WRF ARW
How to Set-up and Run

Courtesy to NCAR

Download WRF

- Download WRF (unified ARW + NMM) source code from

  http://www.mmm.ucar.edu/wrf/users/downloads.html
- Then select the Download tab, and choose WRF V2, where you are interrogated relentlessly
- What you will get is

  `WRFV***.TAR.gz` **`*** version #`**

Unzip and Untar Downloaded WRF File

`cd to_where_you_want_the_source`

- For the practice, this is also where the code will be run.

`tar xvzf WRFV***.TAR.gz`

- After **`gunzip`** and **`untar`**, you should see a directory **`WRFV2/`**

`cd WRFV3`

WRF
Top-Level
Directory

Unified
ARW/NMM

```
Makefile
README
README_test_cases
clean
compile
configure
Registry/
arch/
dyn_em/
dyn_nnm/
external/
frame/
inc/
main/
phys/
share/
tools/
run/
test/
```

build scripts

CASE input files

machine build rules

source code directories

execution directories

Configure WRF Code for Core, Machine, and Parallel Option

- Run configuration script, detects available options based on **`uname -a`**
- Guesses made for netcdf location, an environment variable can be set (B shell)

  `export NETCDF=/usr/local/netcdf`
- To Install ARW model

  `export WRF_EM_CORE=1`
- To Install NMM model

  `export WRF_NMM_CORE=1`

- Do not ***accidently*** get the system configure command

  `./configure`

In the interest of clarity, only the PGI options are shown for the IA32 Linux (classroom test machines).

Note the "allows nesting" and "no nesting" options.

```
Please select from among the following supported platforms.

   1.  PC Linux i486 i586 i686, PGI compiler  (Single-threaded, no nesting)
   2.  PC Linux i486 i586 i686, PGI compiler  (single threaded, allows nesting using RSL without
    MPI)
   3.  PC Linux i486 i586 i686, PGI compiler  SM-Parallel (OpenMP, no nesting)
   4.  PC Linux i486 i586 i686, PGI compiler  SM-Parallel (OpenMP, allows nesting using RSL
    without MPI)
   5.  PC Linux i486 i586 i686, PGI compiler  DM-Parallel (RSL, MPICH, Allows nesting)
   6.  PC Linux i486 i586 i686, PGI compiler  DM-Parallel (RSL_LITE, MPICH, Allows nesting)

Enter selection [1-6] :
```

Then a **configure.wrf** will be created.

Sample of what is inside a **`configure.wrf`** file

```
FC                      =       pgf90
LD                      =       pgf90
CC                      =       gcc -DFSEEKO64_OK
SCC                     =       $(CC)
RWORDSIZE               =       $(NATIVE_RWORDSIZE)
SFC                     =       $(FC)
CFLAGS                  =
FCOPTIM                 =       -O2 # -fast
FCDEBUG                 =       #-g
FCBASEOPTS              =       -w -byteswapio -Mfree -tp
  p6 $(FCDEBUG)
FCFLAGS                 =       $(FCOPTIM) $(FCBASEOPTS)
```

Compiling WRF model

Available compile targets are the names of the directories under **`./WRFV3/test`**, i.e.

`./compile em_real`

em_scm_xy — 1d ideal

em_grav2d_x
em_hill2d_x
em_squall2d_x
em_squall2d_y
em_seabreeze2d_x
} Ideal 2d

em_b_wave
em_quarter_ss
em_les
em_heldsuarez
} Ideal 3d

Ideal

ARW

em_real
nmm_real
} Real 3d → NMM

WRF Executables: Names and Locations

- The WRF executable programs are built in the **`./WRFV3/main`** directory
- The executables are linked both into the **`./WRFV3/run`** directory and to the **`./WRFV3/test/em_real`** directory (if you compiled with real data option)

- For real data cases, the executables built are:

  `ndown.exe`

  `nup.exe`

  `real.exe`

  `wrf.exe`

  `tc.exe`

- For ideal data cases, the executables built are:

  `ideal.exe`

  `wrf.exe`

Running WRF Executables - Preparations

- Real data cases require additional input files from the WPS (package) to be in (or linked into) the run-time directory
- There are several physics related input files that are automatically linked into the run-time directories (look-up tables)
- Edit the run-time configurable options in the **`namelist.input`** file located in the **`./WRFV3/test/<em_real>`** **`(or)`** **`/WRFV3/run`**

WRFV3/run directory

README.namelist
LANDUSE.TBL
GENPARM.TBL
SOILPARM.TBL
VEGPARM.TBL
URBAN_PARAM.TBL
RRTM_DATA
RRTMG_SW_DATA
RRTMG_LW_DATA
CAM_ABS_DATA
CAM_AEROPT_DATA
ozone.formatted
ozone_lat.formatted
ozone_plev.formatted
ETAMPNEW_DATA
tr49t67
tr49t85
tr67t85
…. (a few more)

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

After successful compilation WRFV3/run directory

gribmap.txt
grib2map.tbl
namelist.input* -> ../test/*em_real/namelist.input
ideal.exe -> ../main/ideal.exe
real.exe -> ../main/real.exe
wrf.exe -> ../main/wrf.exe
ndown.exe -> ../main/ndown.exe
tc.exe -> ../main/tc.exe
…. (a few more)

An example after ARW real case compilation

Running Real Data Case

WRFV3/test/em_*real* directory

LANDUSE.TBL -> ../../run/LANDUSE.TBL
ETAMPNEW_DATA -> ../../run/ETAMPNEW_DATA
GENPARM.TBL -> ../../run/GENPARM.TBL
RRTM_DATA -> ../../run/RRTM_DATA
RRTMG_SW_DATA -> ../../run/RRTMG_SW_DATA
RRTMG_LW_DATA -> ../../run/RRTMG_LW_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 Real Data Case

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.d01<date> .

Running Real Data Case

- 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 -np *N* ./real.exe, or

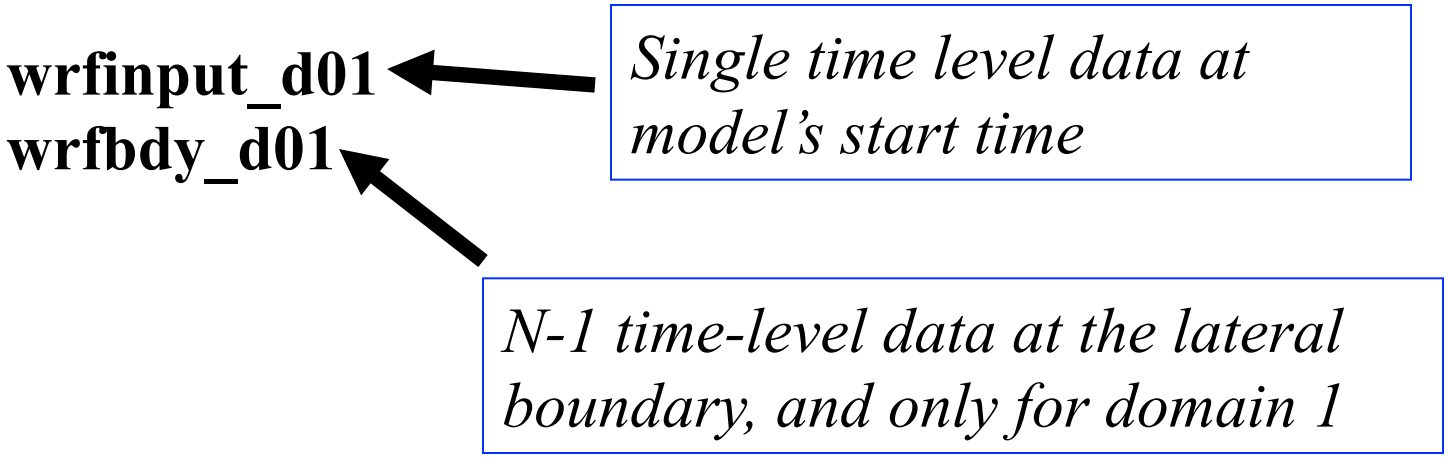
  mpirun -machinefile *file* -np *N* ./real.exe for a MPI job

where ***N*** is the number of processors requested, and

file has a list of CPUs for the MPI job

Running Real Data Case

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



N: the number of time periods processed

- Run the model executable by typing:

  ./wrf.exe >& wrf.out &

  or

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

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

  wrfout_d01_yyyy-mm-dd_hh:00:00

And *restart* file if **restart_interval** is set to a time within the range of the forecast time:

wrfrst_d01_yyyy'-mm'-dd'_hh'::00:00

Note that **yyyy-mm-dd_hh and yyyy'-mm'-dd'_hh' are not same**

Basic namelist Options

What is a name list?

- 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

What the namelist file contains?

&time_control
……..
/
&domains
……..
/
&physics
……..
/
&fdda
……..
/
&tc
……..
/
&bdy_control
……..
/
&dynamics **etc……**

&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,
restart            = .true.,
```

***run_** * time variables:*
– Model simulation length: *wrf.exe* and domain 1only

***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).

→ Time interval between WPS output times, and LBC update frequency

→ Time interval in minutes when a output is written

→ Number of history times written to one file

→ Time interval in minutes when a restart file is written

→ whether this is a restart run

&domains

time_step = 180
time_step_fract_num = 0,
time_step_fract_den = 1,

– 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)
– May be divided by output intervals

max_dom = 1,
e_we = 74, 112, 94,
e_sn = 61, 97, 91,
e_vert = 28, 28, 28,
num_metgrid_levels = 21
num_metgrid_soil_levels = 4
dx = 30000, 10000, 3333,
dy = 30000, 10000, 3333,

grid distances: in meters for ARW

eta_levels = 1.0,0.996,0.99,0.98,… 0.0
p_top_requested = 5000,

– Pressure value at the model top.
– Constrained by the available data from WPS.
– Default is 5000 Pa

• *eta_levels*:
– Specify your own model levels from 1.0 to 0.0.
– If not specified, program *real* will calculate at pre-defined levels

```
&dynamics
 rk_ord                    = 3,      ; time-integration scheme option:
                                     2 = Runge-Kutta 2nd order
                                     3 = Runge-Kutta 3rd order
/
&tc                                  ; controls for tc_em.exe ONLY, no impact on real, ndown, or model
 insert_bogus_storm          = .false.  ; T/F for inserting a bogus tropical storm (TC)
 remove_storm                = .false.  ; T/F for only removing the original TC
 num_storm                   = 1        ; Number of bogus TC
 latc_loc                    = -999.    ; center latitude of the bogus TC
 lonc_loc                    = -999.    ; center longitude of the bogus TC
 vmax_meters_per_second      = -999.    ; vmax of bogus storm in meters per second
 rmax                        = -999.    ; maximum radius outward from storm center
 vmax_ratio                  = -999.    ; ratio for representative maximum winds, 0.75 for 45 km grid, and
                                          0.9 for 15 km grid.
/
```

&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
=11, pNetCDF

Debug print control:
Increasing values give more prints.

Useful alternative:
io_form_restart = 102 :
write output in patch sizes: fast for large grids and useful for restart file

Checking Output: **`ideal.exe`**

- The WRF pre-processor **`ideal.exe`** produces a single input file:

  `wrfinput_d01`

Checking Output: **`real.exe`**

- The WRF pre-processor real.exe produces an input file for each domain, and a lateral boundary file for the outer-most grid

  `wrfbdy_d01`

  `wrfinput_d01, wrfinput_d02`

- Optionally, a lower boundary file is generated containing the SST and sea ice for each domain (for long simulations)

  `wrfloinp_d01, wrfloinp_d02`

Checking Output: **`wrf.exe`**

- Standard out/error files: **wrf.out**, or **rsl**.* files
- Model history file(s): **wrfout_d01_<date>**
- Model restart file(s), **wrfrst_d01_<date>** (optional)

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

Checking Output: Standard Printout

- For WRF, the time steps are of interest:

```
 WRF NUMBER OF TILES =   1
Timing for main: time 2000-01-24_12:03:00 on domain   1:   13.95000 elapsed
     seconds.
Timing for main: time 2000-01-24_12:06:00 on domain   1:    2.53000 elapsed seconds.
Timing for main: time 2000-01-24_12:09:00 on domain   1:    2.54000 elapsed seconds.
Timing for main: time 2000-01-24_12:12:00 on domain   1:    2.54000 elapsed seconds.
Timing for main: time 2000-01-24_12:15:00 on domain   1:    2.56000 elapsed seconds.
Timing for main: time 2000-01-24_12:18:00 on domain   1:    2.55000 elapsed seconds.
Timing for main: time 2000-01-24_12:21:00 on domain   1:    2.56000 elapsed seconds.
Timing for main: time 2000-01-24_12:24:00 on domain   1:    2.56000 elapsed seconds.
Timing for main: time 2000-01-24_12:27:00 on domain   1:    2.56000 elapsed seconds.
Timing for main: time 2000-01-24_12:30:00 on domain   1:   13.29000 elapsed
     seconds.
Timing for Writing wrfout_d01_2000-01-24_12:30:00 for domain        1:    0.29000
     elapsed seconds.
wrf: SUCCESS COMPLETE WRF
```

Nesting Run

Some Nesting Hints

- Allowable domain specifications
- Defining a starting point
- Illegal domain specifications
- 1-way *vs* 2-way nesting

Before You Run ..

• Make sure you have selected basic nest compile options and appropriate executables are created in W**RFV3/main/** directory:

• If you are running a real-data case, be sure that files for *nest* domains from WPS are generated:

– **met_em.d01.<*date*>, met_em.d0*.<*date*>** for ARW

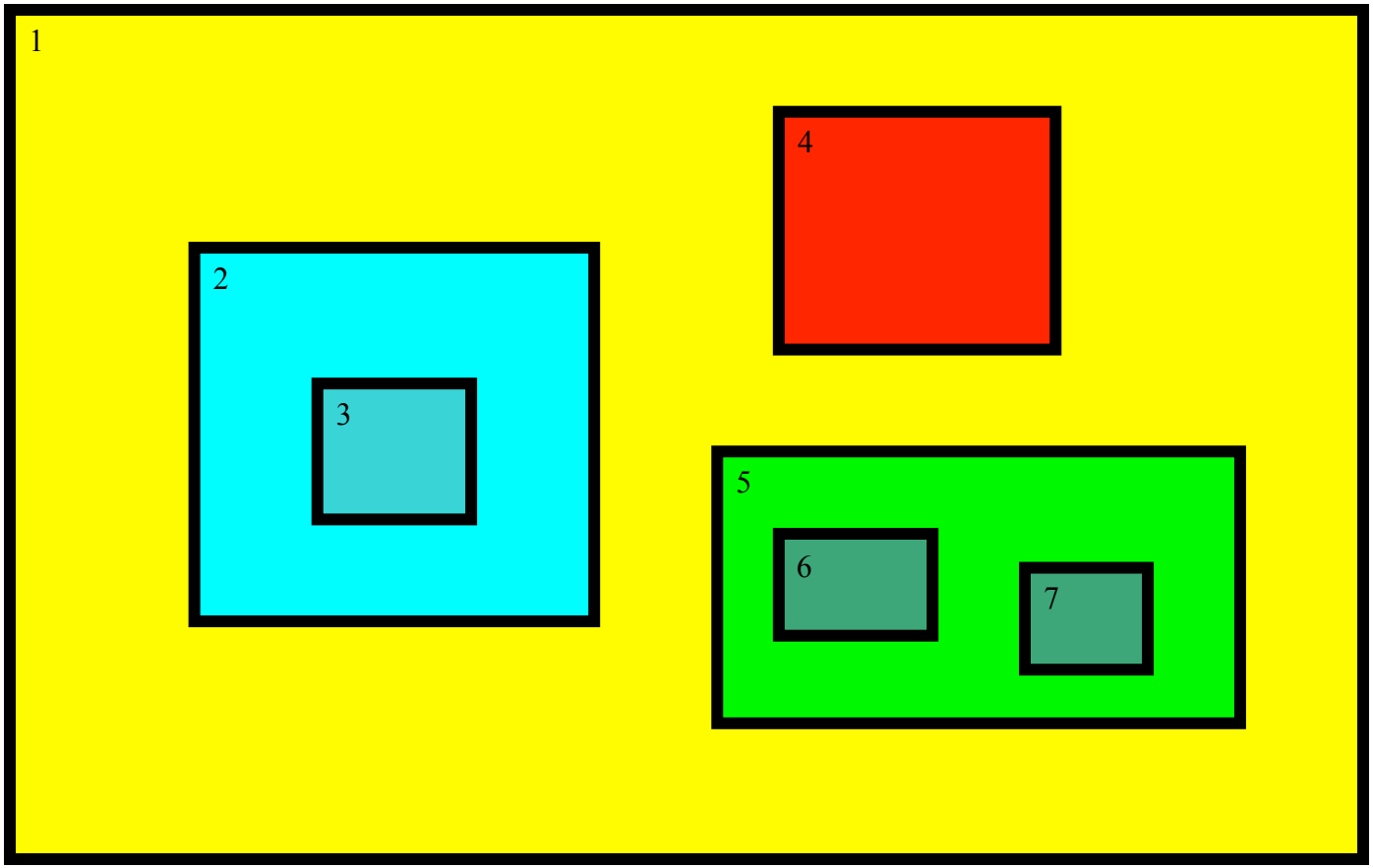
Important to note:

• Key variable: **max_dom** must be set to >= 2

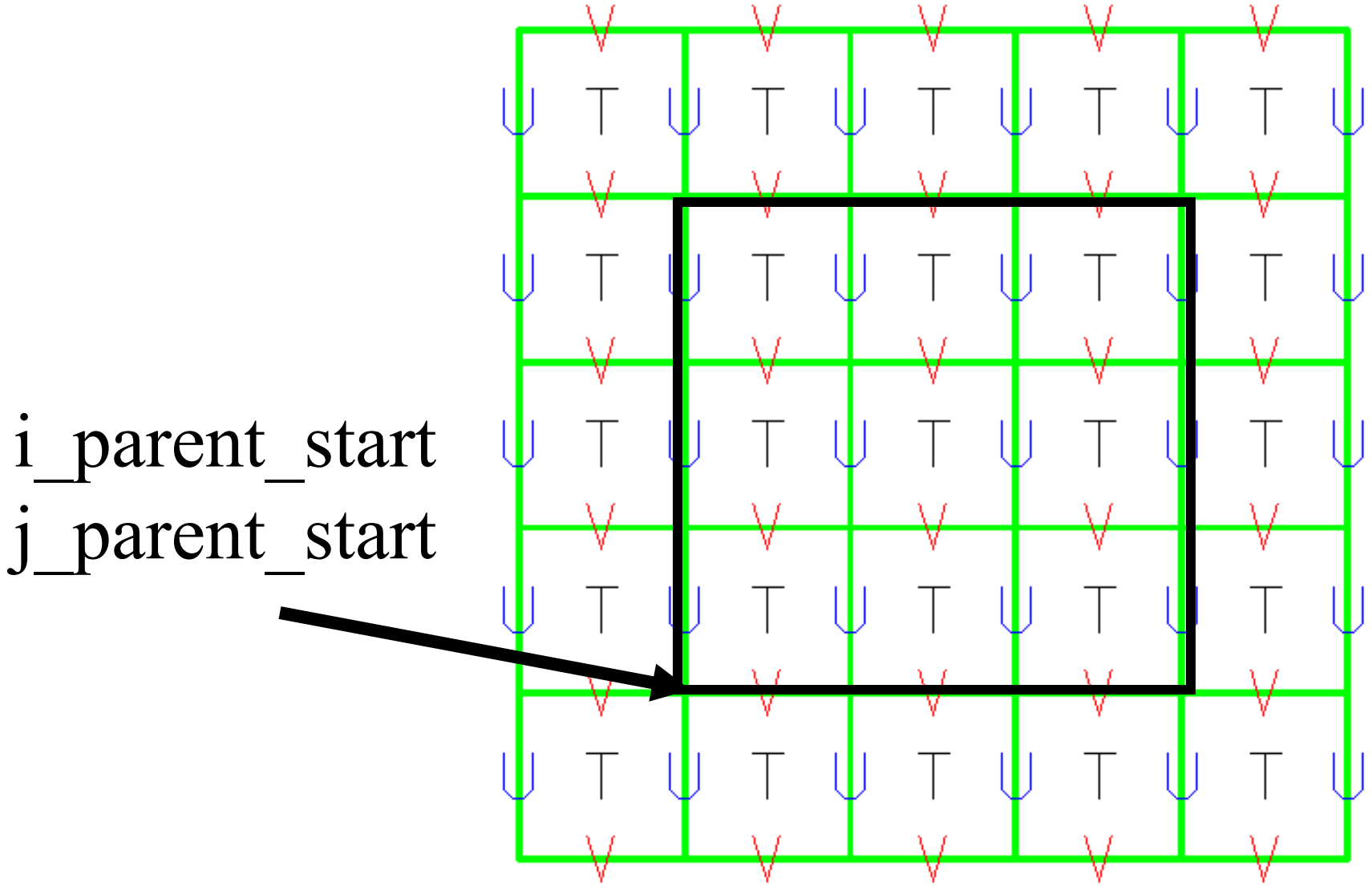
• Need to pay attention to multi-column name lists

This is OK

Telescoped to any depth
Any number of siblings

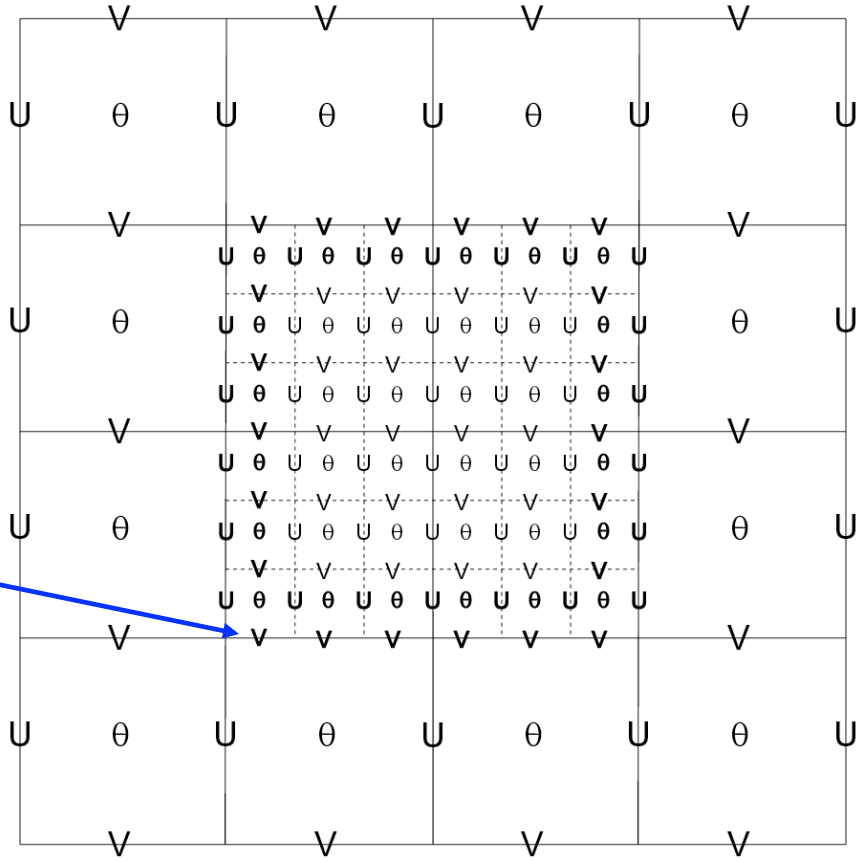


Coarse Grid Staggering



ARW Coarse Grid Staggering 3:1 Ratio

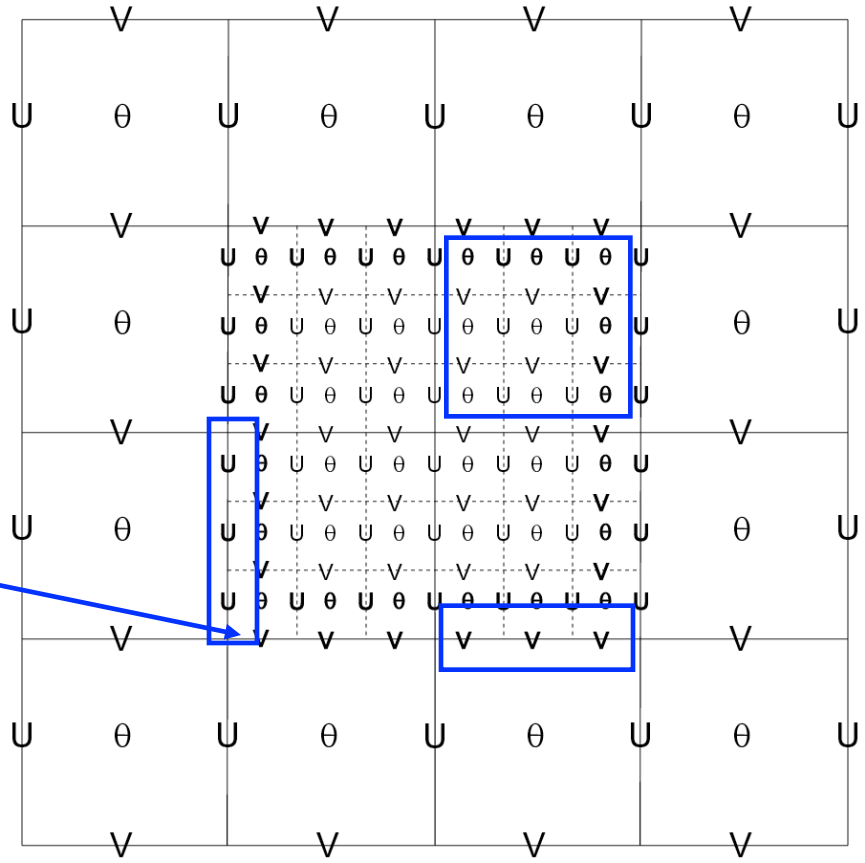
Starting Location I = 31



CG … 30 31 32 33 34

ARW Coarse Grid Staggering 3:1 Ratio

Starting Location I = 31

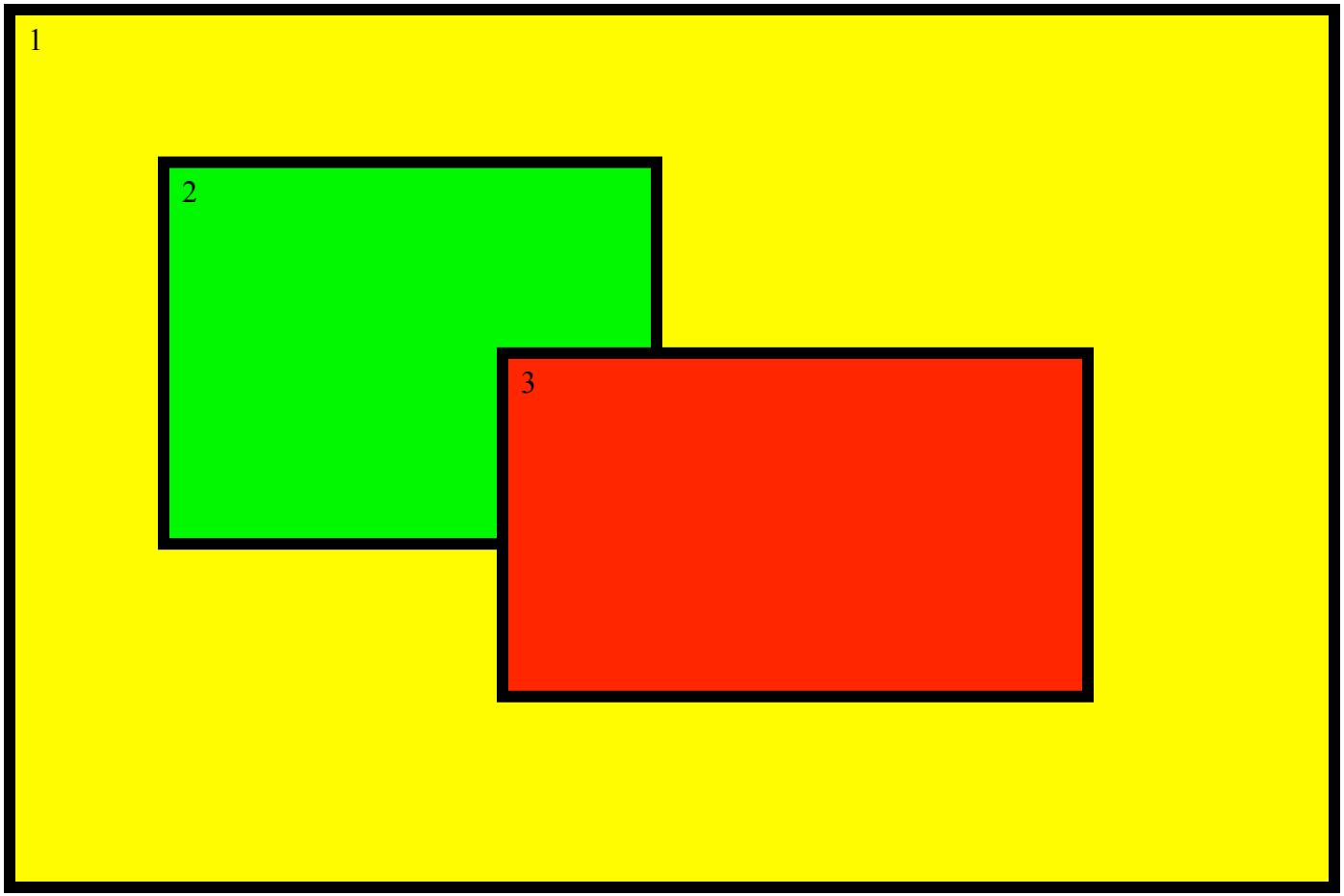


Feedback:
U : column
V : row
T : cell

CG … 30 31 32 33 34

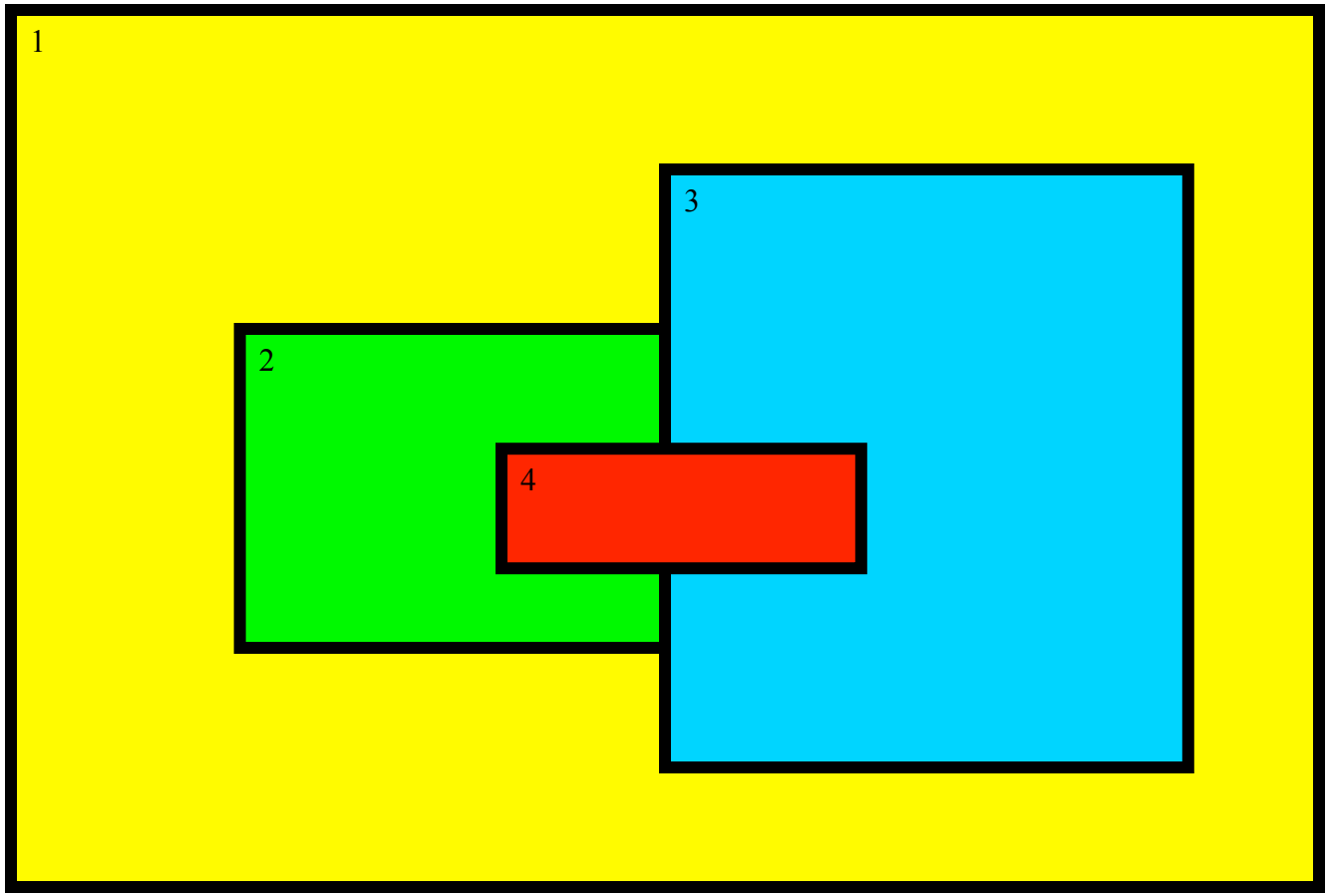
Not OK

Domains may not have overlapping points in the CG



