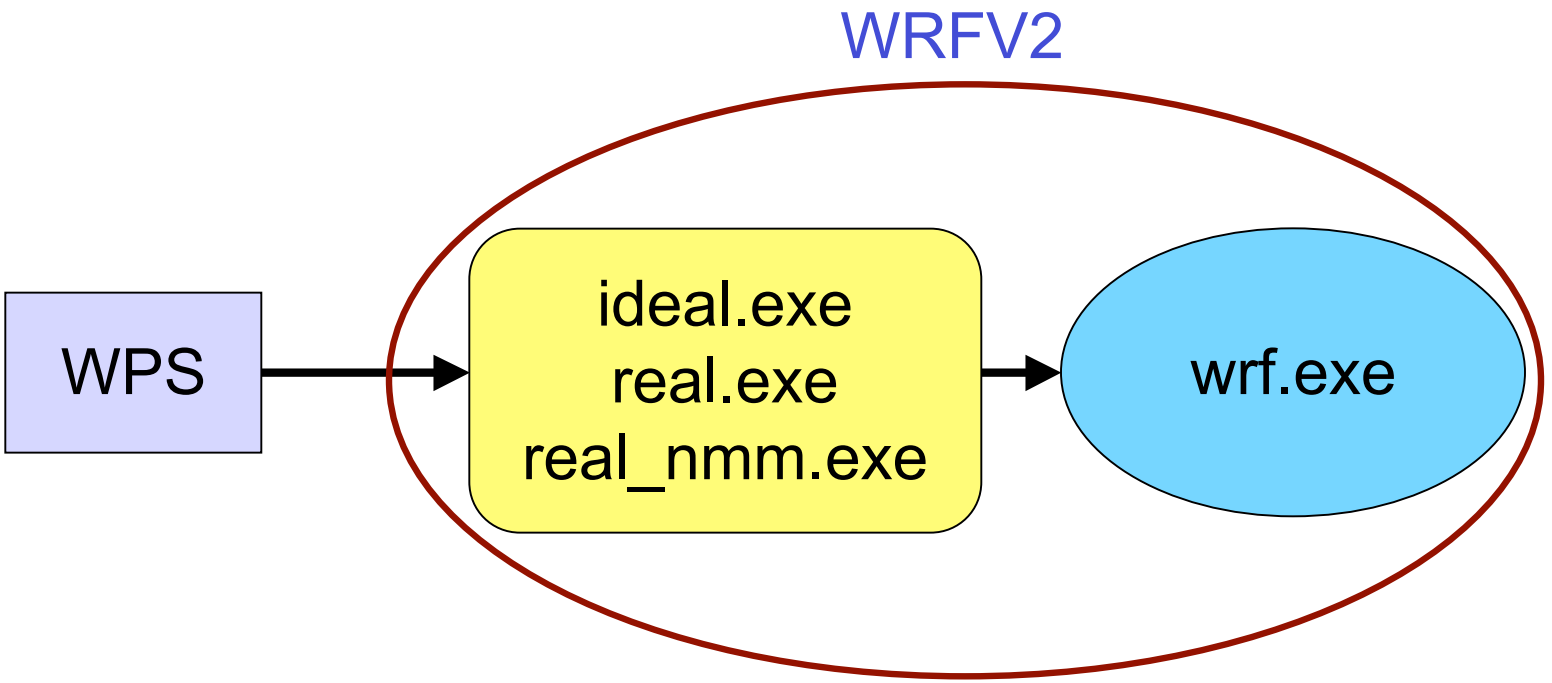
WRF: Set Up and Run

Wei Wang

NCAR/ESSL/MMM

WRF System Flowchart



Outline

- Running WRF code
  - Before you run..
  - Running idealized case
  - Running ARW real-data case
  - Running NMM real-data case
- Basic runtime options (*namelist*)
- Check output

