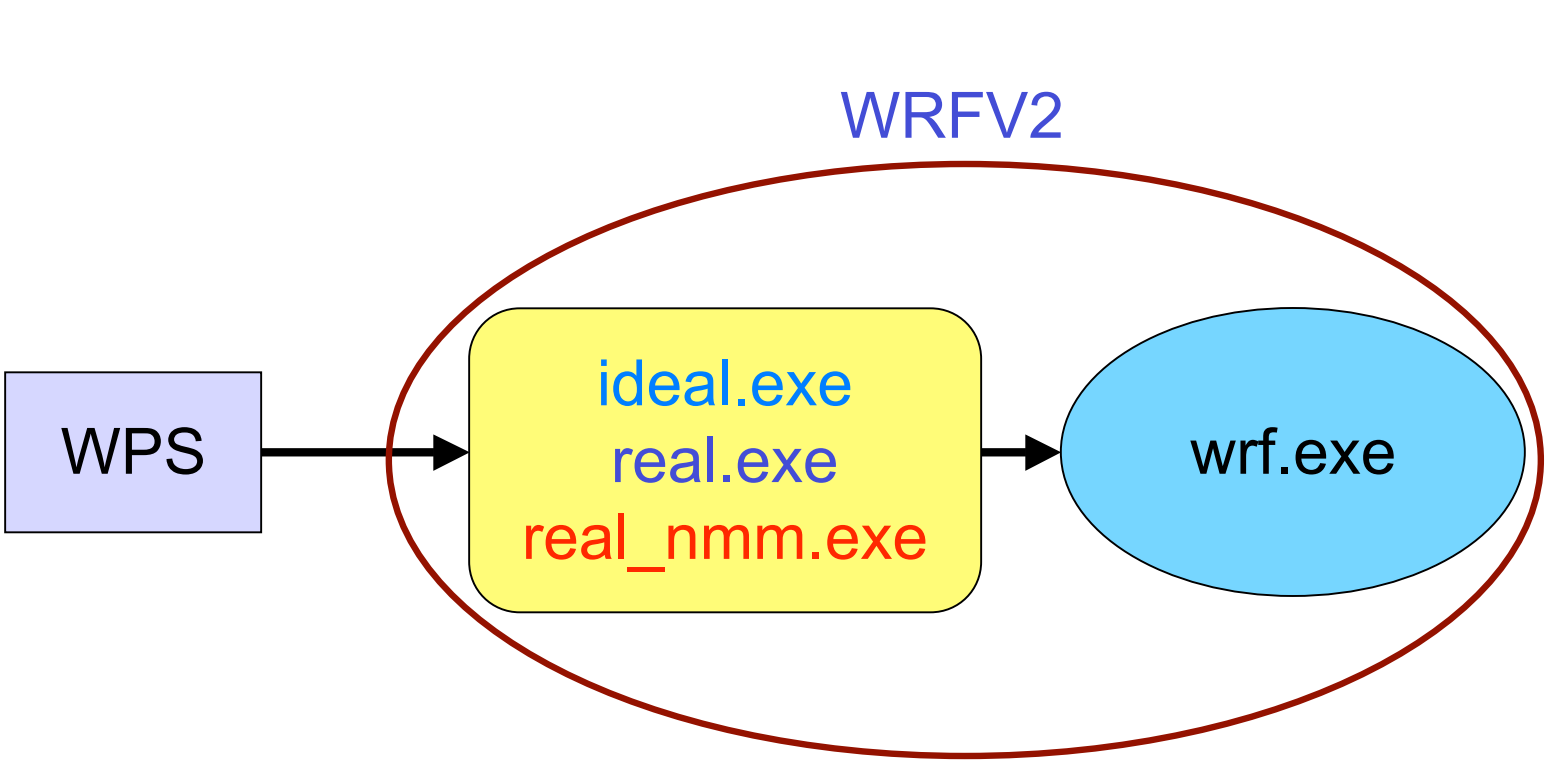
Set Up and Run WRF

(ARW-Ideal, ARW-real, and NMM)

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 for a single domain run (*namelist*)
- Check output
- Simple trouble shooting

This talk is complementary to 'Nesting' talk later.

Before You Run ..

