The NetCDF Format and WRF Output Files

Lecture 4

Ph.D., Senior lecturer Madina Tursumbayeva

Department of Meteorology and Hydrology

Lecture Objective and Learning Outcomes

Lecture objective

To introduce the NetCDF format and the structure of WRF output files, and to show how to read them with command-line tools and Python.

Learning outcomes

By the end of this lecture, students will be able to:

1. describe the structure of a NetCDF file: dimensions, variables and attributes;
2. read a file header with ncdump -h and interpret the dimensions, including the staggered ones;
3. identify the key 2-D and 3-D variables in wrfout and calculate full fields from base state and perturbation;
4. use the global attributes to find the grid and physics settings of a simulation;
5. read and process wrfout files with ncview, NCO, xarray and wrf-python.

NetCDF Files in the WRF Workflow



Almost every file in the WRF chain is a NetCDF file

- Learn to read one of them, and you can read them all
- The only exception: the intermediate files of ungrib (FILE:…)
- In this lecture: the wrfout files, which contain the model results

What Is NetCDF?

Network Common Data Form

- Developed by Unidata (UCAR, USA)
- A standard format in the atmospheric and ocean sciences
- Used for input and output by WRF, CHIMERE, ERA5 and many other models and datasets

Self-describing

Data and their description are stored together

Platform-independent

The same file can be read on any computer

Array-oriented

Designed for multi-dimensional gridded data

Widely supported

Utilities, viewers, Python, R, Fortran libraries

Anatomy of a NetCDF File

`wrfout_d01_2023-01-15_00:00:00`

Dimensions

names and sizes of the axes

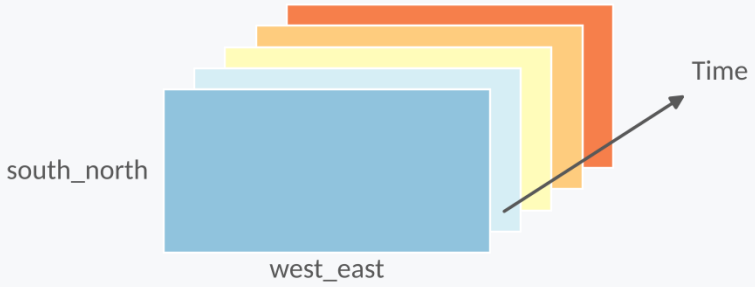
```
Time = 24

bottom_top = 49

south_north = 145

west_east = 226
```

Dimensions define the shape of every variable

Variables

arrays of data on these axes

```
T2(Time, south_north, west_east)

U(Time, bottom_top, …)

XLAT, XLONG, Times, …
```



Attributes

metadata: what the numbers mean

```
T2:units = "K"

T2:description = "TEMP at 2 M"

global: DX, MAP_PROJ, …
```

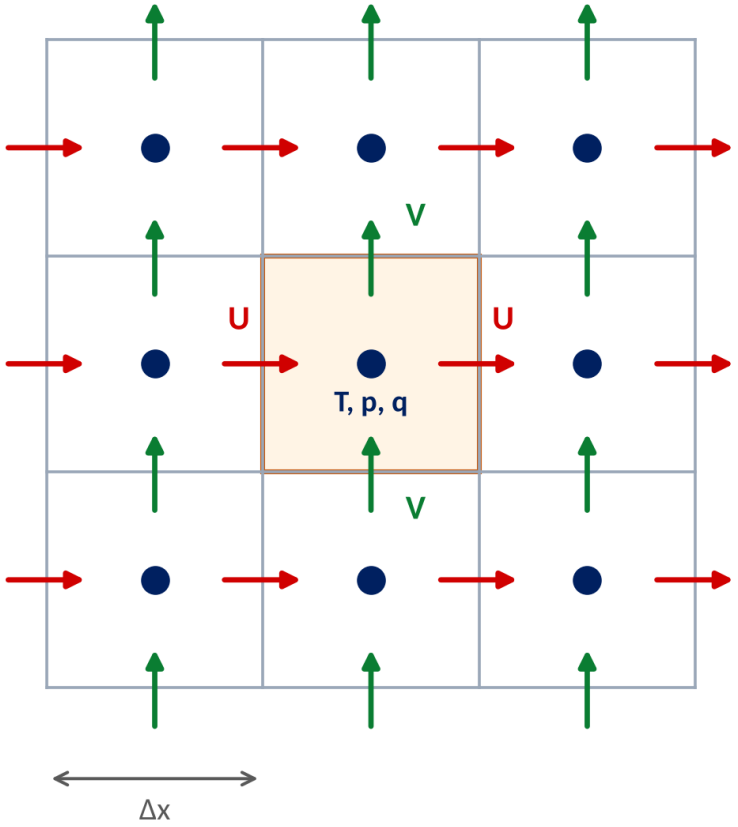
A NetCDF file is self-describing: the data and their description travel together