This talk is complementary to 'Nesting' and 'Nudging' talks later

Assuming …

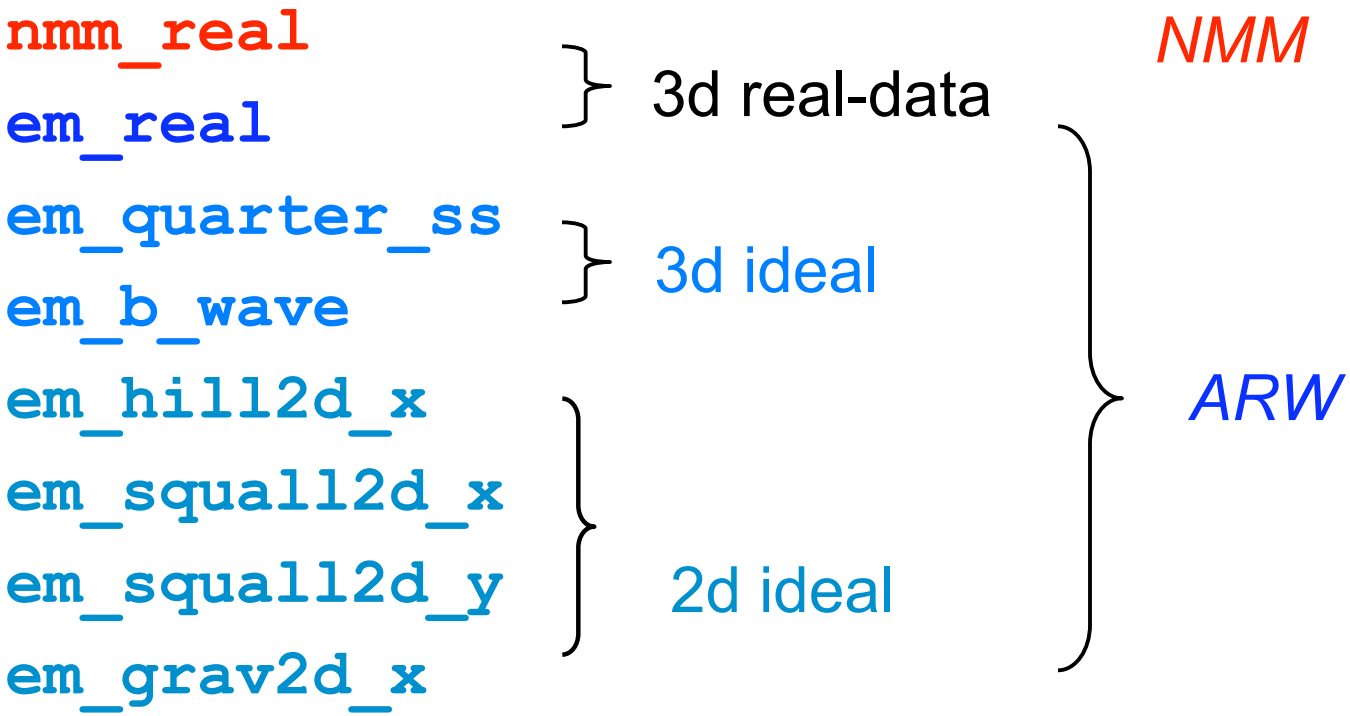
- WRFV2 is successfully compiled, and
- WPS has been run successfully to produce appropriate files for the particular dynamical core you want to run
  - *If you are interested in running an idealized case, use WRF-ARW and skip WPS.*

Check and Make Sure ..

- Executables are created in **`WRFV2/main/`** directory:
  - **`ideal.exe`**, or
  - **`real.exe`** or **`real_nmm.exe`**
  - **`wrf.exe`**
  - **`ndown.exe`** (for ARW one-way nesting)
- If you are running a real-data case, files from WPS are correctly generated:
  - **`met_em.d01.*`**, or
  - **`met_nmm.d01.*`**
- Prepare **`namelist.input`** for runtime options.

Depending on the Interest ..

