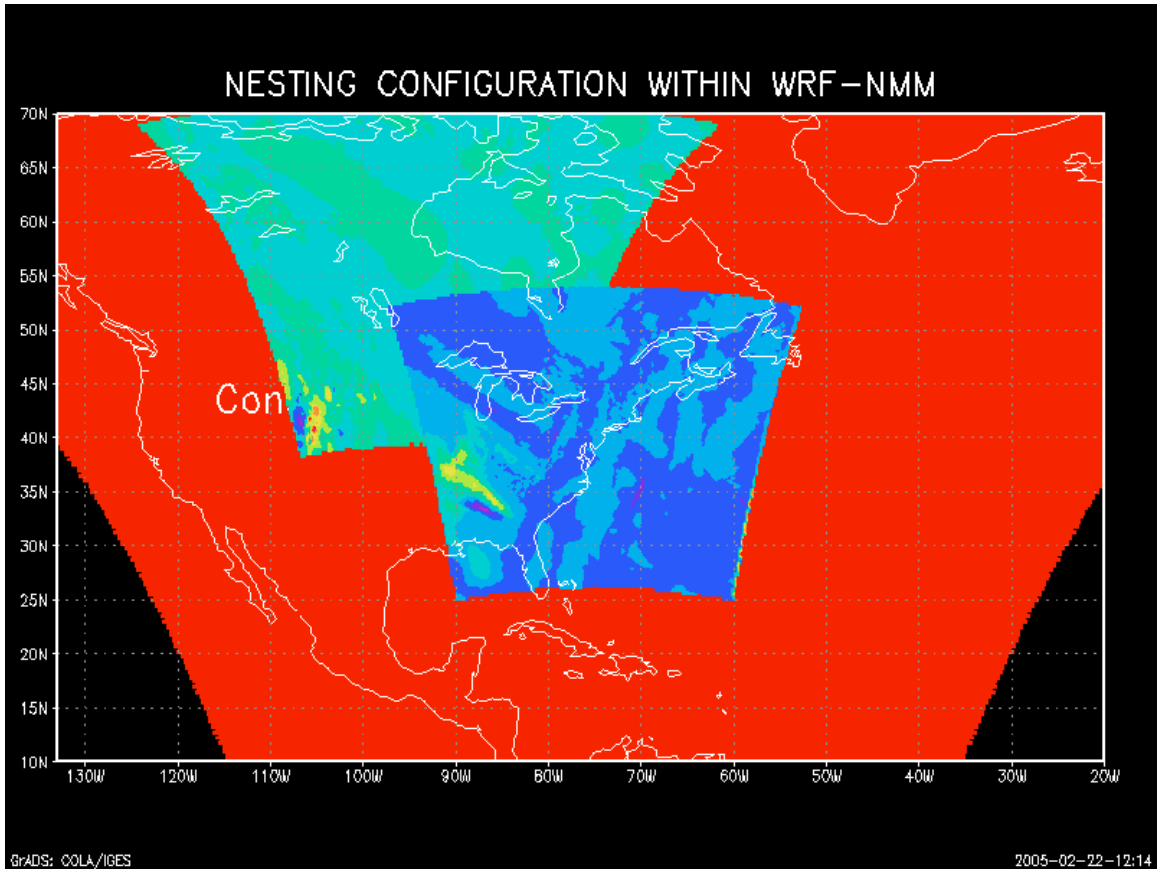
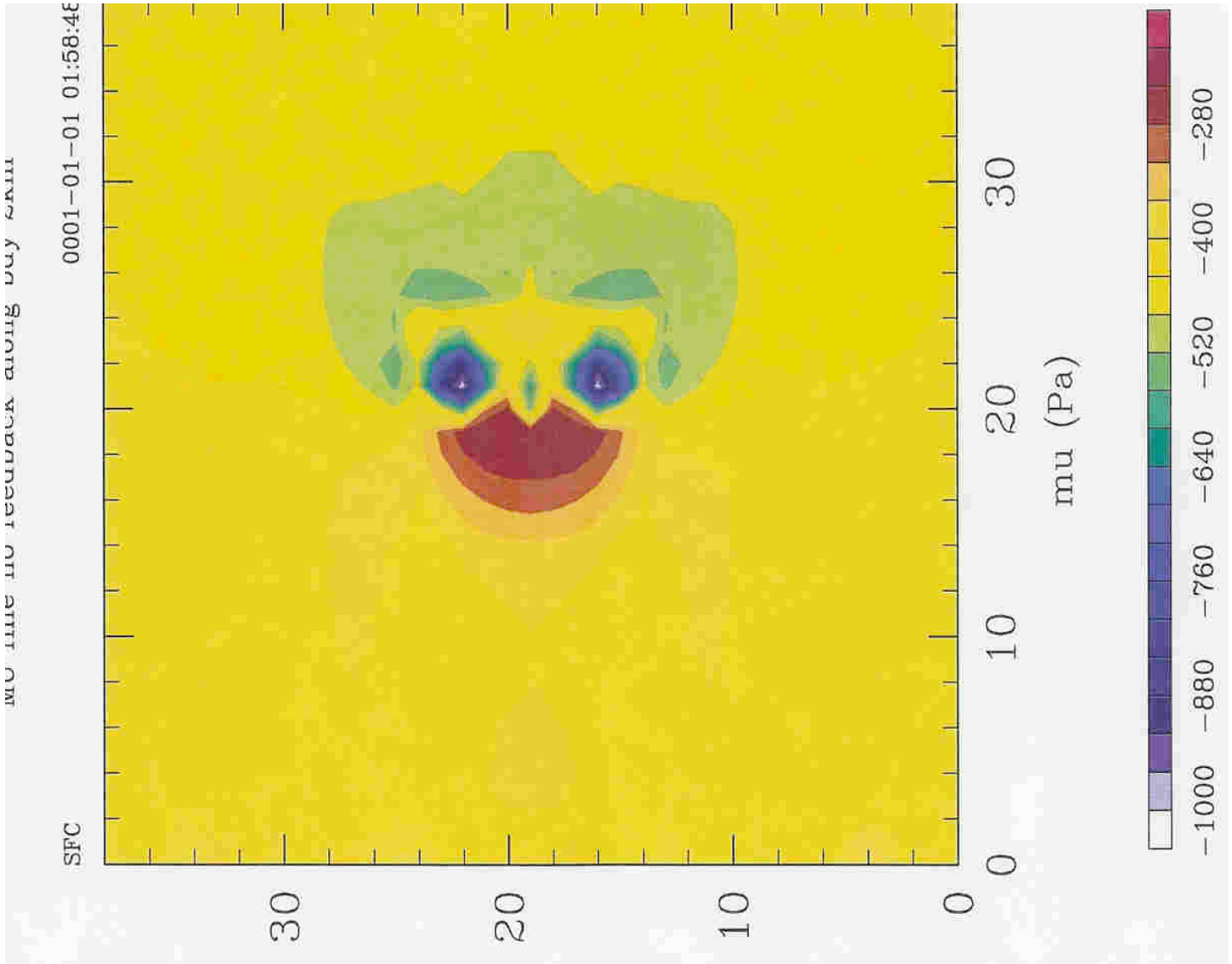
Nesting in WRF

