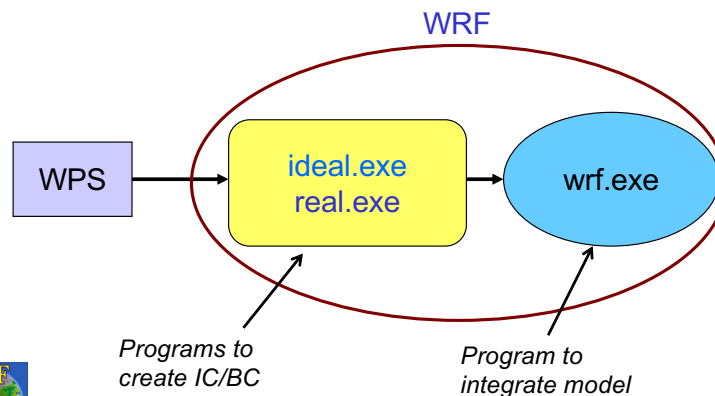


## Set Up and Run WRF (*real* and *ideal* data)

Wei Wang  
January 2019



## WRF System Flowchart



## Outline

- Running WRF code
  - Things to check before you run..
  - Running *real-data* case
  - Running *idealized* case
- Basic runtime options for a **single** domain run (*namelist*)
- Check output
- Simple trouble shooting
- Running a nested case: later



## Before You Run ..

- Top directory is now **WRF/**
- Make sure appropriate executables are created in **WRF/main/** directory:
  - **ideal.exe** – executable to create idealized IC
  - **real.exe** – executable to create IC/BC
  - **wrf.exe** – executable for model integration
  - **ndown.exe** – utility
  - **tc.exe** – utility routine for TC bogusing
- If you are working with real data, be sure that files for **a few time periods** from WPS are correctly generated:
  - **met\_em.d01.\***



## WRF test case directories

You have these choices in **WRF/test/**

(choices made at compile time. E.g. *compile em\_real*):

```

em_real      } 3-dimensional real-data – real.exe
em_quarter_ss
em_b_wave
em_les
em_tropical_cyclone
em_heldsuarez
em_hill2d_x
em_squall2d_x
em_squall2d_y
em_grav2d_x
em_seabreeze2d_x
em_scm_xy    } 1d ideal

```

*3d ideal*

*2d ideal*

*1d ideal*

*ideal.exe*



## Steps to Run

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



## WRF/run directory

```

README.namelist } description of namelists
LANDUSE.TBL
GENPARM.TBL
SOILPARM.TBL
VEGPARM.TBL
URBPARM.TBL
RRTM_DATA
RRTMG_SW_DATA
RRTMG_LW_DATA
CAM_ABS_DATA
CAM_AEROPT_DATA
ozone.formatted
ozone_lat.formatted
ozone_plev.formatted
aerosol.formatted
aerosol_lat.formatted
aerosol_lon.formatted
aerosol_plev.formatted
gribmap.txt
grib2map.tbl
.... (a total of 60 files)

```

*These are model physics data files: they are used to either initialize physics variables, or make physics computation faster*

*\* Some of these files are text files, hence editable*

*for grib IO*



## WRF/run directory after compile

```

LANDUSE.TBL
SOILPARM.TBL
VEGPARM.TBL
GENPARM.TBL
URBPARM.TBL
RRTM_DATA
RRTMG_SW_DATA
RRTMG_LW_DATA
ozone.formatted
ozone_lat.formatted
ozone_plev.formatted
...
namelist.input - copied from ../test/em_real/namelist.input
real.exe -> ../main/real.exe
wrf.exe -> ../main/wrf.exe
ndown.exe -> ../main/ndown.exe
.... (a few more)

```

*An example after em\_real case compile*



## Running a Real-Data Case



9

Mesoscale & Microscale Meteorological Laboratory / NCAR

## Running a 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
  - tc.exe* - program for TC bogusing
- These executables are linked to:
  - WRF/run
  - and
  - WRF/test/*em\_real*



➔ One can go to either directory to run.