You have these choices in **`WRFV2/test/`**



Steps to Run

1. cd to *run/* or one of the *test case* directories
2. Link or copy WPS output files to the directory for real-data cases
3. Edit *namelist.input* file for the appropriate grid and times of the case
4. Run initialization program (*ideal.exe, real.exe*, or *real_nmm.exe*)
5. Run model executable, *wrf.exe*

WRFV2/run directory

README.namelist
LANDUSE.TBL
ETAMPNEW_DATA
GENPARM.TBL
RRTM_DATA
SOILPARM.TBL
VEGPARM.TBL
urban_param.tbl
tr49t67
tr49t85
tr67t85
gribmap.txt
grib2map.tbl
…. (a few more)

these files are model physics data files: they are used to either initialize physics variables, or make physics computation more efficient

WRFV2/run directory after compile

```
LANDUSE.TBL
ETAMPNEW_DATA
GENPARM.TBL
RRTM_DATA
SOILPARM.TBL
VEGPARM.TBL
urban_param.tbl
tr49t67
tr49t85
tr67t85
gribmap.txt
grib2map.tbl
namelist.input -> ../test/em_real/namelist.input
real.exe -> ../main/real.exe
wrf.exe -> ../main/wrf.exe
ndown.exe -> ../main/ndown.exe
```

An example after ARW real case compile

…. (a few more)

Running an Idealized Case

ARW only

Running an *Idealized* Case

- If you have compiled an ideal case, you should have:

  `ideal.exe` - ideal case initialization

  `wrf.exe` - model executable

- These executables are linked to:

  `WRFV2/run`

  and

  `WRFV2/test/`*`test_case`*

➔ One can go to either directory to run.

Running an *Idealized* Case

Go to the desired *ideal* test case directory: e.g.

```
cd test/em_quarter_ss
```

If there is **`'run_me_first.csh'`** in the directory, run it first - this links physics data files to the currect directory:

```
./run_me_first.csh
```

Running an *Idealized* Case

Then run the ideal initialization program:

```
./ideal.exe
```

The input to this program is typically a sounding file (file named ***input_sounding***), or a pre-defined 3D input (e.g. ***input_jet*** in **em_b_wave** case).

Running the program creates WRF initial condition file, called **wrfinput_d01**

Running an *Idealized* Case

- To run the model, type

  `./wrf.exe >& wrf.out &`

  for single processor or SMP run. Or

  `mpirun -np N ./wrf.exe`

  for a MPI run (where ***N*** is the number of processors requested)
- Successful running of the model executable will create a model history file called

  `wrfout_d01_<date>`

  e.g. **`wrfout_d01_0001-01-01_00:00:00`**

Running an *Idealized* Case

- Edit **`namelist.input`** file to change options
- For your own case, you may provide a different sounding

Note:

- For 2D cases and baraclinic wave case, **`ideal.exe`** must be run serially
- For all 2D cases, **`wrf.exe`** must be run serially or with SMP

Running ARW Real-Data Case

Running ARW Real-Data Case

- If you have compiled the *em_real* case, you should have:

  real.exe - real data initialization program

  wrf.exe - model executable

  ndown.exe - program for doing one-way nesting

- These executables are linked to:

  WRFV2/run

  and

  WRFV2/test/*em_real*

➔ One can go to either directory to run.

WRFV2/test/em_*real* directory

```
LANDUSE.TBL -> ../../run/LANDUSE.TBL
ETAMPNEW_DATA -> ../../run/ETAMPNEW_DATA
GENPARM.TBL -> ../../run/GENPARM.TBL
RRTM_DATA -> ../../run/RRTM_DATA
SOILPARM.TBL -> ../../run/SOILPARM.TBL
VEGPARM.TBL -> ../../run/VEGPARM.TBL
urban_param.tbl -> ../../run/urban_param.tbl
tr49t67 -> ../../run/tr49t67
tr49t85 -> ../../run/tr49t85
tr67t85 -> ../../run/tr67t85
gribmap.txt -> ../../run/gribmap.txt
grib2map.tbl -> ../../run/grib2map.tbl
namelist.input          - require editing
real.exe -> ../../main/real.exe
wrf.exe -> ../../main/wrf.exe
ndown.exe -> ../../main/ndown.exe
```