Not OK either

Domains have 1 and only 1 parent



1-way *vs*. 2-way Nesting

- wrf integrates 1 domain at a time
- CG forces FG through lateral boundaries
- No FG to CG feedback
- ndown run between CG wrf and FG wrf (or shut off feedback)

- wrf integrates 2 domains at a time
- CG forces FG at every FG timestep
- FG to CG feedback at every CG timestep
- ndown not required

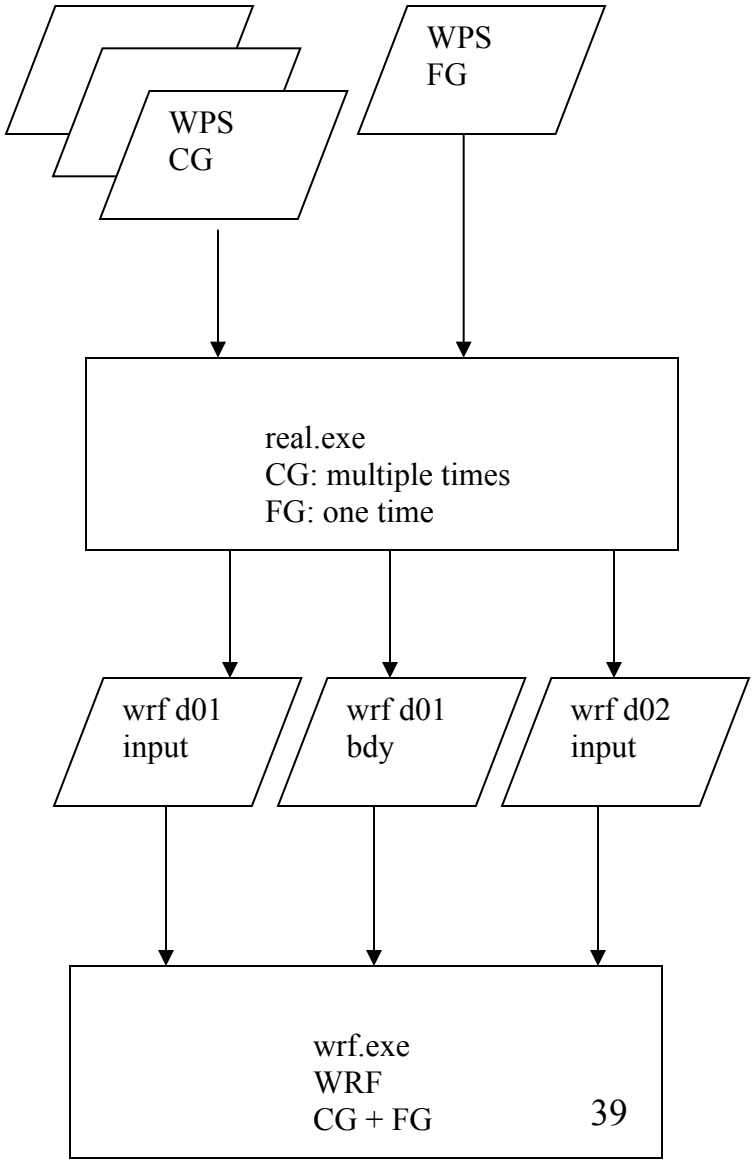
2-Way Nest with 2 Inputs

Coarse and fine grid domains must start at the same time, fine domain may end at any time

Feedback may be shut off to produce a 1-way nest (cell face and cell average)

Any integer ratio for coarse to fine is permitted, odd is usually chosen for real-data cases

Options are available to ingest only the static fields from the fine grid, with the coarse grid data horizontally interpolated to the nest



2-Way Nest with 2 Inputs …cntd

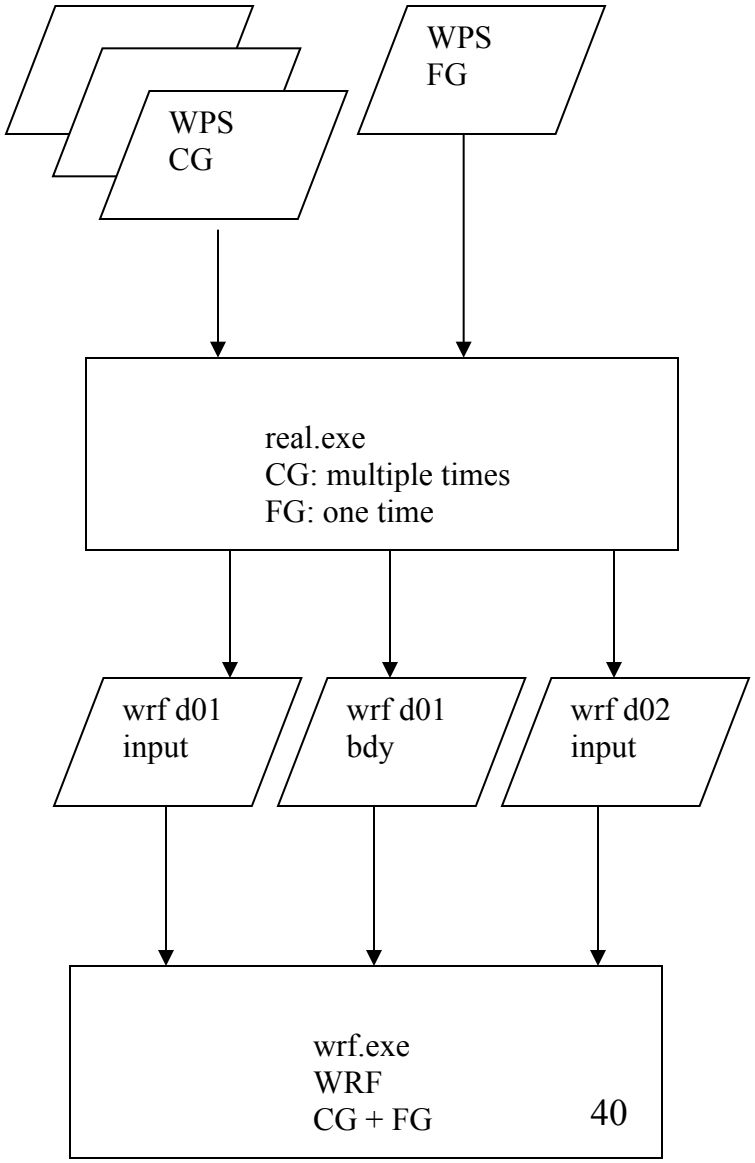
No vertical nesting

Usually the same physics are run on all of the domains (excepting cumulus)

The grid distance ratio is not strictly tied to the time step ratio

Topography smoothly ramps from coarse grid to the fine grid along the interface along the nest boundary

All fine grids must use the nested lateral boundary condition



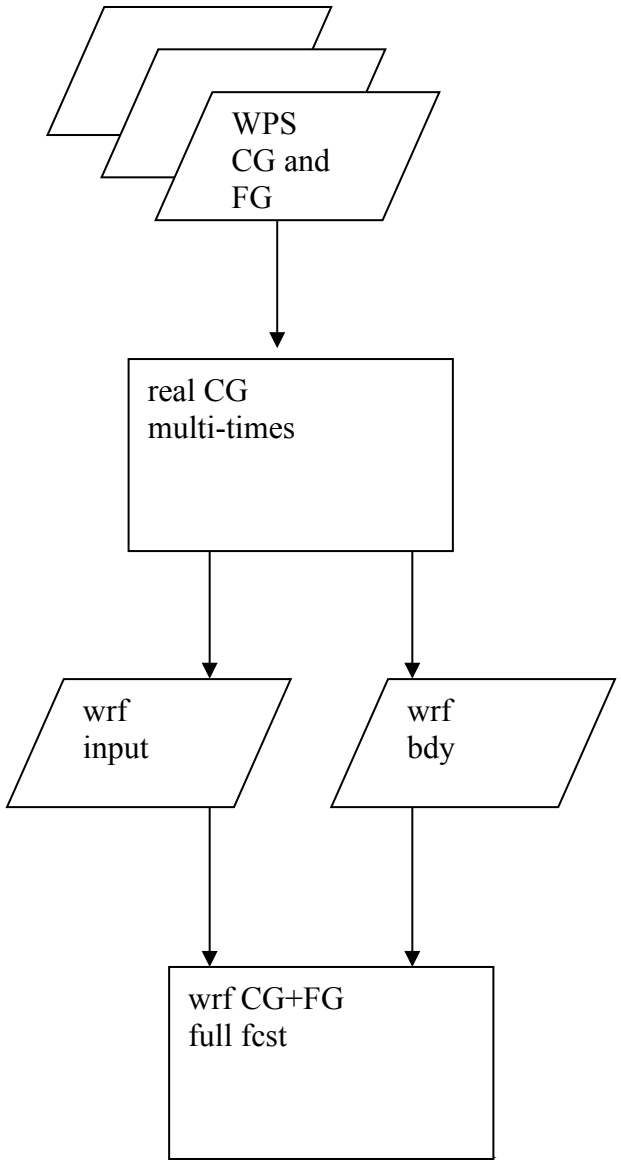
2-Way Nest with 1 Input

A single namelist column entry is tied to each domain

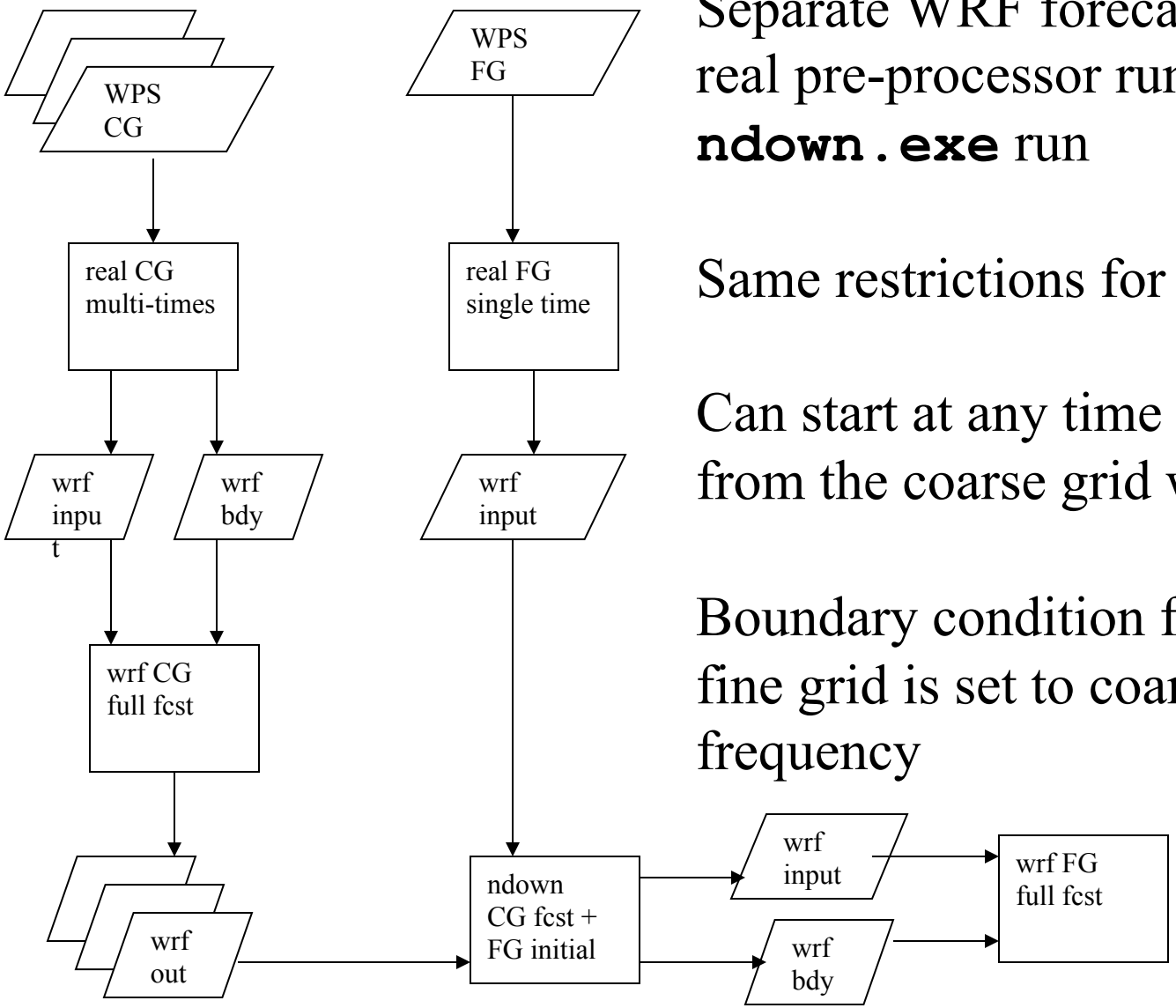
The horizontal interpolation method, feedback, and smoothing are largely controlled through the Registry file

For a 3:1 time step ratio, after the coarse grid is advanced, the lateral boundaries for the fine grid are computed, the fine grid is advanced three time steps, then the fine grid is fed back to the coarse grid (recursively, depth first)

Helpful run*.tar files are located in the **`./WRFV2/test/em_real`** directory



1-Way Nest with 2 Inputs



Separate WRF forecast runs, separate real pre-processor runs, intervening **`ndown.exe`** run

Same restrictions for nest ratios

Can start at any time that an output time from the coarse grid was created

Boundary condition frequency for the fine grid is set to coarse grid output frequency

&time_control

Namelist for nested domains