- Check and make sure appropriate executables are created in **`WRFV2/main/`** directory:

  For ARW:
  - **`ideal.exe`**
  - **`real.exe`**
  - **`wrf.exe`**
  - **`ndown.exe`**

  For NMM:
  - **`real_nmm.exe`**
  - **`wrf.exe`**

- If you are running a real-data case, be sure that files from WPS are correctly generated:
  - **`met_em.d01.*`**, for ARW or
  - **`met_nmm.d01.*`** for NMM
- Prepare **`namelist.input`** for runtime options.

WRF test case directories

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

(made at compile time):

- **`em_real_nmm`** } 3d real-data — *NMM*
- **`em_real`** } 3d real-data — *ARW*
- **`em_quarter_ss`** } 3d ideal — *ARW*
- **`em_b_wave`** } 3d ideal — *ARW*
- **`em_hill2d_x`** } 2d ideal — *ARW*
- **`em_squall2d_x`** } 2d ideal — *ARW*
- **`em_squall2d_y`** } 2d ideal — *ARW*
- **`em_grav2d_x`** } 2d ideal — *ARW*

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 **`ideal.exe`** creates WRF initial condition file:

`wrfinput_d01`

Note that wrfbdy file is not needed in idealized cases

Running an *Idealized* Case

- To run the model interactively, type

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

  for single processor (serial) or SMP run. Or

  `mpirun -n 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.
- You may also edit `dyn_em/module_initialize_<case>.F` to change other aspects of the initialization.

***Note*:**

- For 2D cases and baroclinic 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 (at mininum, one must edit **`&time_control`** for start, end and integration times, and **`&domains`** for grid dimensions)

- Run the real-data initialization program:

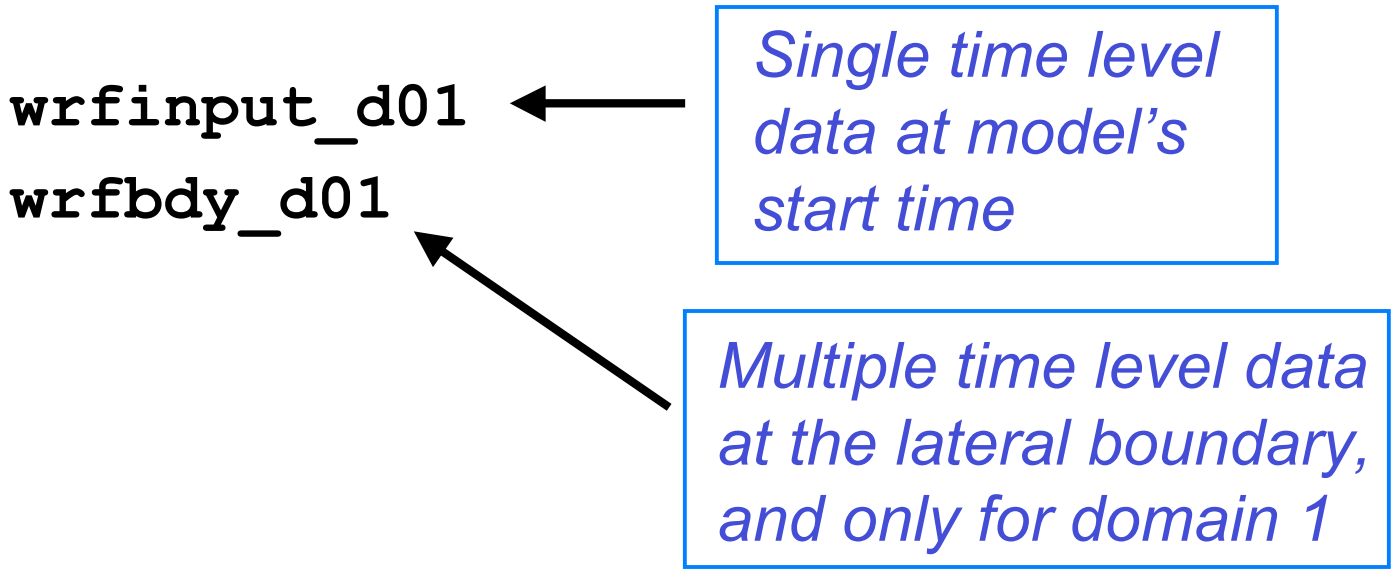
  `./real.exe`, if compiled serially / SMP, or

  `mpirun -n N ./real.exe`, for a MPI job

  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:



Running WRF ARW Real-data Cases

- Run the model executable by typing:

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

  or

  `mpirun -n N ./wrf.exe &`

- Successfully running the model will a create model *history* file:

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

  And *restart* file if selected:

  `wrfrst_d01_<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 (at minimum, one must edit **`&time_control`** for start, end and integration time, and **`&domains`** for grid dimensions)

- Run the real-data initialization program (MPI only):

  `mpirun -n N ./real_nmm.exe`

- Successfully running this program will create model initial and boundary files:

  `wrfinput_d01`

  `wrfbdy_d01`

Running WRF NMM Real-data Cases

- Run the model executable by typing (MPI only):

```
mpirun -n N ./wrf.exe
```

- Successfully running the model will create a model *history* file:

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

And *restart* file if selected:

```
wrfrst_d01_<date>
```

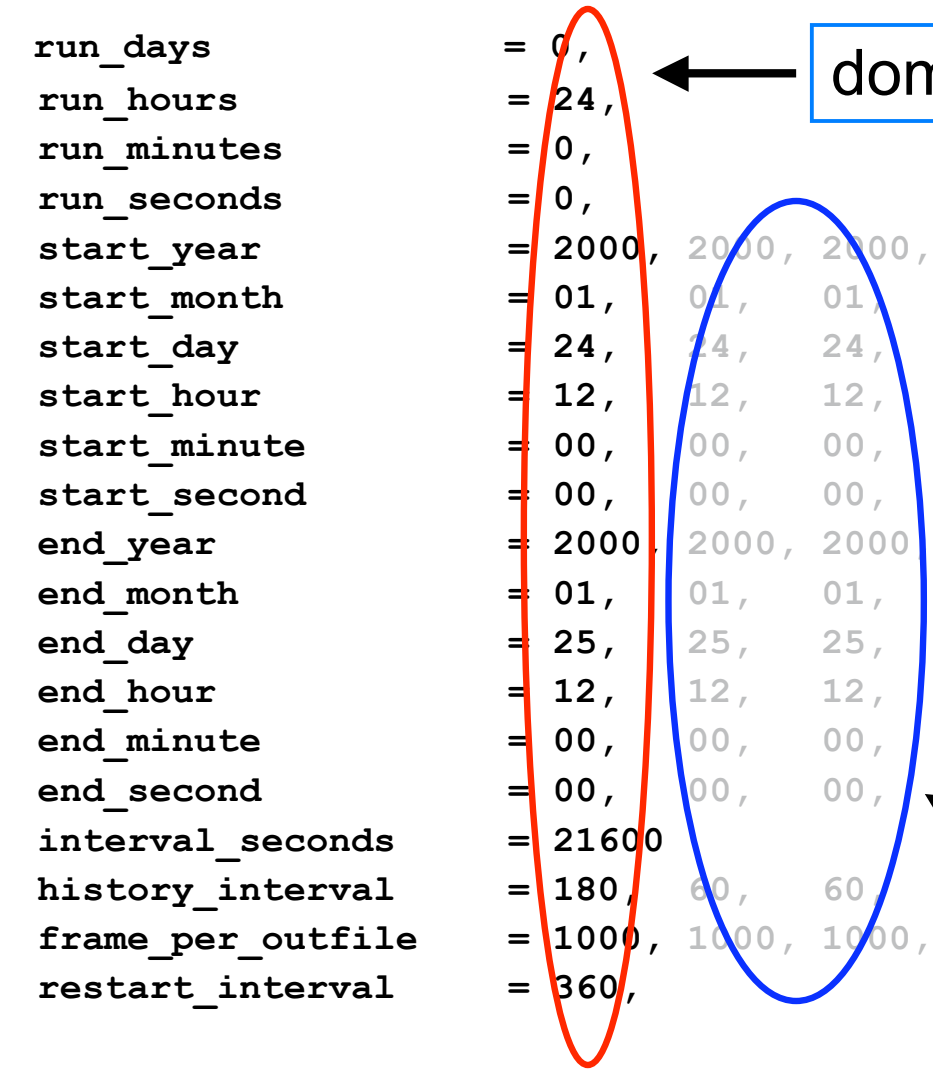
Basic namelist Options

What is a namelist?

- A Fortran namelist contains a list of *runtime* options for the code to read in during its execution. Use of a namelist allows one to change runtime configuration without the need to recompile the source code.