…. (a few more)

Running WRF ARW Real-data Cases

- One must successfully run WPS, and create **met_em.*** file for more than one time period

- Link or copy WPS output files to the run directory:

```
cd test/em_real
ln -s ../../../WPS/met_em.* .
```

Running WRF ARW Real-data Cases

- Edit **`namelist.input`** file for runtime options (**`&time_control`** for integration time, and **`&domains`** for grid dimensions at minimum)

- Run the real-data initialization program:

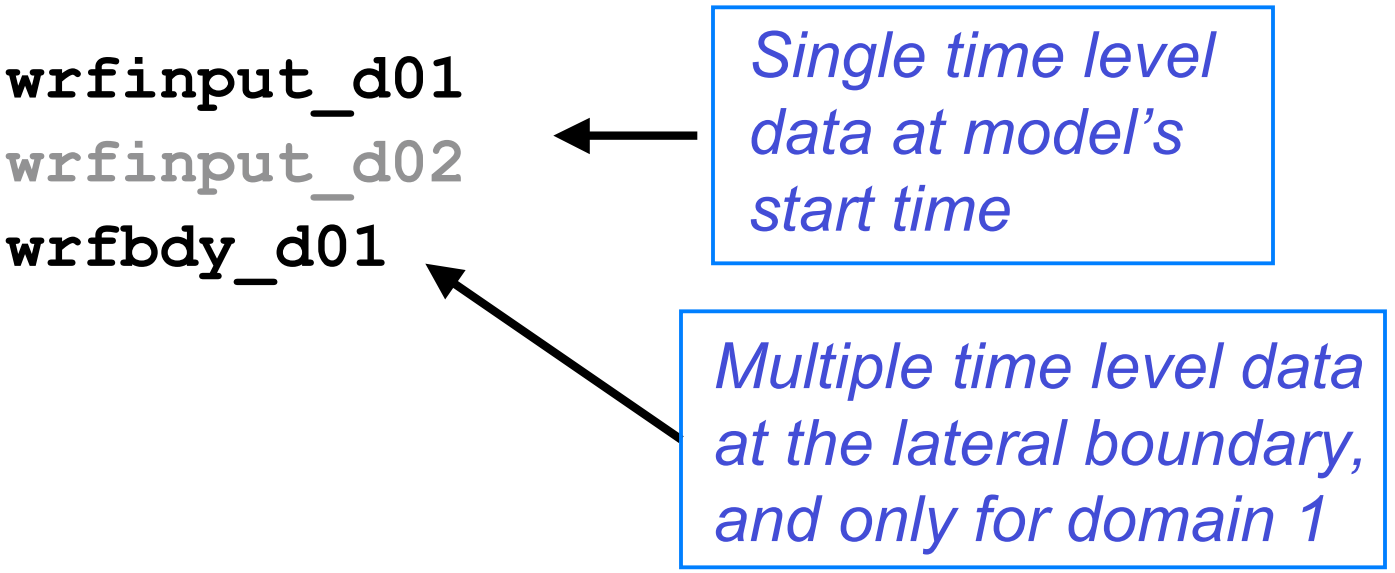
  `./real.exe`, or

  `mpirun -np N ./real.exe`

  where ***`N`*** is the number of processors requested

Running WRF ARW Real-data Cases

- Successfully running this program will create model initial and boundary files: (nested output in grey):



Running WRF ARW Real-data Cases

- Optionally, if one sets *sst_update = 1*, lower boundary file(s) containing time-varying SST and greenness fraction data will be generated: (nested output in grey):

wrflowinp_d01
wrflowinp_d02

These files have the same time frequency as LBC files. This option is reserved for long model simulations.