```
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,
restart            = .true.,
```

***run_** * time variables:*
– Model simulation length: *wrf.exe* and domain 1only

***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).

Time interval between WPS output times, and LBC update frequency

Time interval in minutes when a output is written

Number of history times written to one file

Time interval in minutes when a restart file is written

whether this is a restart run

&time_control

Nest input option: ARW only

input_from_file = .true., .true., .true.,

fine_input_stream = 0, 2, 2,

Specify what fields to use in nest input: they can be all (0), or data specified in I/O stream 2 in Regsitry (2). Useful for a nest starting at a later time.

Whether to produce in *real* and use nest wrfinput files in *wrf*. This is usually the case for real-data runs. For idealized nest runs, set it to .false. .

&domains

```
time_step                = 180
time_step_fract_num      = 0,
time_step_fract_den      = 1,
max_dom                  = 3,
e_we                     = 74,     112,    94,
e_sn                     = 61,     97,     91,
e_vert                   = 28,     28,     28,
num_metgrid_levels       = 21
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,
```

– 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)
– May be divided by output intervals

Namelist for nested domains

grid distances: in meters for ARW

– Pressure value at the model top.
– Constrained by the available data from WPS.
– Default is 5000 Pa

dx = 30000, 10000, 3333.33, **dy = 30000, 10000, 3333.33,** **parent_grid_ratio = 1, 3, 3,** **parent_time_step_ratio = 1,3,3,**	All 4 variables must be specified. *Grid ratio* can be any integer, and *time step ratio* can be different from grid ratio. Grid distance is in meters, even for lat/lon map projection.

Make sure the nest domain parameters match those defined in WPS

Running ARW Nested Cases

- Files available from WPS:

  met_em.d01.<*date*>

  met_em.d02.<*date*> (at least one time) **…**

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

  cd test/*em_real*

  ln -s ../../../WPS/met_em.* .

Running ARW Nested Cases

• Edit **namelist.input** file for runtime options
(set **max_dom >= 2** in **&domains** for a nested run)

• Run the real-data initialization program:
./real.exe, if compiled serially,

or

mpirun -np *N* ./real.exe, for a MPI job
where ***N*** is the number of processors requested

Running ARW Nested Cases

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

wrfinput_d01
wrfinput_d02

Single time level data at model's start time for all domains

wrfbdy_d01

Multiple time-level data at the lateral boundary, and only for domain 1

Moving Nest Case (ARW only)

• The main reason for using this option is to run the model economically.

• Must choose correct compile options when creating **configure.wrf** file

– Choose **preset move**, or **vortex following**

• Other options are controlled by the namelists.

• Can do specified move, and automatic vortex tracking (for tropical cyclone application).

• All nest domains can move.

Moving Nest Case (ARW only)

• Namelists in **&domains**:

num_moves, move_id, move_interval,

move_cd_x, move_cd_y, corral_dist

→ nest can only move one parent-grid-cell at a time.

i.e., **move_cd_x = 1, -1,** or **0**

• Must specify initial nest location

Automatic Moving Case

- Tropical cyclone applications only.
- Works better for well developed storms.
- Namelists in **&domains**:

vortex_interval (default 15 min)

max_vortex_speed (default 40 m/s)

corral_dist (default 8 coarse grid cells)

track_level (default 50000 Pa)

time_to_move (default is 0 h for all nests)

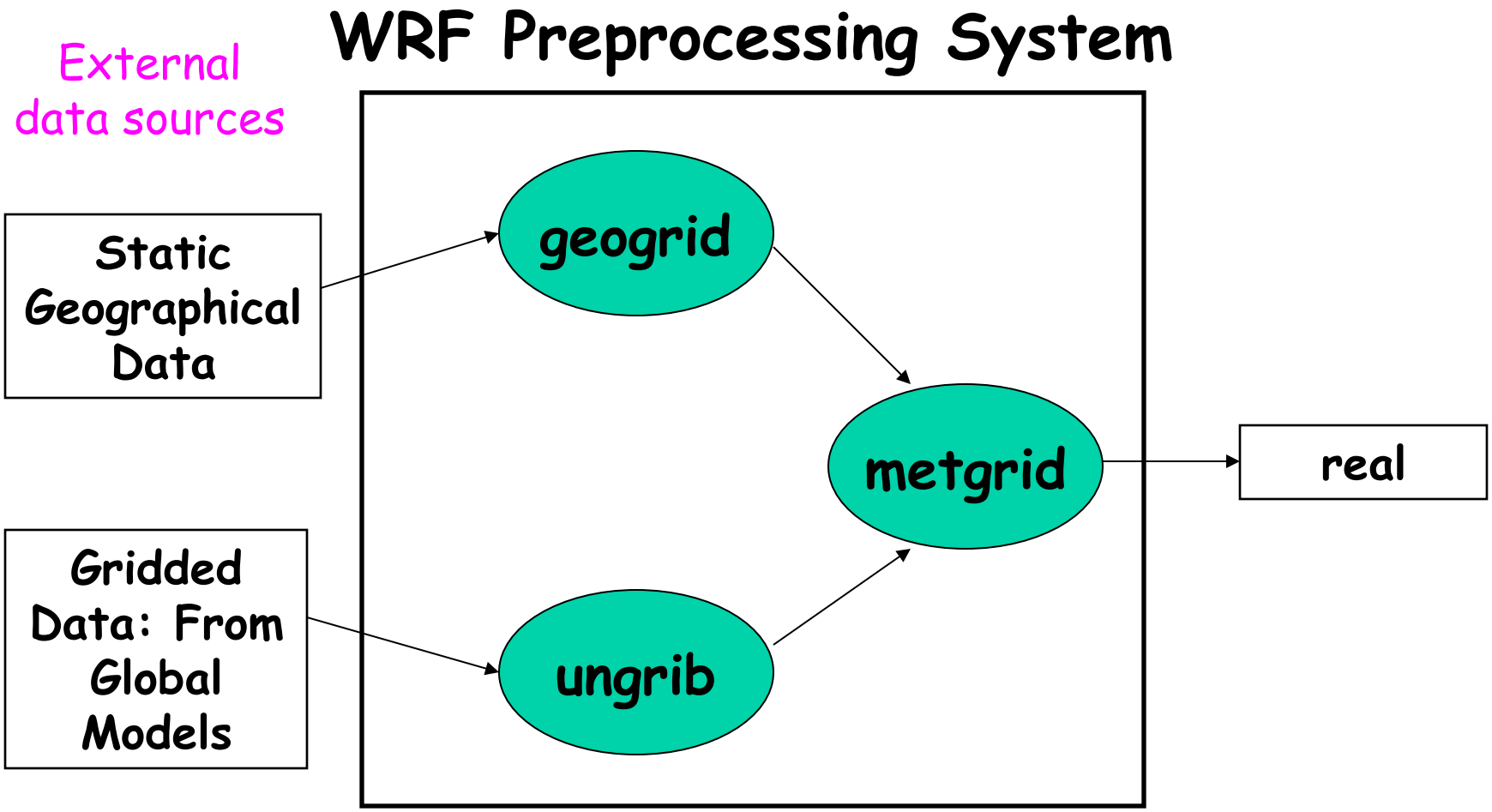
- Must specify initial nest location

The WRF Preprocessing System [WPS]

Outlines

- Installation of WRF preprocessing system (WPS).
- Running of WPS.

Flowchart for WRF Preprocessing System



Required Libraries and compiler

- Libraries:
  - Compulsory:
    - NetCDF (http://www.unidata.ucar.edu/software/netcdf/)
  - Optional (Used for GRIB2 data):
    - Jasper (http://www.ece.uvic.ca/~mdadams/jasper/)
    - PNG (http://www.libpng.org/pub/png/libpng.html)
    - Zlib (http://www.zlib.net/)
- Compiler:
  - Fortran and C compiler

Download static terrestrial data

- The terrestrial fields interpolated by ***geogrid*** may be downloaded from same page as the code:

  http://www.mmm.ucar.edu/wrf/users/download/get_source.html

  - Two options for data: low-res and all resolutions

- Data are static: only need to be downloaded once.
- Extract the data using the command

  tar -zxvf geog.tar.gz

It creates a folder (for ex. geog) with ~10 GB of space (264 MB for low-res only)!

- Data can be shared by users on the same machine by placing files in a common directory
  - Recommended due to size!

Contains of the "geogrid" data

The *geog.tar.gz* file (all resolutions) contains:

- *albedo_ncep* – monthly surface albedo
- *greenfrac* – monthly vegetation fraction
- *islope* – slope index
- *landuse* – land use category (30", 2', 5', and 10' res.)
- *maxsnowalb* – maximum snow albedo (30", 2', 5', and 10' res.)
- *modis_landuse_20class_30s* - MODIS landuse (Noah LSM only)
- *orogwd* – data for gravity wave drag schemes
- *soiltemp* – annual mean deep soil temperature (30", 2', 5', and 10' res.)
- *soiltype_bot* – bottom-layer soil type (30", 2', 5', and 10' res.)
- *soiltype_top* – top-layer soil type (30", 2', 5', and 10' res.)
- *topo* – topography height (30", 2', 5', and 10' res.)

Download WPS source code

- The WPS source code can be obtained from:
  http://www.mmm.ucar.edu/wrf/users/download/get_source.html
- For simplicity, install WPS/ in the same location as WRFV3/
  - After **gunzip** and **untar**, using ***tar –xvzf WPSV***.tar.gz*** , a directory WPS/ will be created.
  - **> cd** WPS
- Set the library paths as follows (in B shell)
  - export NETCDF=/usr/local/netcdf
  - export JASPERINC=/usr/local/jasper/include
  - export JASPERLIB=/usr/local/jasper/lib

Configure WPS

- To conigure WPS for your computer, type:

  ./configure

- This script offers the user choices for Type of compiler, Serial or Distributed memory, GRIB1 or GRIB2, etc.

