Write and read data, and perform diagnostic/graphical analyses

Data formats commonly used in atmospheric science

ASCII/Text format

- Easy to read and write.
- Can directly edit.
- File sizes are large, slow to read and write, not suitable for large datasets.

Notes: ASCII represents "American Standard Code for Information Interchange" for encoding 128 characters (e.g., 1, 2, a, b, A, B) using 7 bits (a sequences of 0/1 binary bits) or about 1 byte (https://en.wikipedia.org/wiki/ASCII). Late is was extended to use 8 bits or 1 byte encode more than 128 characters.

ASCII doesn't support characters like Greek letters. Unicode (Universal Coded Character Set) Transformation Format (UTF-8) uses variable width encoding, using one to four one-byte (8-bit) code to represent characters. UTF-8 can encode 1,112,064 characters (instead of just 128 of 7-bit ASCII).

Using ASCII code, number 102.23245356 requires 12 bytes of storage space to store this single number.

gzip compression of ASCII/text files typically reduces the files size by several times (the more repetition of characters within, the more reduction).

Binary

- File sizes much smaller, much faster (often the fastest) to read and write than ASCII/Text files.
- Different on big endian and little endian computers.
- Language (e.g., C, Fortran) dependent.
- Not self-descriptive. Without knowing exactly how the data is written, almost impossible to read.

- No internal compression.
- A single-precision Fortran floating point number is 32 bits or 4 bytes, and takes that amount of space when written to disk before any compression. gzip compression of binary data files typically reduces the size by half.

GRIB/GRIB2

- World Meteorological Organization (WMO) standard for writing and distributing gridded NWP model data.
- Portable and compact with GRIB2 that supports internal compression.
- Complex encoding, poor documentations, difficult to read without existing subroutines/libraries.
- Conversion software exists for converting GRIB data to NetCDF format.
- See https://www.cpc.ncep.noaa.gov/products/wesley/wgrib2/netcdf.html about commonly used wgrib2 converter.

BUFR

- A WMO standard format for encoding observational data.
- BUFR belongs to the category of table-driven code forms, where the meaning of data elements is determined by referring to a set of tables that are kept and maintained separately from the message itself.
- Not surprisingly, it is difficult to read also.

Data formats (borrowing from a slide)

Text, ASCII:

- METARS, Wyoming upper air sonde profiles
- TXT, DAT, CSV, YAML, HTML, XML, KML

Pros: Can open in almost anything, even a text editor or web browser but too large files are difficult or slow to open.

Cons: inefficient for read/write and storage. Different OSes treat non-printables differently

Excel, *.xls or *.csv

- IMPROVE field experiment data
- Column-ordered data, convertible to ASCII

Pros: mostly easy to import.
Cons: requires proprietary software to read. Limited complexity of data set.

HDF, netCDF – scientific data formats

- ARM, NASA EOS, MPLNet, ECMWF
- Cross-platform binary files

Pros: Cross-platform, efficient read/write and archival. Self-documentable
Cons: Requires libraries to open and access within a dev. environment.

GIF, TIFF, PNG – image formats

PDF, docx, pptx – document formats.

NetCDF (Network Common Data Form) is a set of data formats and associated software developed by Unidata (unidata.ucar.edu), an organization under UCAR (the governing body of NCAR), initially to support data needs by the atmospheric science community, especially the modeling community. It has now become a general scientific data format use by man disciplines.

It is the most widely used data format by many community atmospheric modeling systems, including WRF and NCAR climate system models.

GRIB/GRIB2 is still the (WMO) standard data format used by most operational NWP models (e.g., GFS), there are conversion tools to convert GRIB format data to NetCDF format.

According to

https://www.unidata.ucar.edu/software/netcdf/

NetCDF (Network Common Data Form) is a set of software libraries and machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. It is also a community standard for sharing scientific data. The Unidata Program Center supports and maintains netCDF programming interfaces for C, C++, Java, and Fortran. Programming interfaces are also available for Python, IDL, MATLAB, R, Ruby, and Perl.

Data in netCDF format is:

- **Self-Describing.** A netCDF file includes information about the data it contains.
- **Portable.** A netCDF file can be accessed by computers with different ways of storing integers, characters, and floating-point numbers.
- **Scalable.** Small subsets of large datasets in various formats may be accessed efficiently through netCDF interfaces, even from remote servers.
- **Appendable.** Data may be appended to a properly structured netCDF file without copying the dataset or redefining its structure.
- **Sharable.** One writer and multiple readers may simultaneously access the same netCDF file.
- **Archivable.** Access to all earlier forms of netCDF data will be supported by current and future versions of the software.
- **Efficient.** Supports internal data compression. Support parallel I/O for on large parallel supercomputers.