Running WRF ARW Real-data Cases

- Run the model executable by typing:

  `./wrf.exe >& wrf.out &`

  or

  `mpirun -np N ./wrf.exe`

- Successfully running the model will create model *history* files for each domains (nested output in grey):

  `wrfout_d01_2005-08-28_00:00:00`

  `wrfout_d02_2005-08-28_00:00:00`

  And *restart* files if selected:

  `wrfrst_d01_<date>`

  `wrfrst_d02_<date>`

Running NMM Real-Data Case

Running NMM Real-Data Case

- If you have compiled the *nmm_real*, you should have:

  `real_nmm.exe` - NMM real date initialization program

  `wrf.exe` - NMM model executable

- These executables are linked to:

  `WRFV2/run`

  and

  `WRFV2/test/nmm_real`

➔ One can go to either directory to run.

WRFV2/test/*nmm_real* directory

```
LANDUSE.TBL -> ../../run/LANDUSE.TBL
ETAMPNEW_DATA -> ../../run/ETAMPNEW_DATA
GENPARM.TBL -> ../../run/GENPARM.TBL
RRTM_DATA -> ../../run/RRTM_DATA
SOILPARM.TBL -> ../../run/SOILPARM.TBL
VEGPARM.TBL -> ../../run/VEGPARM.TBL
urban_param.tbl -> ../../run/urban_param.tbl
tr49t67 -> ../../run/tr49t67
tr49t85 -> ../../run/tr49t85
tr67t85 -> ../../run/tr67t85
gribmap.txt -> ../../run/gribmap.txt
grib2map.tbl -> ../../run/grib2map.tbl
namelist.input          - require editing
real_nmm.exe -> ../../main/real_nmm.exe
wrf.exe -> ../../main/wrf.exe
```

Running WRF NMM Real-data Cases

- One must successfully run WPS, and create **met_nmm.*** file for more than one time period

- Link or copy WPS output files to the run directory:

```
cd test/nmm_real
ln -s ../../../WPS/met_nmm.*  .
```

Running WRF NMM Real-data Cases

- Edit **namelist.input** file for runtime options (**&time_control** for integration time, and **&domains** for grid dimensions at minimum)

- Run the real-data initialization program:

  .real_nmm.exe - to run on a single processor

  mpirun -np *N* ./real_nmm.exe

- Successfully running this program will create model initial and boundary files (regardless of nesting):

  wrfinput_d01

  wrfbdy_d01

Running WRF NMM Real-data Cases

- Run the model executable by typing:

  `./wrf.exe` - to run on a single processor, or

  `mpirun -np N ./wrf.exe`

- Successfully running the model will create model *history* files for each domains (nested output in grey):

  `wrfout_d01_2005-08-28_00:00:00`

  `wrfout_d02_2005-08-28_00:00:00`

  And *restart* files if selected:

  `wrfrst_d01_<date>`

  `wrfrst_d02_<date>`

Basic namelist Options

What is a namelist?

- Contains *runtime* options for the code so that one does not need to recompile the code with namelist change

&time_control

```
 run_days                            = 0,
run_hours                            = 24,
run_minutes                          = 0,
run_seconds                          = 0,
start_year                           = 2000, 2000, 2000,
start_month                          = 01,   01,   01,
start_day                            = 24,   24,   24,
start_hour                           = 12,   12,   12,
start_minute                         = 00,   00,   00,
start_second                         = 00,   00,   00,
end_year                             = 2000, 2000, 2000,
end_month                            = 01,   01,   01,
end_day                              = 25,   25,   25,
end_hour                             = 12,   12,   12,
end_minute                           = 00,   00,   00,
end_second                           = 00,   00,   00,
interval_seconds                     = 21600
history_interval                     = 180,  60,   60,
restart_interval                     = 360,
```

coarse grid option

nest options

Note on `&time_control`

- *run_** times:
  - Model simulation length: *wrf* only and domain 1 only
- *start_** and *end_** times:
  - Program *real* will use WPS output between these times to produce lateral boundary file
  - They can be used to specify the start and end of simulation times for the coarse grid. They are used to control the start and end of simulation times for nests.