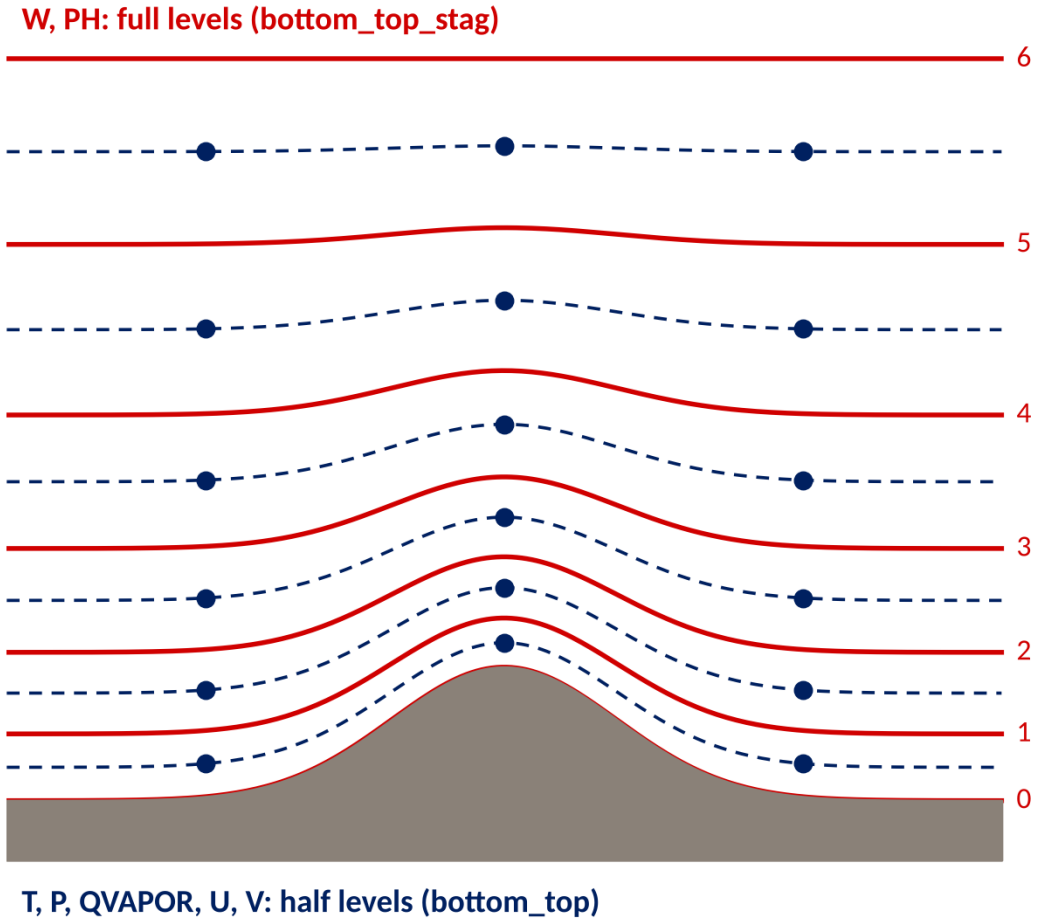
Reading the Header: ncdump -h

```
$ ncdump -h wrfout_d01_2023-01-15_00:00:00
dimensions:
   Time = UNLIMITED ; // (24 currently)
   DateStrLen = 19 ;
   west_east = 226 ;
   south_north = 145 ;
   bottom_top = 49 ;
   bottom_top_stag = 50 ;
   west_east_stag = 227 ;
   south_north_stag = 146 ;
variables:
   char Times(Time, DateStrLen) ;
   float XLAT(Time, south_north, west_east) ;
   float XLONG(Time, south_north, west_east) ;
   float T2(Time, south_north, west_east) ;
   …
```