How to deal with data in various formats? - Using existing software

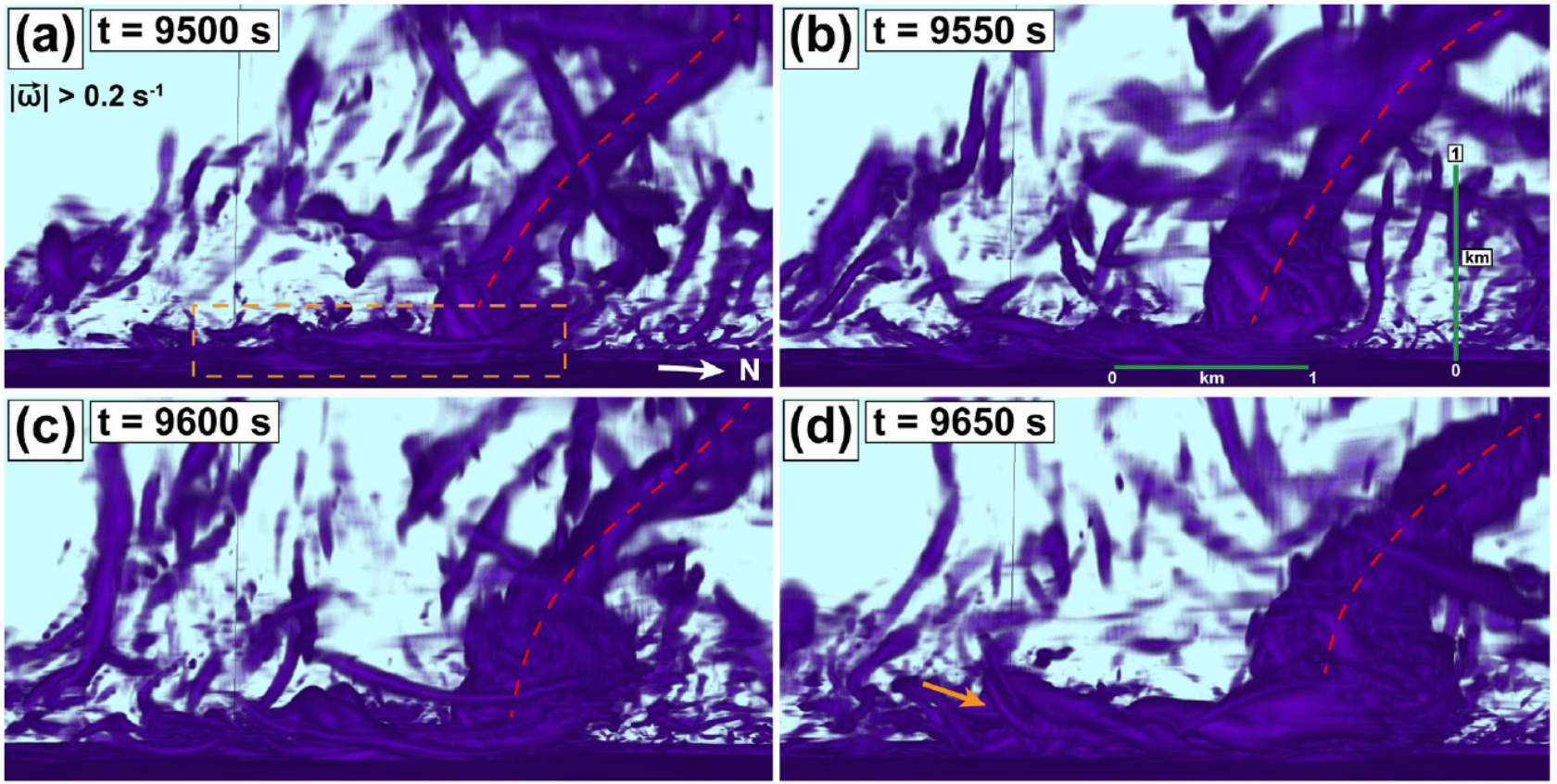
Non-commercial software

- Python, pyart (Python ARM Radar Toolkit)
- R (GNU version of "S" statistical computing/graphics package)
- Unidata packages: AWIPS, GEMPAK, IDV, McIDAS, MetPy
- Ncview – a netCDF visual brower (http://meteora.ucsd.edu/~pierce/ncview_home_page.html) that is very convenient to quickly plot/visualize netCDF data.
- Vapor (https://www.vapor.ucar.edu/) – a 3D visualization software that supports netCDF.
- Github, SourceForge, Community-supported sites

Commercial software

- **Mathematica (OU educational license)**
- **Matlab (OU educational license)**
- **SPSS (OU educational license)**
- IDL
- IGOR
- OriginPro (https://www.originlab.com/)

Vapor example for a tornado simulation:



Volume-rendered plots of three-dimensional vorticity magnitude where it exceeds 0.2 797 s^{-1} at (a) 9500 s, (b) 9550 s, (c) 9600 s, (d) 9650 s. The orange dashed rectangle highlights the early stage of the trailing horizontal vortex (HV). The red 799 dashed line subjectively denotes the tornado axis. Orange arrows indicate downward-bending 800 tails of the trailing HV and the model equivalent of the observed small spiraling vortices 801 shown in Figs. 6-8. The anticyclonic character of the spiraling vortices is highlighted in the 802 insets in (f) and (h), where $\zeta\_ < -0.1$ s^{-1} is shown in blue. The camera points to the southwest.