&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,
frame_per_outfile                   = 1000, 1000, 1000,
restart_interval                    = 360,
```



domain 1 option

nest options

Notes on `&time_control`

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

Notes on `&time_control`

- *Interval_seconds*:
  - Time interval between WPS output times, and LBC update frequency
- *history_interval*:
  - Time interval in minutes when a history output time is written
  - The time stamp in a history file name is the time when the history file is first written, and multiple time periods may be written in one file. e.g. a history file for domain 1 that is first written for 1200 UTC Jan 24 2000 is

    `wrfout_d01_2000-01-24_12:00:00`

Notes on `&time_control`

- *frame_per_outfile*:
  - Number of history times written to one file.
- *restart_interval*:
  - Time interval in minutes when a restart file is written.
  - By default, restart file is not written at hour 0.
  - A restart file contains only one time level data, and its valid time is in its file name, e.g. a restart file for domain 1 that is valid for 0000 UTC Jan 25 2000 is

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

&time_control

```
io_form_history      = 2,
io_form_restart      = 2,
io_form_input        = 2,
io_form_boundary     = 2,
debug_level          = 0,
```

IO format options:
- = 1, binary
- = 2, netcdf (most common)
- = 4, PHDF5
- = 5, Grib 1
- =10, Grib 2

Debug print control: Increasing values give more prints.

Notes on `&time_control`

- *frame_per_outfile*:
  - Number of history times written to one file.
- *restart_interval*:
  - Time interval in minutes when a restart file is written.
  - By default, restart file is not written at hour 0.
  - A restart file contains only one time level data, and its valid time is in its file name, e.g. a restart file for domain 1 that is valid for 0000 UTC Jan 25 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                   = 28,    28,    28,
num_metgrid_levels       = 21
dx                       = 30000, 10000, 3333,
dy                       = 30000, 10000, 3333,
eta_levels               = 1.0,0.996,0.99,0.98,… 0.0
p_top_requested          = 5000,
```

nest options

Notes 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.
  - ARW: 6xDX; NMM: 2.25xDX (DX is grid distance in km)
- *s_we, s_sn, s_vert*:
  - Starting indices in X, Y, and Z direction; 1 for domain 1.
- *e_we, e_sn, e_vert*:
  - Model grid dimensions in X, Y and Z directions.
- *num_metgrid_levels*:
  - Number of *metgrid* (input) data levels.
- *dx, dy*:
  - grid distances in meters for ARW; in degrees for NMM.

Notes on `&domains`

- *p_top_requested*:
  - Pressure value at the model top.
  - Constrained by the available data from WPS.

- *eta_levels*:
  - Specify your own model levels from 1.0 to 0.0.

Where do I start?

- Always start with a *namelist* template provided in a test case directory, whether it is a ideal case, ARW or NMM.
- Use document to guide the modification of the namelist values:
  - run/README.namelist
  - User's Guide, Chapter 5

To run a job in a different directory..

- Directories run/ and test_<*case*>/ are convenient places to run, but it does not have to be.
- Copy or link the content of these directories to another directory, including physics data files, wrf input and boundary files and wrf namelist and executables, and you should be able to run a job anywhere on your system.

Check Output

Output After a Model Run

- Standard out/error files:

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

- Model history file(s):

  `wrfout_d01_<date>`

- Model restart file(s), optional

  `wrfrst_d01_<date>`

Output from a multi-processor run

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

mpirun -n 4 .wrf.exe ➔

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 out File?

Check run log file by typing

`tail wrf.out`, or

`tail rsl.out.0000`

You should see the following if the job is successfully completed:

`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** (great tools!)
- 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
```

Simple Trouble Shooting

Often-seen runtime problems

- `module_configure: initial_config: error reading namelist`

  > Typos exist in *namelist.input* file

- `input_wrf.F: SIZE MISMATCH: namelist ide,jde,num_metgrid_levels= 70 61 27 ; input data ide,jde,num_metgrid_levels= 74 61 27`

  > Grid dimensions in error

Often-seen runtime problems

- `Segmentation fault (core dumped)`
  > Often typing ‘unlimit’ or equivalent can help when this happens quickly in a run.

- `121 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`
  > Model becomes unstable due to various reasons. If it happens soon after the start time, check input data, and/or reduce time step.

References

- Information on compiling and running WRF, and a more extensive list of namelist options and their definition / explanations can be found in the ARW and NMM User's Guide, Chapter 5
- Also see '*Nesting Setup and Run*' talk.