- -h prints only the header, without the data
- First: the list of dimensions
- Then: every variable with its type and dimensions
- Always read the header before writing any code

Dimensions and Staggering





Each staggered dimension is larger by one: west_east_stag, south_north_stag, bottom_top_stag

Anatomy of a Variable

```
float T2(Time, south_north, west_east) ;
   T2:FieldType = 104 ;
   T2:MemoryOrder = "XY " ;
   T2:description = "TEMP at 2 M" ;
   T2:units = "K" ;
   T2:stagger = "" ;
   T2:coordinates = "XLONG XLAT XTIME" ;

float U(Time, bottom_top, south_north, west_east_stag) ;
   U:description = "x-wind component" ;
   U:units = "m s-1" ;
   U:stagger = "X" ;
```

- Type, name and dimensions; Time always comes first
- units: kelvin, not °C
- stagger: empty = mass points; X, Y or Z = staggered
- coordinates: where to find longitude and latitude

Key Variables in wrfout

Surface and 2-D fields

- T2, Q2: temperature and humidity at 2 m
- U10, V10: wind at 10 m
- PSFC: surface pressure
- RAINC, RAINNC: accumulated precipitation
- PBLH: boundary-layer height
- HGT: terrain height
- XLAT, XLONG, Times: coordinates and time

3-D fields

- U, V, W: wind components
- T: perturbation potential temperature
- P, PB: pressure
- PH, PHB: geopotential
- QVAPOR: water vapour mixing ratio

Base State and Perturbation

Pressure

`p = P + PB`

units: Pa

Height of model levels

`z = (PH + PHB) / 9.81`

units: m

Potential temperature

`θ = T + 300`

units: K

- Several WRF variables are stored in two parts: a base state and a perturbation
- The variable T is not the air temperature
- wrf-python can do these calculations for us

Global Attributes: the Passport of a Simulation

```
// global attributes:
    :TITLE = " OUTPUT FROM WRF V4.3 MODEL" ;
    :DX = 12000.f ;
    :DY = 12000.f ;
    :MAP_PROJ = 1 ;
    :CEN_LAT = 46.347f ;
    :CEN_LON = 66.929f ;
    :MP_PHYSICS = 8 ;
    :BL_PBL_PHYSICS = 2 ;
    :RA_LW_PHYSICS = 4 ;
    :RA_SW_PHYSICS = 4 ;
    :SF_SURFACE_PHYSICS = 2 ;
```

- Grid spacing, projection, centre of the domain
- All physics options of the run:
  - 8 = Thompson microphysics
  - 2 = Mellor–Yamada–Janjic PBL
  - 4 = RRTMG radiation
  - 2 = Noah land surface model
- The schemes used in our Almaty study

Physics options as in Tursumbayeva et al. (2026), Meteorological Applications, 33, e70183

Quick-Look Tools

```
# which times are in the file?
ncdump -v Times wrfout_d01_2023-01-15_00:00:00

# look at any field as a map
ncview wrfout_d01_2023-01-15_00:00:00

# extract a few variables into a small file
ncks -v T2,U10,V10,PBLH wrfout_d01_* t2_wind.nc

# average over time / difference of two files
ncra  in.nc mean.nc
ncdiff run2.nc run1.nc diff.nc
```

- ncdump: print the header or the values of a variable
- ncview: a simple viewer for a first check
- NCO (netCDF Operators): ncks, ncra, ncdiff and others
- Recommended in the NCAR WRF tutorial