Writing codes of your own to read and write data

In this class, we will focus on reading and writing NetCDF files using Python. Documentations for reading and writing NetCDF files using other languages can be found at https://www.unidata.ucar.edu/software/netcdf/.

There are different packages/modules available for reading/writing NetCDF format data. We will mainly look at the most commonly used netCDF4 and the xarray/rioarray packages.

http://unidata.github.io/netcdf4-python/ is the Unidata netcdf4-python website that contains tutorials.

Documentations for NetCDF library interfaces or APIs for other languages can be found for Fortran-90, Fotran-77, C++ and C at the following sites:

https://www.unidata.ucar.edu/software/netcdf/docs-fortran/f90_The-NetCDF-Fortran-90-Interface-Guide.html

https://www.unidata.ucar.edu/software/netcdf/docs-fortran/nc_f77_interface_guide.html

https://docs.unidata.ucar.edu/netcdf-cxx/current/functions_func.html

https://docs.unidata.ucar.edu/netcdf-c/current/

OU Library's Jupyter Lab that we will use in this class.

Go to http://jupyter.lib.ou.edu/, log in using your OU's 4+4 ID. The first time you log in, you will go through CILogon and may be asked to choose University of Oklahoma as the organization.

Dr. Mark Laufersweiler from OU library will provide guest lectures and tutorials on using the platform.

Let's try to download some simple example Python programs from

https://www.unidata.ucar.edu/software/netcdf/examples/programs/

and try to run within this environment.

Modes of operations

OU Lib's JupyterLab provides three modes of operations: Notebook, Console and Terminal.

Terminal is a dumbed down version of Linux bash shell, it provides basic functions of a Linux system.

Use Terminal to manage files (cp, mv, rm, ls files etc), edit code if you like (using e.g., vi), and install additional python packages.

Basic Linux cammands: https://www.hostinger.com/tutorials/linux-commands, https://kinsta.com/blog/linux-commands/.

Try the following Unix/Linux shell commands:

```
pwd
ls  -l
touch testfile.txt
mv testfile.txt newfilename.txt; ls -l
mkdir testdirectory; ls -l
mv newfilename.txt testdirectory; ls -l
cd testdirectory; ls -l
cp newfilename.txt ../oldfilename.txt
cd ..; ls -l
```

Running python programs

You can also run python programs via the terminal interface, via

```
python yourpython_program.py
```

Within the Notebook window, you can also run your terminal commands by putting ! in front of the commands, e.g., `!ls -l. or !python yourpython_program.py.`

You can also directly run python script using `%run yourpython_program` (without .py)

Installing python packages

The meteorology server option already has the most used python packages installed.

`conda list` lists currently installed packages.

`conda install -c conda-forge rioxarray` installs rioxarray package (https://corteva.github.io/rioxarray/stable/) used by one our example programs.

```
conda install -c conda-forge nc-time-axis
```

After installing additional packages, restart kernel by clicking Kernel when you run a Jupyter notebook.

https://www.youtube.com/watch?v=VH-PCQ991fw - Quick intro on reading NetCDF4 files.

https://www.youtube.com/watch?v=-kHxOOGtPhI - NetCDF with Python (netCDF4) - Metadata, Dimensions, and Variables

https://www.youtube.com/watch?v=mnoGmS2XtDg - Create a netCDF Dataset with Python (netCDF4). Including using QGIS to visualize NetCDF files.

Try Puython examples from https://www.unidata.ucar.edu/software/netcdf/examples/programs/

https://www.youtube.com/watch?v=ue8ft9ehy_Y – reading ndfCDF files using xarray and rioxarray

https://opensourceoptions.com/blog/a-better-way-to-read-netcdf-with-python-rioxarray/

https://www.youtube.com/watch?v=eoIS68sSvGI - Read and plot netCDF file in python | easy method to handle netcdf files. WRF out files. (using cartopy, xarray, metpy

https://www.youtube.com/watch?v=g0d86VLkoy0
Read and plot netCDF file in python | easy method to handle netcdf files
Downloading data from thredds.daac.ornl.gov