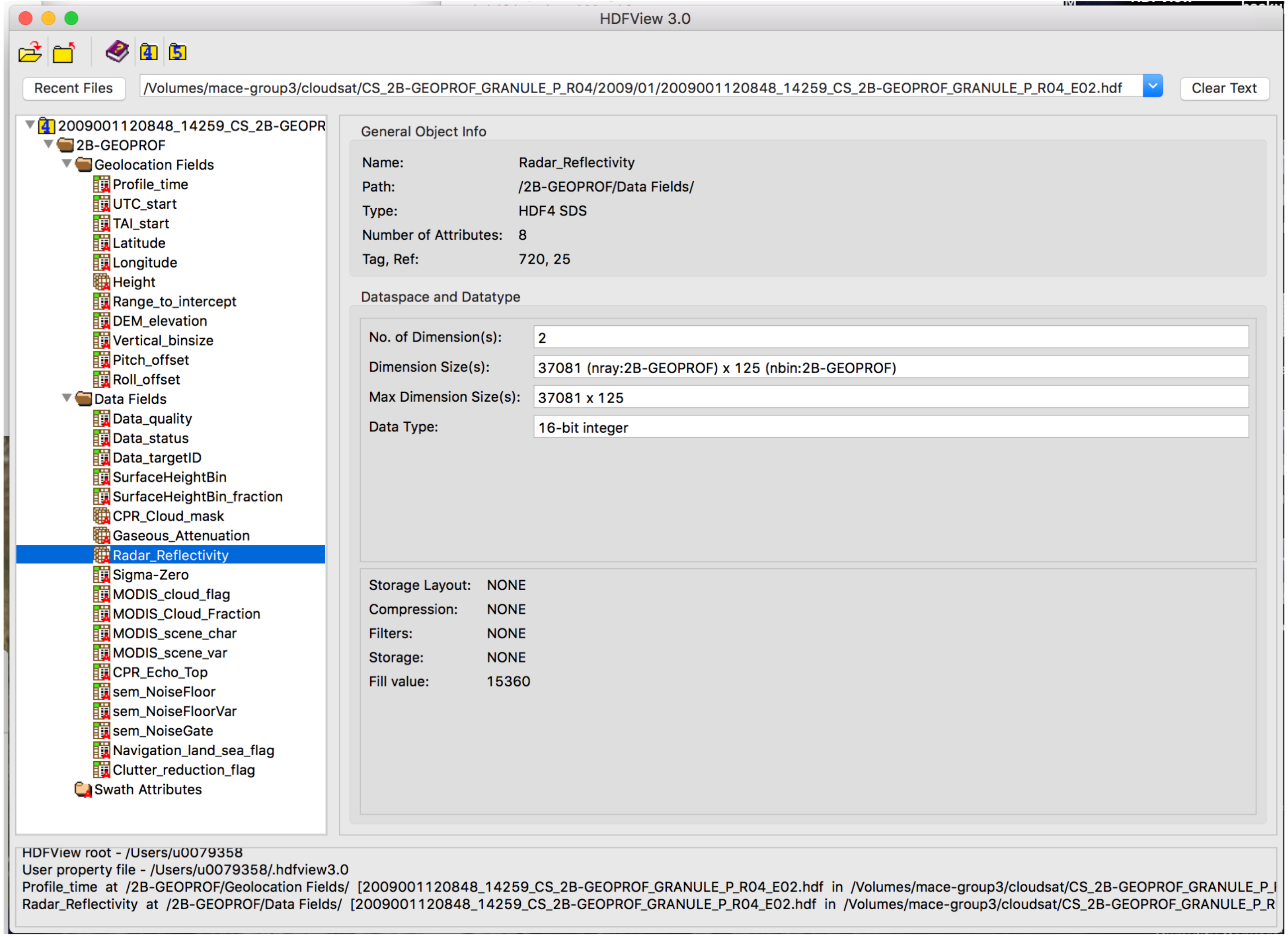
```
lat | lon | temp
----|-----|-----
 12 |  23 |  3.1
 15 |  24 |  4.2
 17 |  21 |  3.6
```

Language specific files must be included in applications
C: Add #include hdf5.h

FORTRAN: Add USE HDF5 and call h5open_f and h5close_f to initialize and close the HDF5
FORTRAN interface

Python: Add import h5py / import numpy

HDFView



/Volumes/mace-group3/cloudsat/CS_2B-GEOPROF_GRANULE_P_R04/2009/01/2009001120848_14259_CS_2B-GEOPROF_GRANULE_P_R04_E02.hdf

- 2009001120848_14259_CS_2B-GEOPR
  - 2B-GEOPROF
    - Geolocation Fields
      - Profile_time
      - UTC_start
      - TAI_start
      - Latitude
      - Longitude
      - Height
      - Range_to_intercept
      - DEM_elevation
      - Vertical_binsize
      - Pitch_offset
      - Roll_offset
    - Data Fields
      - Data_quality
      - Data_status
      - Data_targetID
      - SurfaceHeightBin
      - SurfaceHeightBin_fraction
      - CPR_Cloud_mask
      - Gaseous_Attenuation
      - Radar_Reflectivity
      - Sigma-Zero
      - MODIS_cloud_flag
      - MODIS_Cloud_Fraction
      - MODIS_scene_char
      - MODIS_scene_var
      - CPR_Echo_Top
      - sem_NoiseFloor
      - sem_NoiseFloorVar
      - sem_NoiseGate
      - Navigation_land_sea_flag
      - Clutter_reduction_flag
    - Swath Attributes