Reading a File with xarray

```
import xarray as xr

ds = xr.open_dataset("wrfout_d01_2023-01-15_00:00:00")
print(ds)        # dimensions, variables, attributes

t2 = ds["T2"].isel(Time=0) - 273.15   # K -> °C
print(t2.shape)  # (145, 226)

t2.plot()        # a quick map
```

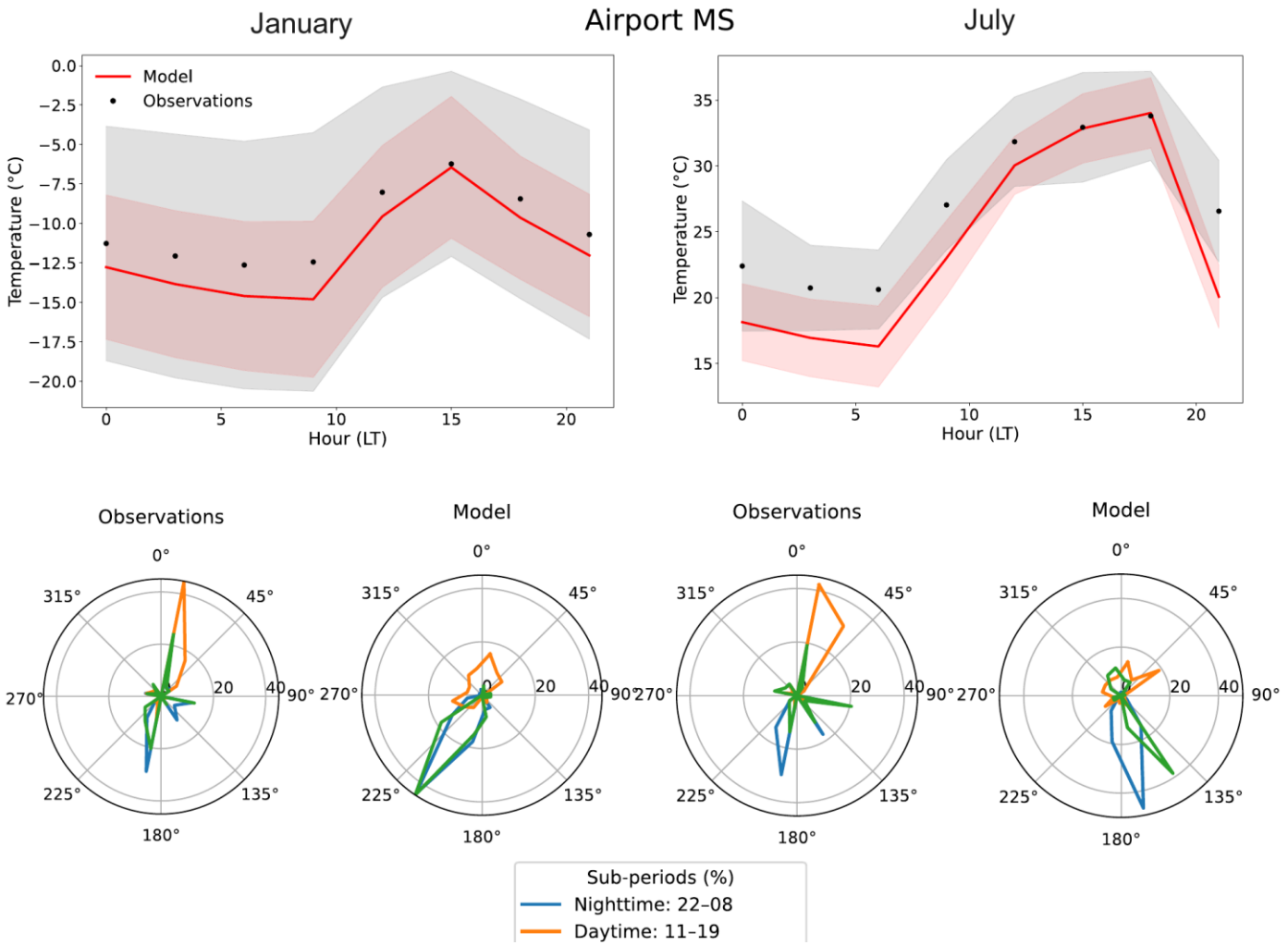
- open_dataset: open the file
- ds["T2"]: select a variable by name
- isel(Time=0): select the first time
- Data are read from disk only when needed