```
Will use NETCDF in dir: /usr/local/netcdf-pgi
    $JASPERLIB or $JASPERINC not found in environment, configuring to build without
    grib2 I/O...
    ------------------------------------------------------------------------
Please select from among the following supported platforms.

   1.  PC Linux i486 i586 i686, PGI compiler      serial, NO GRIB2
   2.  PC Linux i486 i586 i686, PGI compiler      serial
   3.  PC Linux i486 i586 i686, PGI compiler      DM parallel, NO GRIB2
   4.  PC Linux i486 i586 i686, PGI compiler      DM parallel
   5.  PC Linux i486 i586 i686, Intel compiler    serial, NO GRIB2
   6.  PC Linux i486 i586 i686, Intel compiler    serial
   7.  PC Linux i486 i586 i686, Intel compiler    DM parallel, NO GRIB2
   8.  PC Linux i486 i586 i686, Intel compiler    DM parallel
   9.  PC Linux i486 i586 i686, g95 compiler,     serial, NO GRIB2
  10.  PC Linux i486 i586 i686, g95 compiler,     serial

    Enter selection [1-10] : 1
------------------------------------------------------------
Configuration successful. To build the WPS, type: compile
------------------------------------------------------------
```

- This creates a file called **configure.wps**

Compile WPS

Reminder: A successful compilation of WRF is required prior to WPS compilation!

- If configuration was successful, compile WPS:

  ./compile >& compile_wps.log

- If the compilation is successful, it will create three executables:
  - geogrid.exe: define size/location of domain(s)
  - ungrib.exe: extract meteorological fields from GRIB files
  - metgrid.exe: horizontally interpolate meteorological fields (from ungrib) to simulation grid(s) (defined by geogrid)

- If compilation is successful, it will create the following executables in **util/**:
  - **avg_tsfc.exe**
  - **g1print.exe**
  - **g2print.exe**
  - **mod_levs.exe**
  - **rd_intermediate.exe**
  - **calc_ecmwf_p.exe**

- If NCAR Graphics libraries are available it will also create in **util/**:
  - **plotgrids.exe**
  - **plotfmt.exe**

Note: sharing of WPS installation is also possible.

Directory structure of WPS

```
> ls
drwxr-xr-x 2    4096 arch
-rwxr-xr-x 1    1672 clean
-rwxr-xr-x 1    3510 compile
-rw-r--r-- 1   85973 compile.output
-rwxr-xr-x 1    4257 configure
-rw-r--r-- 1    2486 configure.wps
drwxr-xr-x 4    4096 geogrid
lrwxrwxrwx 1      23 geogrid.exe -> geogrid/src/geogrid.exe
-rwxr-xr-x 1    1328 link_grib.csh
drwxr-xr-x 3    4096 metgrid
lrwxrwxrwx 1      23 metgrid.exe -> metgrid/src/metgrid.exe
-rw-r--r-- 1    1101 namelist.wps
-rw-r--r-- 1    1987 namelist.wps.all_options
-rw-r--r-- 1    1075 namelist.wps.global
-rw-r--r-- 1     652 namelist.wps.nmm
-rw-r--r-- 1    4786 README
drwxr-xr-x 4    4096 ungrib
lrwxrwxrwx 1      21 ungrib.exe -> ungrib/src/ungrib.exe
drwxr-xr-x 3    4096 util
```

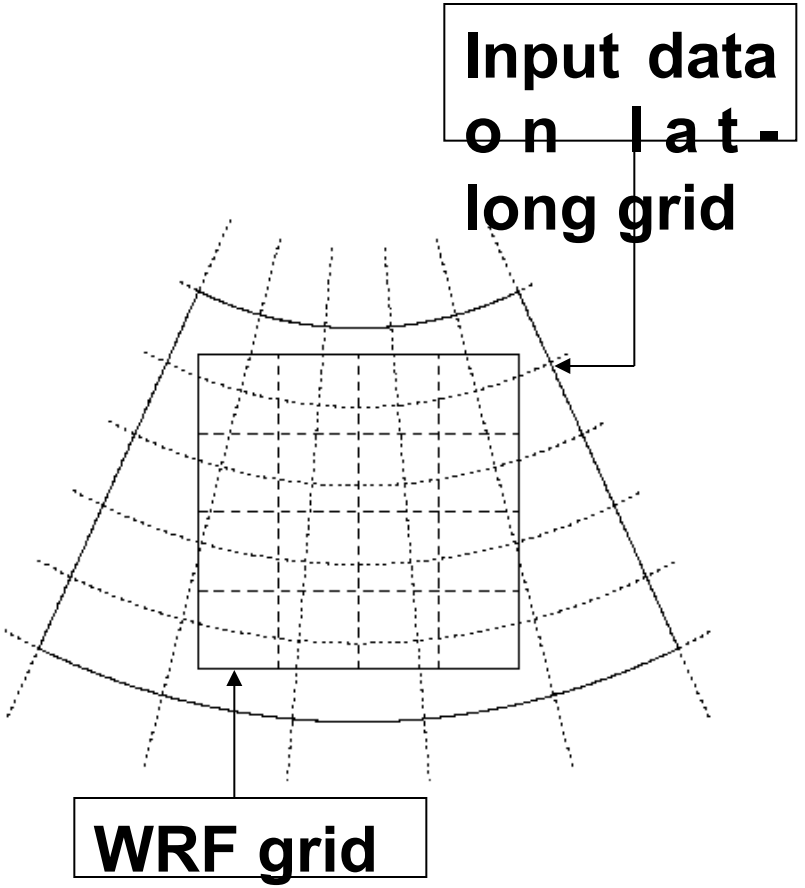
The "geogrid" program

- Geogrid provides values for static (time-invariant) fields at each model grid point
  - Compute latitude, longitude, map scale factor, and Coriolis parameters at each grid point
  - Horizontally interpolate static terrestrial data (e.g., topography height, land use category, soil type, vegetation fraction, monthly surface albedo)

- For WRF model domains, geogrid defines:
  - Map projection (all domains must use the same projection)
  - Geographic location of domains
  - Dimensions of domains

Interpolating the static fields:

- Given definitions of all computational grids, geogrid interpolates terrestrial, time-invariant fields
  - Topography height
  - Land use categories
  - Soil type (top layer & bottom layer)
  - Annual mean soil temperature
  - Monthly vegetation fraction
  - Monthly surface albedo



Interpolation options in geogrid

- 4-point bilinear
- 16-point overlapping parabolic
- 4-point average (simple or weighted)
- 16-point average (simple or weighted)
- Grid cell average
- Nearest neighbor
- Breadth-first search

Why have so many interpolation option?

- Different interpolators work best for different fields and different relative grid resolutions
  - Some interpolators preserve positive definiteness
  - Some interpolators produce "smoother" fields
  - Some interpolators are best suited for discrete or categorical fields
  - Some are good when going from a fine grid to a coarse grid

Program flexibility in geogrid:

- The GEOGRID.TBL file determines
  - Which fields will be produced by geogrid
  - What sources of data will be used
  - How the data will be interpolated/smoothed
  - Any derived fields (e.g., dominant cat., df/dx)

- Acceptable defaults exist in GEOGRID.TBL, so user will not generally need to edit the file

- *geogrid* is flexible enough to ingest and interpolate new static fields
  - handles either continuous or categorical fields

- New data sets must be written to simple binary format

- User needs to add an entry to the file GEOGRID.TBL

Advanced features in "geogrid"

- The GEOGRID.TBL file
  - What is a GEOGRID.TBL file?
  - How to ingest new static fields?

The GEOGRID.TBL file

- GEOGRID.TBL is the file that determines which fields are interpolated by geogrid *at runtime*
  - Each entry in GEOGRID.TBL corresponds to one data source
  - When new data sources are involved, or when the default treatment of fields is inadequate, user may want/need to edit GEOGRID.TBL
  - However, default GEOGRID.TBL is sufficient to initialize a WRF simulation

- Format of GEOGRID.TBL file is simple text, with specifications of the form *keyword=value*

- Example entry for a 30" landuse data set:

```
==========================================
   name=LANDUSEF # India, TX urban data
                priority = 1
                dest_type = categorical
                z_dim_name = land_cat
                interp_option = 30s:nearest_neighbor
                abs_path = 30s:/user2/sujata/India
==========================================
```

Note: A complete set of possible "keywords" are provided in the user guide.

- Using the GEOGRID.TBL, we can
  - Change the method(s) used to interpolate a field
  - Apply smoothing filters to continuous fields
  - Derive fields from others
    - e.g., dominant category or slope fields
  - *Add new data for geogrid to interpolate*

- There are three basic types of new data to be added through the GEOGRID.TBL file:

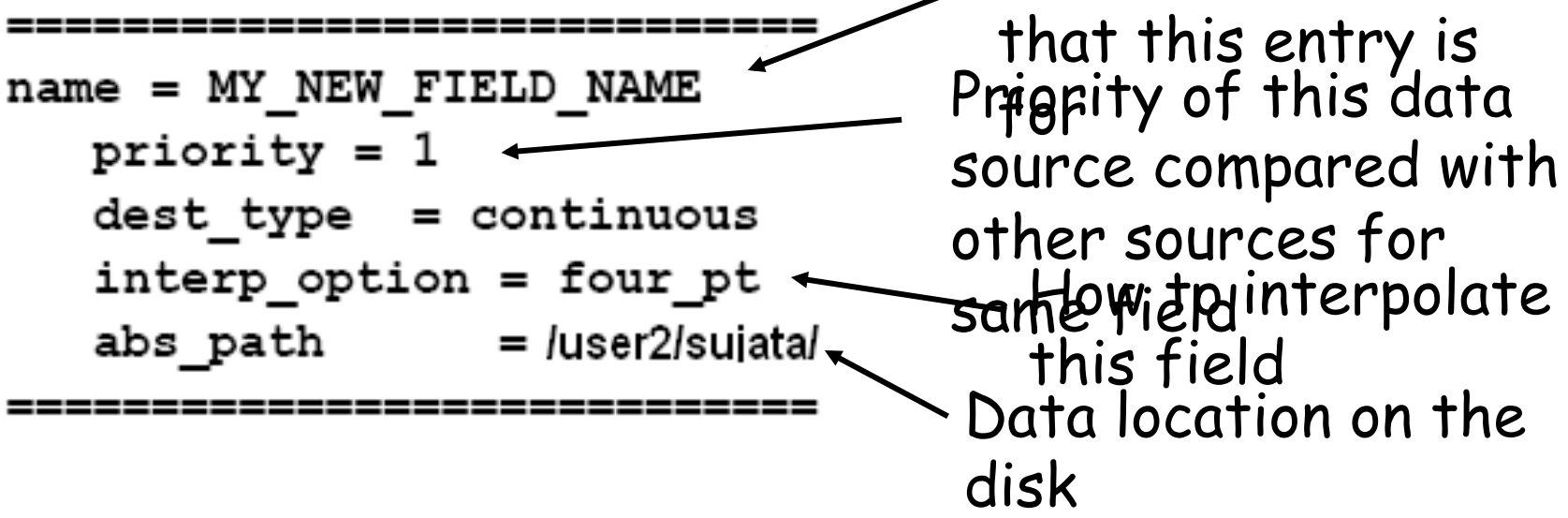
  1) Completely new fields
     - fields that were previously not processed by geogrid

  2) Different resolution data sets for an existing field
     - Such sources *do not need to be supplemented* by existing data
     - E.g., Adding a 90-meter resolution topography data set

  3) Alternative sources for a field that *must be used in addition to an existing source*
     - E.g., A new soil category data set exists, but covers only Iberian Peninsula

1) Completely New Fields

- For a new field, simply add an entry in GEOGRID.TBL file



2) Different resolution data set

For different resolution data sets for an existing field, specify the path to the new data set and which interpolation method should be used for the new resolution in the existing entry for that field.