General Object Info

Name:	Radar_Reflectivity
Path:	/2B-GEOPROF/Data Fields/
Type:	HDF4 SDS
Number of Attributes:	8
Tag, Ref:	720, 25

Dataspace and Datatype

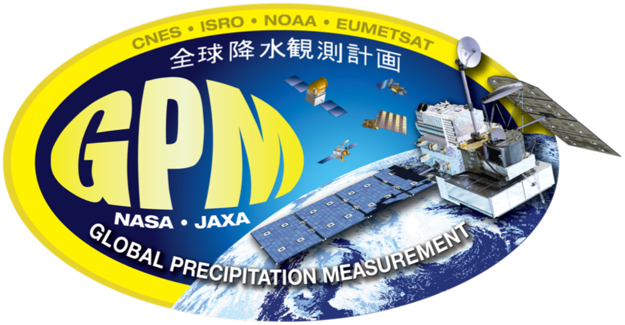
No. of Dimension(s):	2
Dimension Size(s):	37081 (nray:2B-GEOPROF) x 125 (nbin:2B-GEOPROF)
Max Dimension Size(s):	37081 x 125
Data Type:	16-bit integer

Storage Layout:	NONE
Compression:	NONE
Filters:	NONE
Storage:	NONE
Fill value:	15360

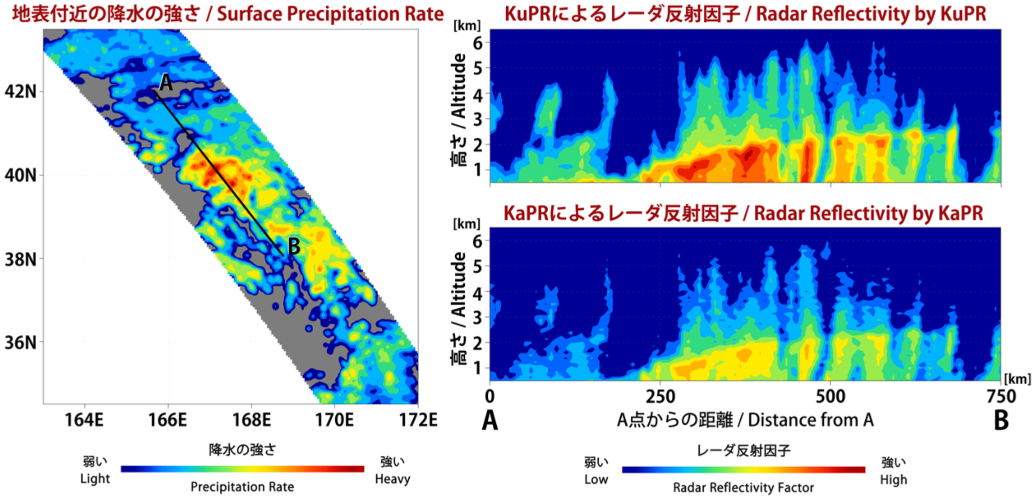
```
HDFView root - /Users/u0079358
User property file - /Users/u0079358/.hdfview3.0
Profile_time  at  /2B-GEOPROF/Geolocation Fields/  [2009001120848_14259_CS_2B-GEOPROF_GRANULE_P_R04_E02.hdf  in  /Volumes/mace-group3/cloudsat/CS_2B-GEOPROF_GRANULE_P_I
Radar_Reflectivity  at  /2B-GEOPROF/Data Fields/  [2009001120848_14259_CS_2B-GEOPROF_GRANULE_P_R04_E02.hdf  in  /Volumes/mace-group3/cloudsat/CS_2B-GEOPROF_GRANULE_P_R
```

Go to IDL program

GPM


CNES • ISRO • NOAA • EUMETSAT
全球降水観測計画
GPM
NASA • JAXA
GLOBAL PRECIPITATION MEASUREMENT

- Global Precipitation Measurement Mission
- A comprehensive description of global precipitation can only be achieved from the vantage point of space.
- NASA and JAXA deployed the GPM Core Observatory on February 28, 2014.
- Carries the first spaceborne dual-frequency phased array precipitation radar, the DPR, operating at Ku and Ka bands (13 and 35 GHz) and a conical scanning multi-channel (10-183 GHz) microwave imager (GMI)



GPM data documentation

Let’s use some files

- /uufs/chpc.utah.edu/common/home/mace-group4/gpm/GPM_L2/GPM_2AKa.05/2018/010
- /uufs/chpc.utah.edu/common/home/mace-group3/cloudsat/CS_2B-GEOPROF_GRANULE_P_R05/2007/044
- /uufs/chpc.utah.edu/common/home/mace-group3/arm/grw/grwvceil25kM1.b1/2010
- /uufs/chpc.utah.edu/common/home/mace-group4/modis/MYD06_L2/2008/092
- /uufs/chpc.utah.edu/common/home/mace-group4/Capricorn2/CBand

Go to python program