Note on `&time_control`

- *Interval_seconds*:
  - Time interval between WPS output times, and LBC update frequency
- *restart_interval*:
  - Time interval in minutes when a restart file is written
  - Restart files contain only one time level data, and its valid time is in its file names, e.g. a restart file for domain that is valid for 0000 UTC Jan 2000 is

```
wrfrst_d01_2000-01-25_00:00:00
```

&domains

```
time_step                           = 180
time_step_fract_num                 = 0,
time_step_fract_den                 = 1,
max_dom                             = 1,
s_we                                = 1,     1,      1,
e_we                                = 74,    112,    94,
s_sn                                = 1,     1,      1,
e_sn                                = 61,    97,     91,
s_vert                              = 1,     1,      1,
e_vert                              = 27,    28,     28,
num_metgrid_levels                  = 21
dx                                  = 30000, 10000,  3333,
dy                                  = 30000, 10000,  3333,
```

nests

Note on **&domains**

- *time_step, time_step_fract_num, time_step_frac_den*:
  - Time step for model integration in seconds
  - Fractional time step specified in separate integers of numerator and denominator.
- *e_we, e_sn, e_vert*:
  - Grid dimensions in X, Y and Z directions
- *num_metgrid_levels*:
  - Number of metgrid data levels.
- *dx, dy*:
  - grid distances in meters for ARW; in degrees for NMM

&time_control for a nested run

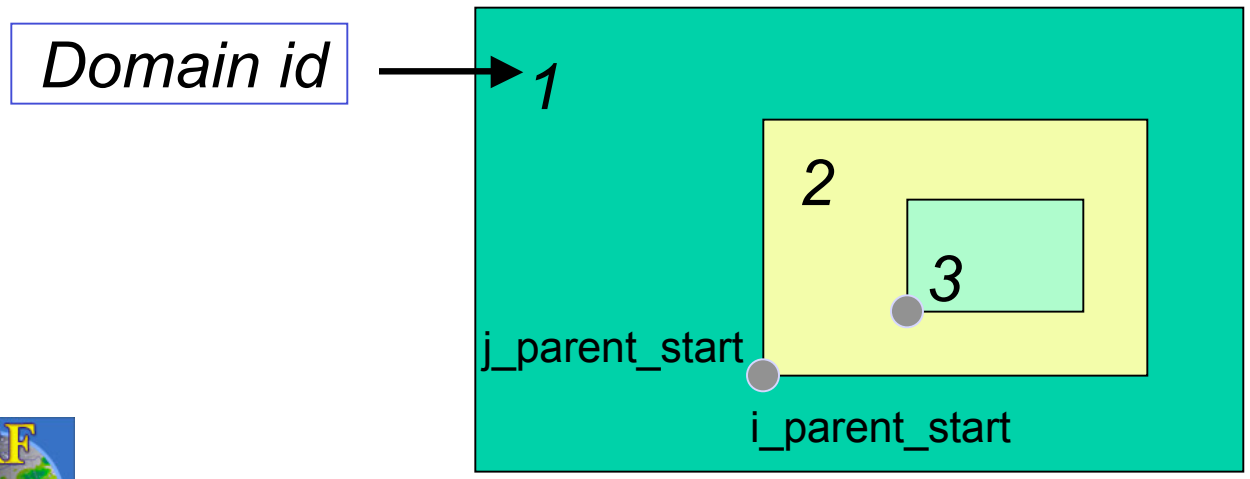
```
run_days                = 0,
run_hours               = 24,
run_minutes             = 0,
run_seconds             = 0,
start_year              = 2000, 2000, 2000,
start_month             = 01,   01,   01,
start_day               = 24,   24,   24,
start_hour              = 12,   12,   12,
start_minute            = 00,   00,   00,
start_second            = 00,   00,   00,
end_year                = 2000, 2000, 2000,
end_month               = 01,   01,   01,
end_day                 = 25,   25,   25,
end_hour                = 12,   12,   12,
end_minute              = 00,   00,   00,
end_second              = 00,   00,   00,
interval_seconds        = 21600
history_interval        = 180,  60,   60,
restart_interval        = 360,
```