```
==================================
name = HGT_M
     priority = 1
     dest_type = continuous
     smooth_option = smth-desmth
     interp_option =          30s:special(4.0)+four_pt
     interp_option =      my_res:four_pt
     interp_option =     default:four_pt
     rel_path=       30s:topo_30s/
     rel_path=   my_res:new_topo_directory/
     rel_path=  default:topo_2m/
 ==================================
```

3) Alternative data sources

- Add a new entry for the field that has the same name as the field's existing entry, but make priority of new entry higher:

```
==================================
name = HGT_M
       priority = 2
       dest_type = continuous
       interp_option = default:four_pt
       rel_path      = default:some_path/
==================================
name = HGT_M
      priority = 1
      dest_type = continuous
      interp_option = default:four_pt
       rel_path      = default:topo_2m/
  ==================================
```

Map Projections

- A **map projection** is any method of representing the surface of a sphere or other shape on a plane which is necessary for creating maps.

- Importance of map projection:
  - The real earth is (roughly) an ellipsoid
  - But WRF computational domains are defined by rectangles in the plane

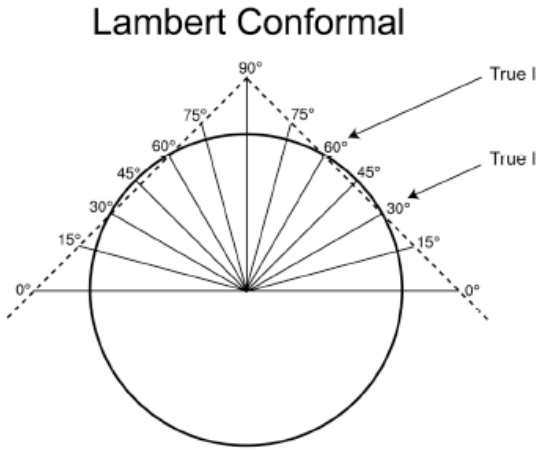
- ARW supports four types of map projections:
  - Lambert conformal
  - Mercator
  - Polar stereographic
  - Latitude-longitude

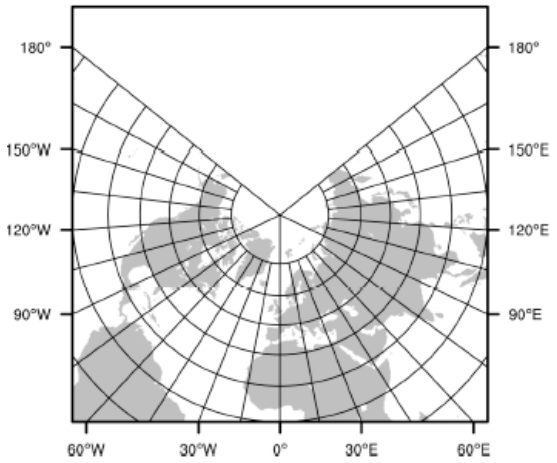
Descriptions of different "Map Projections"

- **Lambert conformal**

  The **Lambert conformal** is a conic map projection, which is often used for aeronautical charts. The projection superimposes a cone over the sphere of the Earth, with two reference parallels secant to the globe and intersecting it.

  - Well-suited for mid-latitudes
  - Domain cannot contain either pole
  - Domain cannot be periodic in west-east direction
  - Either one or two *true latitudes* may be specified
  - If two are given, the order doesn't matter

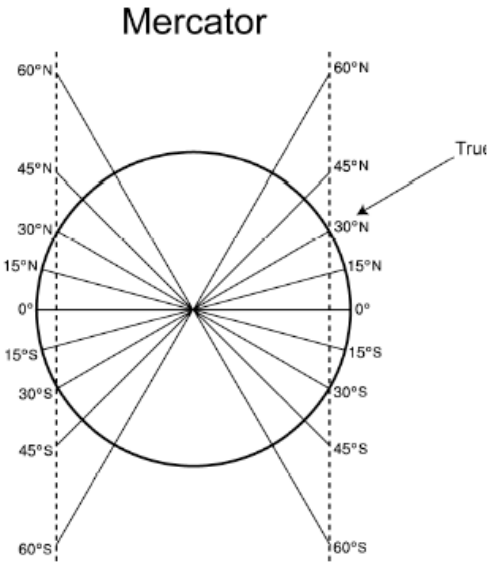


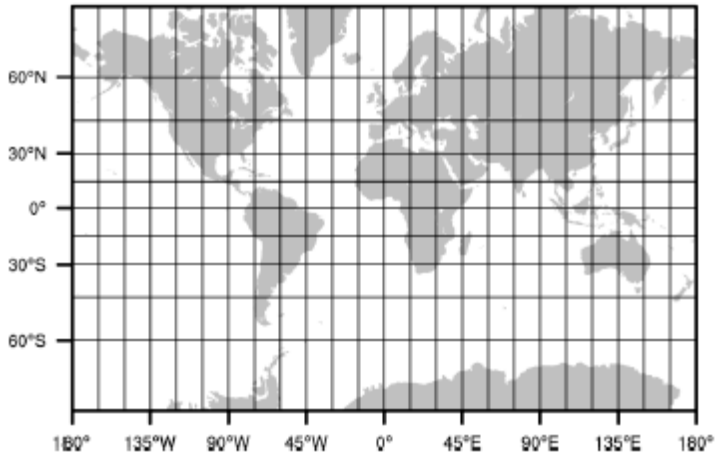


- **Mercator**

The **Mercator projection** is a cylindrical map projection. It is the standard map projection for nautical purposes because of its ability to represent lines of constant course, known as rhumb lines, as straight segments

  - Well-suited for low-latitudes
  - May be used for "channel" domain (periodic domain in west-east direction)
  - A single true latitude is specified
    - Cylinder intersects the earth's surface at +/- truelat

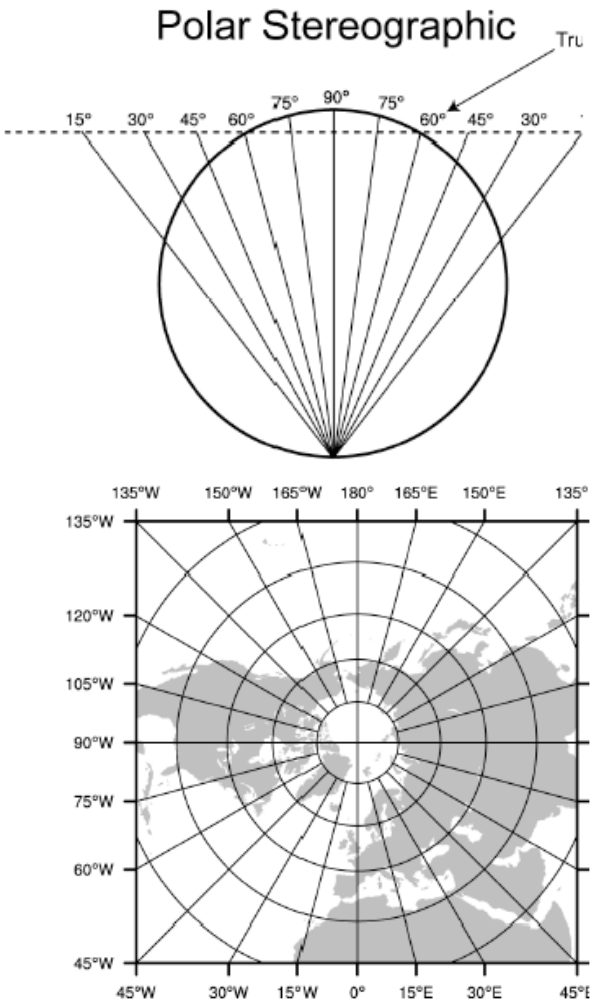




- **Polar Stereographic**

  The polar stereographic projection specifies a projection plane or grid tangent to the Earth's surface at 70 degrees northern and southern latitude. This planar grid is designed so that the grid cells at 70 degrees latitude are exactly the nominal grid resolution.

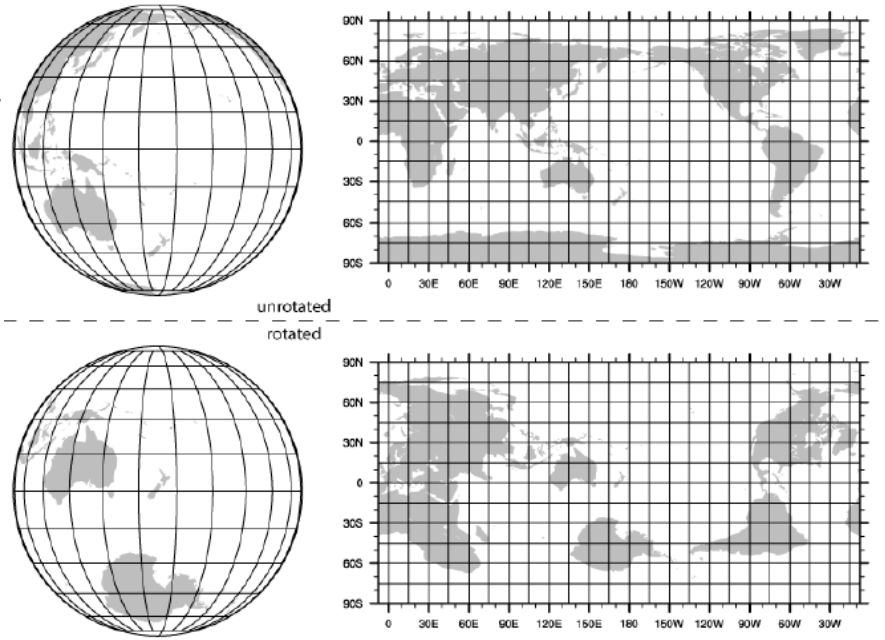
  - Good for high-latitude domains, especially if domain must contain a pole
  - A single true latitude is specified



- **Latitude-Longitude**

  It is the equidistance map projection and can be used for any lat-long location.

  - Can be used for regional and global domains
  - Can be used in its normal or rotated aspect



namelist.wps

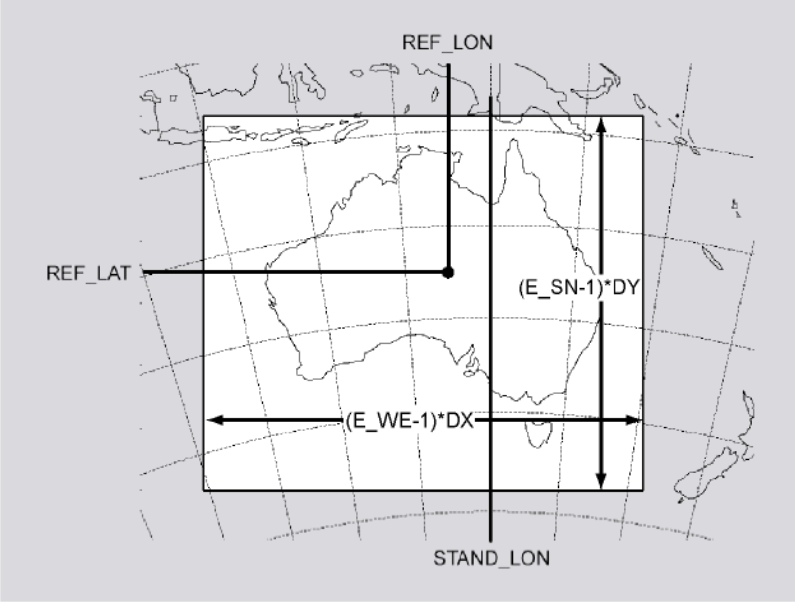
```
&share
 wrf_core = 'ARW',
 max_dom = 2,
 start_date = '2008-03-24_12:00:00','2008-03-24_12:00:00',
 end_date   = '2008-03-24_18:00:00','2008-03-24_12:00:00',
 interval_seconds = 21600,
 io_form_geogrid = 2
/