wrf-python: getvar and Diagnostics

```
from netCDF4 import Dataset
from wrf import getvar, ll_to_xy, ALL_TIMES

nc = Dataset("wrfout_d01_2023-01-15_00:00:00")

t2  = getvar(nc, "T2", timeidx=ALL_TIMES) - 273.15
slp = getvar(nc, "slp")     # sea-level pressure

# grid point nearest to Almaty MS
x, y = ll_to_xy(nc, 43.24, 76.93)
t2_almaty = t2[:, y, x]
```

- getvar: variables from the file and diagnostics
  - tk, rh, slp, z, pressure, theta
  - ua, va: destaggered wind
  - uvmet10: wind rotated to north and east
- ALL_TIMES: read every time in the file
- ll_to_xy: nearest grid point to a location

From File to Result: Model vs Observations



Airport MS, Almaty: model (red) and observations (black). Source: Tursumbayeva et al. (2026), Meteorological Applications, 33, e70183

Four Typical Mistakes

Staggered variables

U, V and W have different shapes: destagger them first

Perturbation variables

T is not temperature: $\theta = T + 300$, $p = P + PB$, $z = (PH + PHB) / 9.81$

Accumulated precipitation

RAINC + RAINNC is the total since the start: take differences in time

Time and wind direction

Times are in UTC; U10 and V10 are relative to the model grid

Summary

Key points

- NetCDF is self-describing: dimensions, variables, attributes
- Almost all WRF files use this format
- ncdump -h shows what is inside a file
- Remember staggered dimensions and base state + perturbation
- xarray and wrf-python: read and analyse in a few lines of Python

Next lecture

Mapping meteorological fields

Maps of temperature, wind and pressure from WRF output

Thank you for your attention!

Self-Check Questions

1. What does “self-describing” mean for a NetCDF file?
2. What does ncdump -h show, and why should the header be read before writing any code?
3. Why are the staggered dimensions larger by one than the unstaggered ones?
4. How are pressure, height and potential temperature obtained from P, PB, PH, PHB and T?
5. How is the precipitation of a given period obtained from RAINC and RAINNC?
6. What do getvar and ll_to_xy do in wrf-python?

References

1. Skamarock, W. C., et al. (2019). A Description of the Advanced Research WRF Model Version 4. NCAR Technical Note NCAR/TN-556+STR. https://doi.org/10.5065/1dfh-6p97
2. Tursumbayeva, M., Ciarelli, G., Di Antonio, L., Bettineschi, M., Kissebayev, D., & Baimatova, N. (2026). Modelling the Black Carbon Variability and Dynamics in Almaty, Kazakhstan, During Winter and Summer Seasons. Meteorological Applications, 33, e70183. https://doi.org/10.1002/met.70183
3. NSF NCAR Mesoscale and Microscale Meteorology Laboratory. WRF Tutorial: recorded lectures (YouTube, wrfhelp NCAR-MMM, 2021) and lecture slides, including “Post-Processing Tools: Python”. https://www2.mmm.ucar.edu/wrf/wrf_tutorial/lectures.html
4. Unidata. NetCDF: Network Common Data Form. https://www.unidata.ucar.edu/software/netcdf/
5. Ladwig, W. (2017). wrf-python (Version 1.3) [Software]. UCAR/NCAR. https://doi.org/10.5065/D6W094P1
6. Hoyer, S., & Hamman, J. (2017). xarray: N-D labeled Arrays and Datasets in Python. Journal of Open Research Software, 5(1), 10. https://doi.org/10.5334/jors.148
7. Zender, C. S. (2008). Analysis of self-describing gridded geoscience data with netCDF Operators (NCO). Environmental Modelling & Software, 23(10), 1338–1342.