&domains for a nested run

```
time_step                = 180
time_step_fract_num      = 0,
time_step_fract_den      = 1,
max_dom                  = 3,
s_we                     = 1,     1,     1,
e_we                     = 74,    112,   94,
s_sn                     = 1,     1,     1,
e_sn                     = 61,    97,    91,
s_vert                   = 1,     1,     1,
e_vert                   = 28,    28,    28,
num_metgrid_levels       = 21
dx                       = 30000, 10000, 3333.33,
dy                       = 30000, 10000, 3333.33,
```

&domains for a nested run

```
grid_id                             = 1,     2,     3,
parent_id                           = 0,     1,     2,
parent_grid_ratio                   = 1,     3,     3,
parent_time_step_ratio              = 1,     3,     3,
i_parent_start                      = 1,     31,    30,
j_parent_start                      = 2,     17,    30,
```



Note on **&domains**

- *grid_id, parent_id*:
  - Identify domains and parent of the nest
- *i_parent_start, j_parent_start*:
  - Starting (i,j) location of the nest in its parent domain
- *parent_grid_ratio, parent_time_step_ratio*:
  - An integer ratio of grid distances and time steps between coarse grid and nest grid. For NMM, it must be 3.

Where do I start?

- Always start with a *namelist* template provided in the test case directory, whether it is a ideal case, ARW or NMM.
- Use document to guide the modification of the namelist values.

Check Output

Output After a Model Run

- Standard output:

  `wrf.out`, or **`rsl`**.* files

- Model history file(s):

  `wrfout_d0*_<date>`

- Model restart file(s), optional

  `wrfrst_d0*_<date>`

Output from a multi-processor run

The standard out and error will go to the following files for a MPI run:

`show_domain_0000:` domain-decomposition info from RSL

`rsl.out.0000`	`rsl.error.0000`
`rsl.out.0001`	`rsl.error.0001`
`rsl.out.0002`	`rsl.error.0002`
`rsl.out.0003`	`rsl.error.0003`

There is one pair of files for each processor requested

What to Look for in a Standard Output File?

Check run log file by typing

`tail wrf.out`, or

`tail rsl.out.0000`

You should see the following:

`wrf: SUCCESS COMPLETE WRF`

How to Check Model History File?

- Use **ncdump**:

  ncdump -v Times wrfout_d01_<date>

  to check output times. Or

  ncdump -v U wrfout_d01_<date>

  to check a particular variable (U)

- Use **ncview** or **ncBrowse**
- Use post-processing tools (see talks later)

What is in a *wrf.out* or *rsl* file?

- A print of namelist options
- Time taken to compute one model step

```
Timing for main: time 2000-01-24_12:03:00 on domain   1:    3.25000 elapsed seconds.
Timing for main: time 2000-01-24_12:06:00 on domain   1:    1.50000 elapsed seconds.
Timing for main: time 2000-01-24_12:09:00 on domain   1:    1.50000 elapsed seconds.
Timing for main: time 2000-01-24_12:12:00 on domain   1:    1.55000 elapsed seconds.
```

- Time taken to write history and restart file

```
Timing for Writing wrfout_d01_2000-01-24_18:00:00 for domain   1:   0.14000 elapsed seconds.
```

- Any model error prints: (example from ARW run)

```
5 points exceeded cfl=2 in domain 1 at time 4.200000 MAX AT i,j,k: 123 48 3
     cfl,w,d(eta)= 4.165821
```

References

- Information on compile and run, and a more extensive list of namelist and their explanations can be found in the ARW/NMM User's Guide, Chapter 5
- Also see '*Namelist*' and '*Nesting*' lectures on Wednesday