&geogrid
 parent_id         =   1,   1,
 parent_grid_ratio =   1,   3,
 i_parent_start    =   1,  31,
 j_parent_start    =   1,  17,
 s_we              =   1,   1,
 e_we              =  74, 112,
 s_sn              =   1,   1,
 e_sn              =  61,  97,
 geog_data_res     = '10m','2m',
 dx = 30000,
 dy = 30000,
 map_proj = 'lambert',
 ref_lat   = 34.83,
 ref_lon   = -81.03,
 truelat1  =  30.0,
 truelat2  =  60.0,
 stand_lon = -98.,
 geog_data_path = '/mmm/users/wrfhelp/WPS_GEOG/'
/
&ungrib
 out_format = 'WPS',
 prefix     = 'FILE'
/
&metgrid
 fg_name                         = 'FILE',
 io_form_metgrid                 = 2,
/
```

Defining the model domain:

- Define projection of domains using a subset of the following parameters
  - MAP_PROJ: 'lambert', 'mercator', 'polar', 'lat-lon', or 'rotated_ll'
  - TRUELAT1: First true latitude
  - TRUELAT2: Second true latitude (only for Lambert conformal)
  - POLE_LAT, POLE_LON: Location of North Pole in WRF computational grid (only for 'lat-lon')
  - STAND_LON: The meridian parallel to y-axis
- All parameters reside in the file namelist.wps

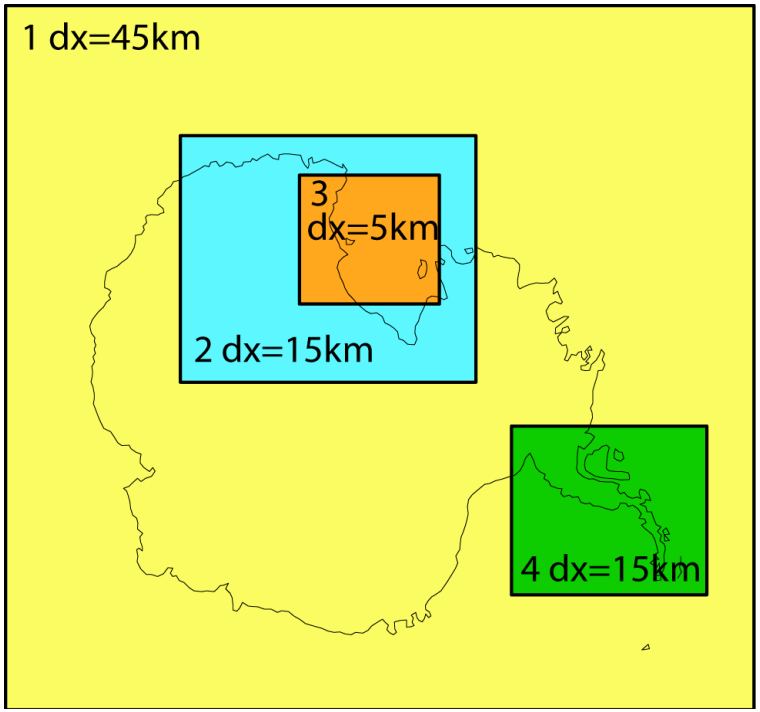
Defining the model domain:

- **Define the area covered (dimensions and location) for coarse domain using the following:**
  - REF_LAT, REF_LON: The (lat,lon) location of a known location in the domain (by default, the center point of the domain)
  - DX, DY: Grid distance where map factor = 1
    - For Lambert, Mercator, and polar stereographic: meters
    - For (rotated) latitude-longitude: degrees
  - E_WE: Number of velocity points in west-east direction for ARW;
  - E_SN: Number of velocity points in south-north direction for ARW;



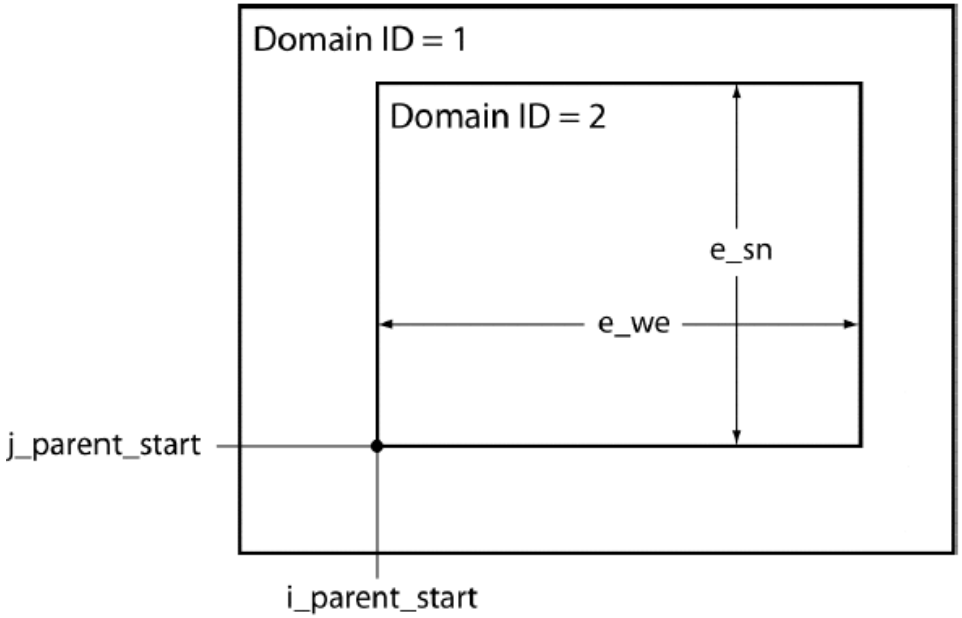
Nesting in "geogrid"

- A nested domain is a domain that is wholly contained within its parent domain and that receives information from its parent, and that may also feed information back to its parent
  - A nested domain has exactly one parent
  - A domain may have one or more children
- 2-way nests on the same nesting level must not overlap in coverage!



Defining Nesting Domain

- Define the dimensions and location of nested domains using:
  - PARENT_ID: Which domain is the parent?
  - PARENT_GRID_RATIO: What is the ratio of grid spacing in parent to grid spacing in this nest?
  - I_PARENT_START: *i*-coordinate in parent of this nest's lower-left corner
  - J_PARENT_START: *j*-coordinate in parent of this nest's lower-left corner
  - E_WE: Number of velocity points in west-east direction
  - E_SN: Number of velocity points in south-north direction



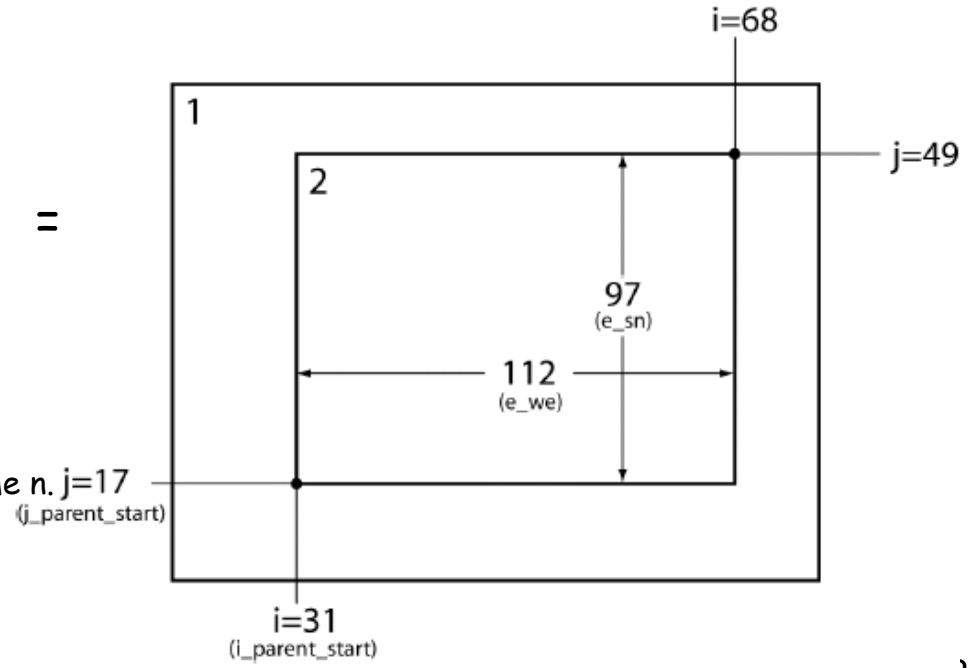
The grid spacing of domain 2 is determined by grid spacing of domain 1 and the parent_grid_ratio

Example:

Assuming parent_grid_ratio = 3

Nest dimension:

n* parent_grid_ratio+1, for some n.



Running "geogrid"

- Step-1: Edit namelist.wps (for geogrid, only the &share and &geogrid namelists need to be edited in namelist.wps)

```
&share
 wrf_core = 'ARW',
 max_dom = 2,
 io_form_geogrid = 2,
/
```

Which WRF core

Total model domains

Format for geogrid output files:

2: NetCDF

```
&geogrid
 parent_id         =   1,    1,
 parent_grid_ratio =   1,    3,
 i_parent_start    =   1,   20,
 j_parent_start    =   1,   17,
 e_we              =  220,  181,
 e_sn              =  175,  181,
 geog_data_res     = '5m',  '2m',
 dx                = 15000,
 dy                = 15000,
 map_proj          = 'lambert',
 ref_lat           = 37.0,
 ref_lon           = -97.0,
 truelat1          = 45.0,
 truelat2          = 30.0,
 stand_lon         = -97.0,
 geog_data_path = '/data/static/geog/'
/
```

Nesting

Domain sizes

Static data resolution

Grid space or resolution in meters

Map projection

Static data directory

Running geogrid

- **Step-2:** Make sure GEOGRID.TBL is linked to the correct version of GEOGRID.TBL
  - There are multiple GEOGRID.TBL files to support multiple dynamical cores in WRF
  - GEOGRID.TBL.ARW must be used for ARW

```
> ls geogrid/GEOGRID.TBL
GEOGRID.TBL -> GEOGRID.TBL.ARW
```

Running geogrid

- Step-3: Run geogrid.exe

```
Parsed 11 entries in GEOGRID.TBL
Processing domain 1 of 2
  Processing XLAT and XLONG
  Processing MAPFAC
  Processing F and E
  Processing ROTANG
  Processing LANDUSEF
  Calculating landmask from LANDUSEF
  Processing HGT_M
  ...
```

Geogrid processes each domain individually. There will be one section of messages of each domain.

As each field is processed, a message will be written to the screen and to the geogrid.log file.

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!  Successful completion of geogrid.                !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

If the geogrid ran successfully, this message will be printed. If there is an error, then check the error in "geogrid.log" file.

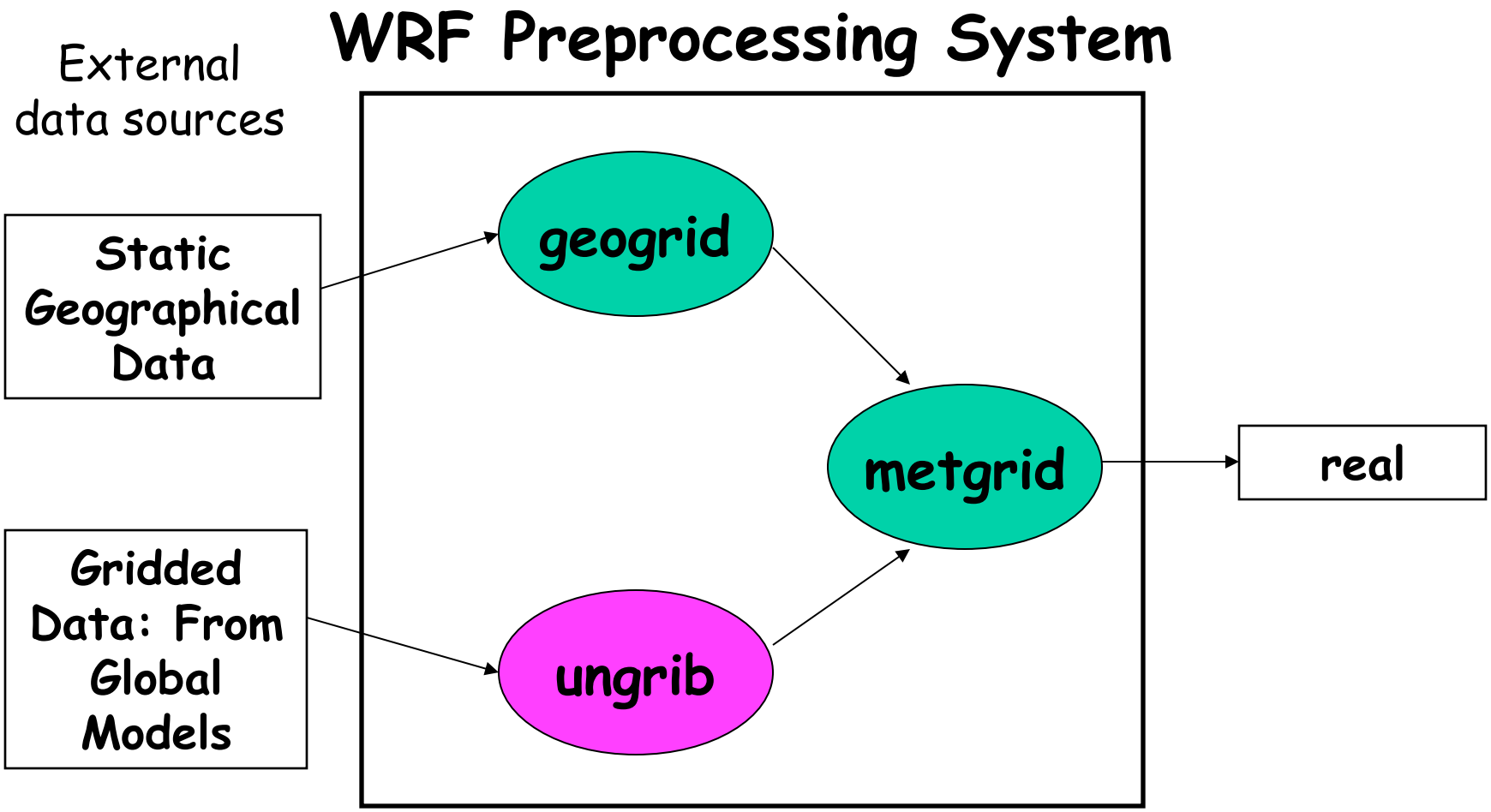
Output from geogrid:

- The parameters defining each domain, plus interpolated static fields, are written using the WRF I/O API
  - One file per domain for ARW

- Filenames: geo_em.d0*n*.nc (where *n* is the domain ID)

- Example:
  - geo_em.d01.nc
  - geo_em.d02.nc (nest)
  - geo_em.d03.nc (nest)

Ungrib.exe



ungrib: think un+grib

GRIB file:

- GRIB is a WMO standard file format for storing regularly-distributed (e.g., gridded) fields
  - "General Regularly-distributed Information in Binary"
- Fields within a GRIB file are compressed with a glossy compression
  - Think of truncating numbers to a fixed number of digits
- A record-based format
- Fields in a file are identified only by code numbers
  - These numbers must be referenced against an external table to determine the corresponding field

The "ungrib" program

- Read GRIB Edition 1 and GRIB Edition 2 files
- Extract meteorological fields using Vtables (Variable tables)
  - Vtables are files that give the GRIB codes for fields to be extracted from GRIB input files
  - One Vtable for each source of data
  - Vtables are provided for: GFS, NAM, AGRMET, NARR, ECMWF and others
- If necessary, derive required fields from related ones
  - E.g., Compute RH from T, P, and Q
- Write requested fields to an intermediate file format

Example of a GRIB1 Vtable:

GRIB1 Param	Level Type	From Level1	To Level2	UNGRIB Name	UNGRIB Units	UNGRIB Description
11	100	*		T	K	Temperature
33	100	*		U	m s-1	U
34	100	*		V	m s-1	V
52	100	*		RH	%	Relative Humidity
7	100	*		HGT	m	Height

1:0:D=2008042800:HGT:1000 mb:kpds=7,100,1000:anl:winds are N/S:"Geopotential height [gpm]
2:114114:D=2008042800:HGT:975 mb:kpds=7,100,975:anl:winds are N/S:"Geopotential height [gpm]

27:2918084:D=2008042800:TMP:1000 mb:kpds=11,100,1000:anl:winds are N/S:"Temp. [K]
28:2999618:D=2008042800:TMP:1000 mb:kpds=11,100,975:anl:winds are N/S:"Temp. [K]

144	112	100	200	SM100200	kg m-3	Soil Moist 100-200 cm below gr lay
85	112	0	10	ST000010	K	T 0-10 cm below ground layer (Uppe
85	112	10	40	ST010040	K	T 10-40 cm below ground layer (Upp
85	112	40	100	ST040100	K	T 40-100 cm below ground layer (Up
85						layer (E
91						
81						a in GRI
7						alysis
11						for SST
65						h
223						
224						y
225						

Level	Level Type	From Level1	To Level2
Upper-air	100	*	(blank)
Surface	1	0	(blank)
Sea-level	102	0	(blank)
Levels at a specified height AGL	105	Height, in meters, of the level above ground	(blank)
Fields given as layers	112	Starting level for the layer	Ending level for the layer

GRIB2 data entries in Vtable

metgrid Description	GRIB2 Discp	GRIB2 Catgy	GRIB2 Param	GRIB2 Level
Temperature	0	0	0	100
U	0	2	2	100
V	0	2	3	100
Relative Humidity	0	1	1	100
Height	0	3	5	100
Temperature at 2 m	0	0	0	103
Relative Humidity at 2 m	0	1	1	103
U at 10 m	0	2	2	103
V at 10 m	0	2	3	103
Surface Pressure	0	3	0	1
Sea-level Pressure	0	3	1	101
Soil Moist 0-10 cm below grn layer (Up)	2	0	192	106
Soil Moist 10-40 cm below grn layer	2	0	192	106
Soil Moist 40-100 cm below grn layer	2	0	192	106
Soil Moist 100-200 cm below gr layer	2	0	192	106
Soil Moist 10-200 cm below gr layer	2	0	192	106
T 0-10 cm below ground layer (Upper)	0	0	0	106
T 10-40 cm below ground layer (Upper)	0	0	0	106
T 40-100 cm below ground layer (Upper)	0	0	0	106
T 100-200 cm below ground layer (Bottom)	0	0	0	106
T 10-200 cm below ground layer (Bottom)	0	0	0	106
Ice flag	0	2	0	1
Land/Sea flag (1=land, 0 or 2=sea)	2	0	0	1
Terrain field of source analysis	2	0	7	1
Skin temperature (can use for SST also)	0	0	0	1
Water equivalent snow depth	0	1	13	1
Dominant soil type cat.(not in GFS file)	2	3	0	1
Dominant land use cat. (not in GFS file)	2	0	198	1

- What if a data source has no existing Vtable?
  - Create a Vtable
  - Get a listing of GRIB codes for fields in the source
  - Check documentation from originating center or use utility such as *wgrib, g1print, wgrib2, g2print*
  - Use existing Vtable as a template

- Intermediate file format:
  - After extracting fields listed in Vtable, ungrib writes those fields to intermediate format
  - For meteorological data sets not in GRIB format, the user may write to intermediate format directly
    - *Allows WPS to ingest new data sources*; basic programming required of user
    - Simple intermediate file format is easily read/written using routines from WPS (read_met_module.F and write_met_module.F)

Running ungrib

- Step-1: Edit namelist.wps (for ungrib, only the &share and &ungrib namelists need to be edited)