Mesoscale & Microscale Meteorological Laboratory / NCAR 10

## WRF/test/*em\_real* directory

```
LANDUSE.TBL -> ../../run/LANDUSE.TBL
GENPARM.TBL -> ../../run/GENPARM.TBL
SOILPARM.TBL -> ../../run/SOILPARM.TBL
VEGPARM.TBL -> ../../run/VEGPARM.TBL
URBPARM.TBL -> ../../run/URBPARM.TBL
RRTM_DATA -> ../../run/RRTM_DATA
RRTMG_SW_DATA -> ../../run/RRTMG_SW_DATA
RRTMG_LW_DATA -> ../../run/RRTMG_LW_DATA
ozone.formatted -> ../../run/ozone.formatted
ozone_lat.formatted -> ../../run/ozone_lat.formatted
ozone_plev.formatted -> ../../run/ozone_plev.formatted
...
namelist.input - editing required
real.exe -> ../../main/real.exe
wrf.exe -> ../../main/wrf.exe
ndown.exe -> ../../main/ndown.exe
.... (many more)
```



Mesoscale & Microscale Meteorological Laboratory / NCAR 11

## Running a Real-data Case

- One must successfully run WPS to prepare data required, and create *met\_em.\** files for multiple time periods for initial and boundary conditions
- Move or link WPS/metgrid output files to the run directory:

```
cd test/em_real
ln -s ../../../../WPS/met_em.d01.* .
```



Mesoscale & Microscale Meteorological Laboratory / NCAR 12

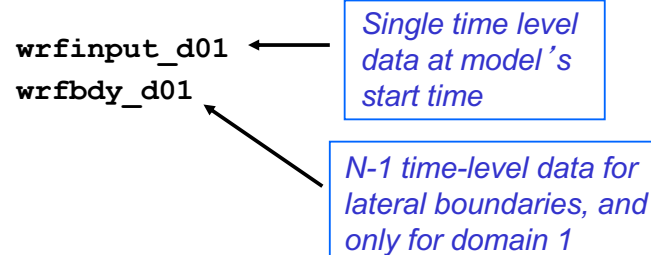
## Running a Real-data Case

- Edit `namelist.input` file for runtime options (*at minimum*, 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 -np N ./real.exe` for a MPI job  
 where `N` is the number of processors requested.



## Running a Real-data Case

- Successfully running `real.exe` will create model initial and boundary files:



*N: the number of time periods processed*

`ncdump -v Times wrfbdy_d01`



## Running a Real-data Case

- Typing '`ncdump -v Times wrfbdy_d01`' will give you, for a 24 hour period, 6 hourly data interval:  
 .. a bunch of prints and at the end:

```
data:
Times =
  "2005-08-28_00:00:00",
  "2005-08-28_06:00:00",
  "2005-08-28_12:00:00",
  "2005-08-28_18:00:00" ;
```

\* BC data consists of values at the start of the time interval and rate of change in the time interval.



## Running a Real-data Case

- Run the model executable by typing:  
`./wrf.exe >& wrf.out &`  
 or  
`mpirun -np N ./wrf.exe &`
- Successfully running the model will create model history file:  
`wrfout_d01_2005-08-28_00:00:00`  
*Based on start date set in namelist*  
 and a restart file if `restart_interval` is set to a time within the range of the forecast time:

`wrfirst_d01_2005-08-28_12:00:00`

*Exact time at a restart*



## Running a Real Data Case

wrfout\_d01\_2005-08-28\_00:00:00

*Based on start date set in namelist*

```
start_year      = 2008, 2008, 2008,
start_month     = 08, 08, 08,
start_day       = 28, 28, 28,
start_hour      = 00, 00, 00,
start_minute    = 00, 00, 00,
start_second    = 00, 00, 00,
end_year        = 2008, 2008, 2008,
end_month       = 08, 08, 08,
end_day         = 29, 29, 29,
end_hour        = 00, 00, 00,
end_minute      = 00, 00, 00,
end_second      = 00, 00, 00,
```



## Running an Idealized Case



## Running an *Idealized* Case

- An idealized case refers to data in the initial condition file (no need to run WPS)
- If you have compiled an ideal case, you should have:
  - `ideal.exe` – program to create idealized initial condition
  - `wrf.exe` – model executable
- These executables are linked to:
  - `WRF/run`
  - and
  - `WRF/test/em_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 relevant physics data files to the current 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 2D input (e.g. *input\_jet* in *em\_b\_wave* case).

Running *ideal.exe* only creates WRF initial condition file: *wrfinput\_d01*



## Running an *Idealized* Case

Note that wrfbdy file is not needed for idealized cases.

Instead, the boundary condition options are set in the *namelist.input* file. For example, these are for options in east-west, or x direction:

```
periodic_x      = .true.,  
symmetric_xs    = .false.,  
symmetric_xe    = .false.,  
open_xs         = .false.,  
open_xe         = .false.,
```



## Running an *Idealized* Case

- To run the model interactively, type  

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

for single processor (serial) or SMP run. Or  

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

for a MPI run (3D cases only)
- 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*

*Based on start date set in namelist*  
(dates are important for radiation physics)



## Running an *Idealized* Case

*wrfout\_d01\_0001-01-01\_00:00:00*

*Based on start date set in namelist*

```
start_year      = 0001, 0001, 0001,  
start_month     = 01, 01, 01,  
start_day       = 01, 01, 01,  
start_hour      = 00, 00, 00,  
start_minute    = 00, 00, 00,  
start_second    = 00, 00, 00,  
end_year        = 0001, 0001, 0001,  
end_month       = 01, 01, 01,  
end_day         = 01, 01, 01,  
end_hour        = 00, 00, 00,  
end_minute      = 120, 120, 120,  
end_second      = 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 (*more on Thur.*)

### 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
- For the 1D case, compile and run serially



## 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.
- Fortran 90 namelist has very specific format, so edit with care:  

```
&namelist-record - start  
/  
- end
```
- As a general rule:
  - Multiple columns: domain dependent
  - Single column: value valid for all domains

A namelist file may contain a number of records



## namelist record `&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,  
frames_per_outfile = 1000, 1000, 1000,  
restart_interval = 360,  
restart        = .true.,
```

domain 1 option

for nests



## 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 if `run_*` variables are not set (or set to 0)



## Notes on `&time_control`

- `interval_seconds`:
  - Time interval between WPS output times, and lateral BC (and lower BC) update frequency
- `history_interval`:
  - Time interval in minutes when a history output is written (note output is instantaneous)
  - If the `time_step` cannot be evenly divided by `history_interval`, then nearest time step output is used
  - 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`

- `frames_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 valid for 0000 UTC Jan 25 2000 is  
`wrfrst_d01_2000-01-25_00:00:00`
- `restart`:
  - whether this is a restart run



## Notes on `&time_control`

Example 1: all output times are in a single file

```
history_interval = 180, 60, 60,  
frames_per_outfile = 1000, 1000, 1000,  
wrfout_d01_2000-01-24_12:00:00
```

Example 2: each output file only contains a single time

```
history_interval = 180, 60, 60,  
frames_per_outfile = 1, 1, 1,  
wrfout_d01_2000-01-24_12:00:00  
wrfout_d01_2000-01-24_15:00:00  
wrfout_d01_2000-01-24_18:00:00
```





## Notes on *restart*

- What is a *restart* run?
  - A restart run is a continuation of a model run
- How to do a *restart* run:
  - In the first run, set *restart\_interval* to a value that is within the model integration time
  - A restart file will be created. e.g.  
`wrfrst_d01_2000-01-25_00:00:00`
- When doing a restart run:
  - Set *restart* = .true.,
  - Set start time to restart time
  - Set run\_\* to be the hours remaining in the run



## &time\_control

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

### IO format options:

= 1, binary  
 = 2, **netcdf** (most common)  
 = 4, PHDF5  
 = 5, Grib 1  
 = 10, Grib 2  
 = 11, pnetCDF

### For large files:

*io\_form\_restart* = 102 :  
 write output in patch  
 sizes: fast for large grid  
 and useful for restart file