```
&share
   wrf_core = 'ARW',
   max_dom = 2,
   start_date  = '2006-04-01_00:00:00',
   end_date   = '2006-04-01_12:00:00',
   interval_seconds = 21600
   io_form_geogrid = 2,
/
```

Data frequency

Data time range

```
&ungrib
   out_format = 'WPS',
   prefix = 'GFS',
/
```

Intermediate file format

Intermediate file names

- Step-2: Link the correct Vtable to the file name Vtable

  > ln –s ungrib/Variable_Tables/Vtable.GFS Vtable

  > ls Vtable

  Vtable ->ungrib/Variable_Table/Vtable.GFS

Running ungrib

- Step-3: Link GRIB files to the correct file names in the run directory.
  - ungrib always expects the names of the GRIB files like GRIBFILE.AAA, GRIBFILE.AAB, GRIBFILE.AAC…… using "link_grib.csh" script.

> link_grib.csh <GFS file paths>

- Step-4: Running ungrib.exe

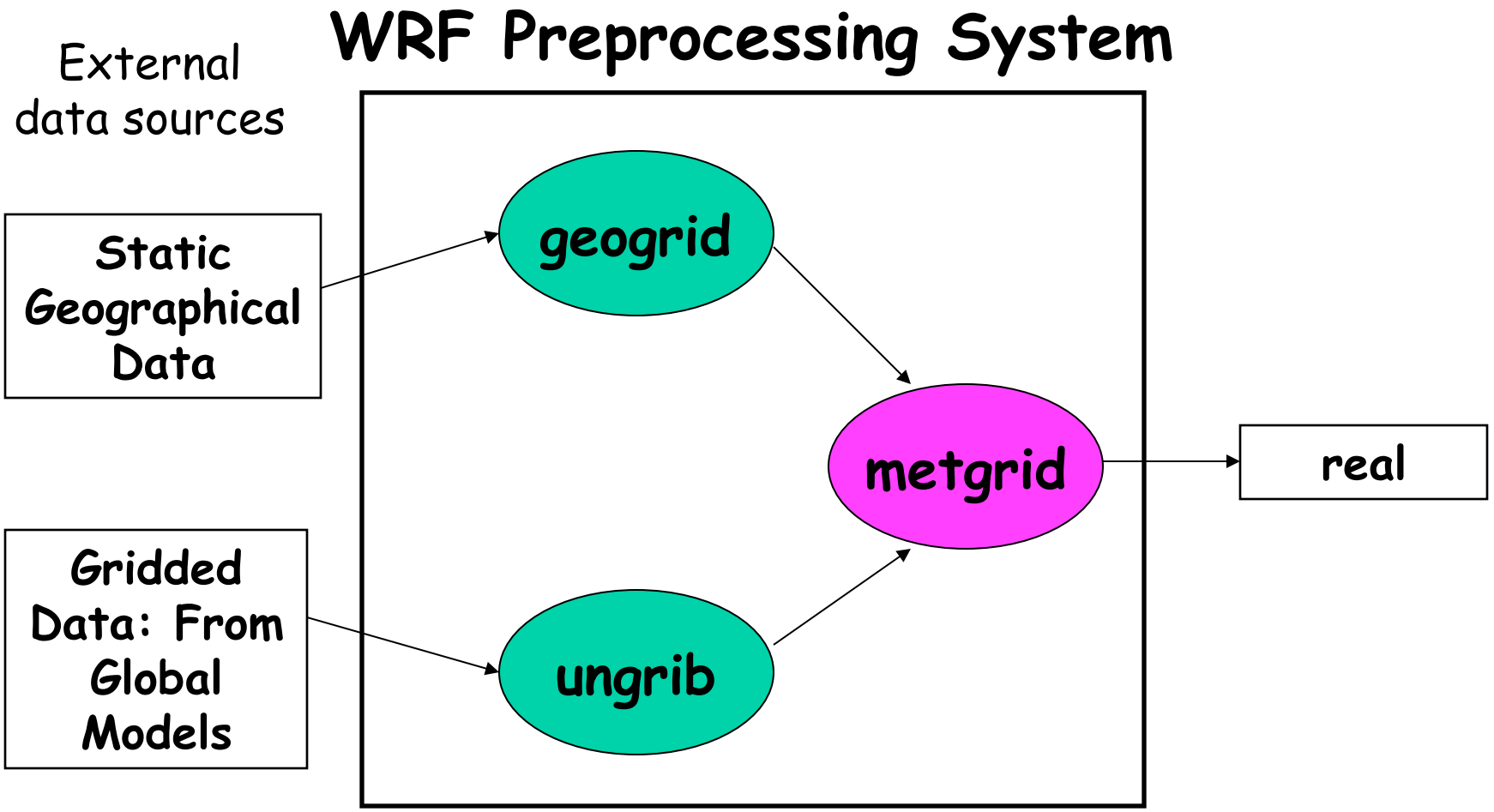
```
 *** Starting program ungrib.exe ***
Start_date =  2006-08-16_12:00:00 ,      End_date = 2006-08-16_12:00:00
output format is WPS
Path to intermediate files is ./
 ungrib - grib edition num            2

################################################################
Inventory for date = 2006-08-16 12:00:00

PRES    TT       UU       VV       RH       HGT
----------------------------------------------------------------
2013.0  o        o        o        o        o        o
2001.0  x        x        x        x        o        x
1000.0  x        x        x        x        x
 975.0  x        x        x        x        x
 950.0  x        x        x        x        x
 925.0  x        x        x        x        x
 900.0  x        x        x        x        x
  .........

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!  Successful completion of ungrib.          !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Output of ungrib program

- Output files named *FILE:YYYY-MM-DD_HH*
  - *YYYY* is year of data in the file; *MM* is month; *DD* is day; *HH* is hour
  - All times are UTC

- Example:
  - FILE:2010-11-24_00
  - FILE:2010-11-24_06
  - FILE:2010-11-24_12

Flowchart for WRF Preprocessing System



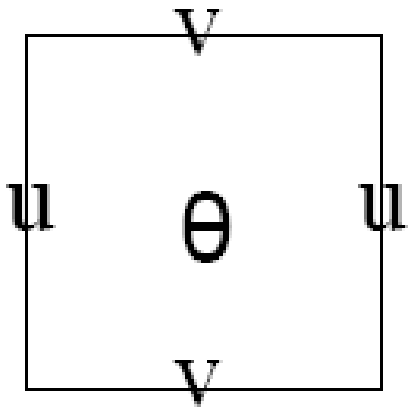
metgrid: think meteorological

The "metgrid" program

- Horizontally interpolate meteorological data (*extracted by ungrib*) to simulation domains (*defined by geogrid*)
  - Masked interpolation for masked fields
- Rotate winds to WRF grid
  - i.e., rotate so that U-component is parallel to *x*-axis, V-component is parallel to *y*-axis

ARW grid staggering

- For ARW, wind U-component interpolated to "u" staggering
- Wind V-component interpolated to "v" staggering
- Other meteorological fields interpolated to "θ" staggering by default (*can change this!*)



A single ARW grid-cell with **u**, **v** and **θ** points.

- Interpolation in "metgrid":
  - 4-point bilinear
  - 16-point overlapping parabolic
  - 4-point average (simple or weighted)
  - 16-point average (simple or weighted)
  - Grid cell average
  - Nearest neighbor
  - Breadth-first search

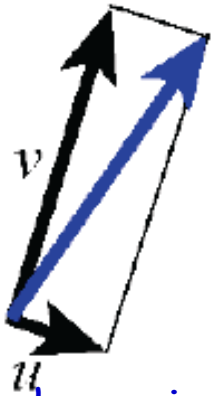
- Masked interpolation:
  - *Masked fields* may only have valid data at a subset of grid points e.g., SST field only valid on water points.

  - When metgrid interpolates masked fields, it must know which points are invalid (masked)
    - Can use separate mask field (e.g., LANDSEA)
    - Can rely on special values (e.g., 1×1030) in field itself to identify masked grid points

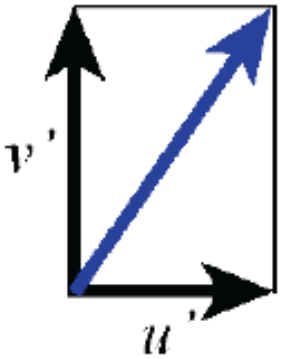
Wind rotation in "metgrid"

- Input wind fields (U-component + V-component) are either:
  - **Earth-relative**: U-component = westerly component; Vcomponent = southerly component
  - **Relative to source grid**: U-component (V-component) parallel to source model x-axis (y-axis)
- WRF expects wind components to be relative to the simulation grid

Example:



A wind vector, shown in terms of its U and V components with respect to the source grid.



The same vector, in terms of its U and V components with respect to the WRF simulation grid.

This process may require two interactions: one from source grid to earth grid and a second from the earth grid to WRF grid.

- The "constant field" in metgrid:
  - For short simulations, some fields may be constant e.g., SST or sea-ice fraction
  - Use namelist option CONSTANTS_NAME option to specify such fields: CONSTANTS_NAME = 'SST_FILE:2010-11-24_00'
- Flexibility in metgrid program:
  - User may specify interpolation methods and related options in the METGRID.TBL file and it is similar to the file GEOGRID.TBL
- Output from metgrid program:
  - For coarse domain, one file per time period, also get the first time period for all nested grids
  - Files contain static fields from geogrid plus interpolated meteorological fields with filenames like:

    met_em.d0*n*.*YYYY-MM-DD_HH:mm:ss*.nc (where *n* is the domain ID #)

Advanced features in "metgrid"

- The METGRID.TBL file
  - What is a METGRID.TBL file?
  - How to define the interpolation option for a new field?

The METGRID.TBL file

- The METGRID.TBL file controls how meteorological fields are interpolated:
  - Unlike GEOGRID.TBL, METGRID.TBL *does not determine which fields will be processed*, only *how to process them* if they are encountered
  - Every field in intermediate files will be interpolated
    - If no entry in METGRID.TBL for a field, a default interpolation scheme (nearest neighbor) will be used
    - It is possible to specify in METGRID.TBL that a field should be discarded
  - Suitable entries in METGRID.TBL are provided for common fields
    - *Thus, many users will rarely need to edit METGRID.TBL*
  - When necessary, different interpolation methods (and other options) can be set in METGRID.TBL
    - Interpolation options can depend on the source of a field

Running metgrid

- Step-1: Edit namelist.wps (for metgrid, only &share and &metgrid namelists need to be edited)

```
&share
   wrf_core = 'ARW',
   max_dom = 2,
   start_date  = '2006-04-01_00:00:00',  '2006-04-01_00:00:00',
   end_date    = '2006-04-01_12:00:00',  '2006-04-01_00:00:00',
   interval_seconds = 21600
   io_form_geogrid = 2,
/
```

Data time ranges for different domains

```
&metgrid
   fg_name = 'GFS',
   constants_name = 'SST:2006-04-01_00',
   io_form_metgrid = 2,
/
```

Intermediate file format

Optional

Metgrid I/O format

- Step-2: Makesure the METGRID.TBL is linked to the correct file i.e. METGRID.TBL.ARW

Running metgrid

- Step-3: Running metgrid.exe

```
 Processing domain 1 of 2
     SST:2006-04-01_00

  Processing 2006-04-01_00
     GFS
  Processing 2006-04-01_06
     GFS
  Processing 2006-04-01_12
     GFS

 Processing domain 2 of 2
     SST:2006-04-01_00
  Processing 2006-04-01_00
     GFS
......
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!  Successful completion of metgrid.  !
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Fields from the constant file (given using constants_name) are processed before any varying field.

Metgrid processes for all time period of one domain before going to the next domain.

After successful completion of the metgrid process.

- Step-4: Check the successful run of metgrid program from metgrid.log file and also the output files (format: met_em.d01.YYYY-MM-DD_HH.nc).

WPS utility programs

- Besides geogrid, ungrib, and metgrid, some simple utility programs are distributed with WPS:
  - For checking contents of intermediate format files
  - For listing contents of GRIB1 & GRIB2 files
  - To assist in locating domains
  - For computing 3d pressure field for ECMWF data
- Some programs use NCAR Graphics libraries for plotting
  - For these utilities, *NCAR Graphics must be installed*
- All utilities are found in the **WPS/util** directory

WPS utilities

- plotgrids:
  - plotgrids can be used to iteratively refine the locations of grids.
  - plotgrids uses the namelist.wps file only, so there is no need to run geogrid first!
- rd_intermediate:
  - The rd_intermediate lists information about the fields found in an intermediate-format file
- plotfmt:
  - The plotfmt program plots the fields in the ungrib intermediate-formatted files
- g1print and g2print:
  - The g1print and g2print programs list the contents of a GRIB1 or GRIB2 file
- calc_ecmwf_p:
  - The calc_ecmwf_p utility creates intermediate files with a pressure (and possibly GHT and RH) field

Thanks