## namelist record &domains

```
time_step = 180
time_step_fract_num = 0,
time_step_fract_den = 1,
max_dom = 1,
e_we = 74, 112, 94,
e_sn = 61, 97, 11,
e_vert = 33, 28, 28,
num_metgrid_levels = 32,
num_metgrid_soil_levels = 4
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\_fract\_den*:
  - Time step for model integration in seconds
  - Fractional time step specified in separate integers of numerator and denominator
  - Typically 5 to 6xDX (DX is grid distance in km)
- *e\_we*, *e\_sn*, *e\_vert*:
  - Model grid dimensions (staggered) in X, Y and Z directions, need to match those defined in *geogrid* program
- *num\_metgrid\_levels*:
  - Number of *metgrid* (input) data levels
- *num\_metgrid\_soil\_levels*:
  - Number of soil data levels in the input data
- ➡ Both can be found by typing `ncdump -h met_em.d01.<date> | more`
- *dx*, *dy*:
  - grid distance: in meters



## Notes on &domains

- *p\_top\_requested*:
  - Pressure value at the model top
  - Constrained by the available data from WPS
  - Default is 5000 Pa (recommended as lowest model top)
- *eta\_levels*:
  - Specify your own model levels from 1.0 to 0.0
  - If not specified, program *real* will calculate a set of levels
    - V4 has a new and better way to compute the levels
  - Use a minimum of 33 or more levels with 5000 Pa model top to limit vertical grid distance < 1 km. Use more vertical levels when decreasing horizontal grid sizes.



## namelist record &bdy\_control

	typical	optional	
<b>spec_bdy_width</b>	= 5,	(10)	
spec_zone	= 1,	(1)	← do not change
relax_zone	= 4,	(9)	
specified	= .true.,		
nested	= .false.,		

May change **relax\_zone**  
and **spec\_bdy\_width**  
(**spec\_zone** + **relax\_zone**  
= **spec\_bdy\_width**)

\* Wider boundary zone may work  
better for coarser driving data



## Other namelists

### &physics:

- Model physics options

### &dynamics:

- Damping, diffusion options
- Advection options
- In 4.0, the hybrid vertical coordinate option is the default.  
Turn it off by setting the following for *real* and *wrf*:  
**hybrid\_opt** = 0



## Where do I start?

- Always start with a *namelist* template provided in a test case directory, whether it is an ideal case, or a real data case.
  - A number of namelist templates are provided in *test/test\_<case>/* directories

For example: in *test/em\_real/*, there are

**namelist.input.4km** ~ 4 km grid size  
**namelist.input.jun01** ~ 10 km grid size  
**namelist.input.jan00** ~ 30 km grid size



## Where do I start?

- For different applications, please refer to p5-38 to 5-40 of the ARW User's Guide:
  - 2 or 4 km microphysics-only runs
  - 20 – 30 km, 2 – 3 day runs
  - Antarctic region
  - Tropical storm forecasting
  - Regional climate
  - Try physics suites (since V3.9)



## Where do I start?

- Use document to guide the modification of the namelist values:
  - `run/README.namelist`
  - `test/em_real/examples.namelist`
  - User's Guide, Chapter 5 (online version has the latest)
  - Full list of namelists and their default values can be found in Registry files: [Registry.EM\\_COMMON](#), `registry.io_boilerplate` (for IO options) and other registry files - look for character string '*namelist*'



## 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, 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 `rs1.*` files
- Model history file(s):  
`wrfout_d01_<date>`
- Model restart file(s), optional  
`wrfirst_d01_<date>`



## Output from a multi-processor run

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

```
mpirun -np 4 ./wrf.exe ➔
```

<code>rs1.out.0000</code>	<code>rs1.error.0000</code>
<code>rs1.out.0001</code>	<code>rs1.error.0001</code>
<code>rs1.out.0002</code>	<code>rs1.error.0002</code>
<code>rs1.out.0003</code>	<code>rs1.error.0003</code>

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 rs1.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** (great tool!)
- Use post-processing tools (see talks later)



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

- Model version, decomposition info:

```
Ntasks in X      2, ntasks in Y      4
WRF V4.0 MODEL
```

- Time taken to compute one model step:

```
Timing for main: time 2000-01-24_20:03:00 on domain 1: 0.89475 elapsed seconds
Timing for main: time 2000-01-24_20:06:00 on domain 1: 0.09011 elapsed seconds
Timing for main: time 2000-01-24_20:09:00 on domain 1: 0.08634 elapsed seconds
Timing for main: time 2000-01-24_20:12:00 on domain 1: 0.09004 elapsed seconds
```

- Time taken to write history and restart file:

```
Timing for Writing wrfout_d01_2000-01-25_00:00:00 for domain 1: 0.07091 elapsed seconds
```

- Any model error prints:

```
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
```



→ An indication the model has become numerically unstable

## Simple Trouble Shooting



## Often-seen runtime problems

- module\_configure: initial\_config: **error reading**

**namelist: &dynamics**

> Typos or erroneous namelist variables exist in namelist record *&dynamics* 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 '**ulimit -s unlimited**' or equivalent can help when this happens quickly in a run, and on a small computer

- If you do: `grep cfl rsl.error.*` and see

```
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 [User's Guide, Chapter 5](#)
- Also see nesting talk and demonstration tomorrow.