Dave Gill
Matthew Pyle



ARW Nesting


SFC
0001-01-01 01:58:48
0
10
20
30
mu (Pa)
-1000
-880
-760
-640
-520
-400
-280

Nesting Basics - What is a nest

- A nest is a ***finer-resolution*** model run. It may be ***embedded*** simultaneously within a coarser-resolution (parent) model run, or ***run independently*** as a separate model forecast.
- The nest ***covers a portion*** of the parent domain, and is driven along its ***lateral boundaries*** by the parent domain.
- Nesting enables running at finer resolution without the following problems:
  - Uniformly high resolution over a large domain - prohibitively expensive
  - High resolution for a very small domain with mismatched time and spatial lateral boundary conditions

Nesting Basics - NMM

- Static, one- or two-way nesting
  - Static: The nest location is fixed in space
  - One-way: Information exchange between the parent and the nest is strictly down-scale. The nest solution does not feedback to the coarser/parent solution.
  - Two-way: Information exchange between the parent and the nest is bi-directional. The nest feedback impacts the coarse-grid domain's solution.
  - Fine grid input is for non-meteorological variables.
- Automatic moving nests are available, primarily for hurricane tracking (HWRF)

Nesting Basics - ARW

- One-way nesting via multiple model forecasts
- One-way nesting with a single model forecast, without feedback

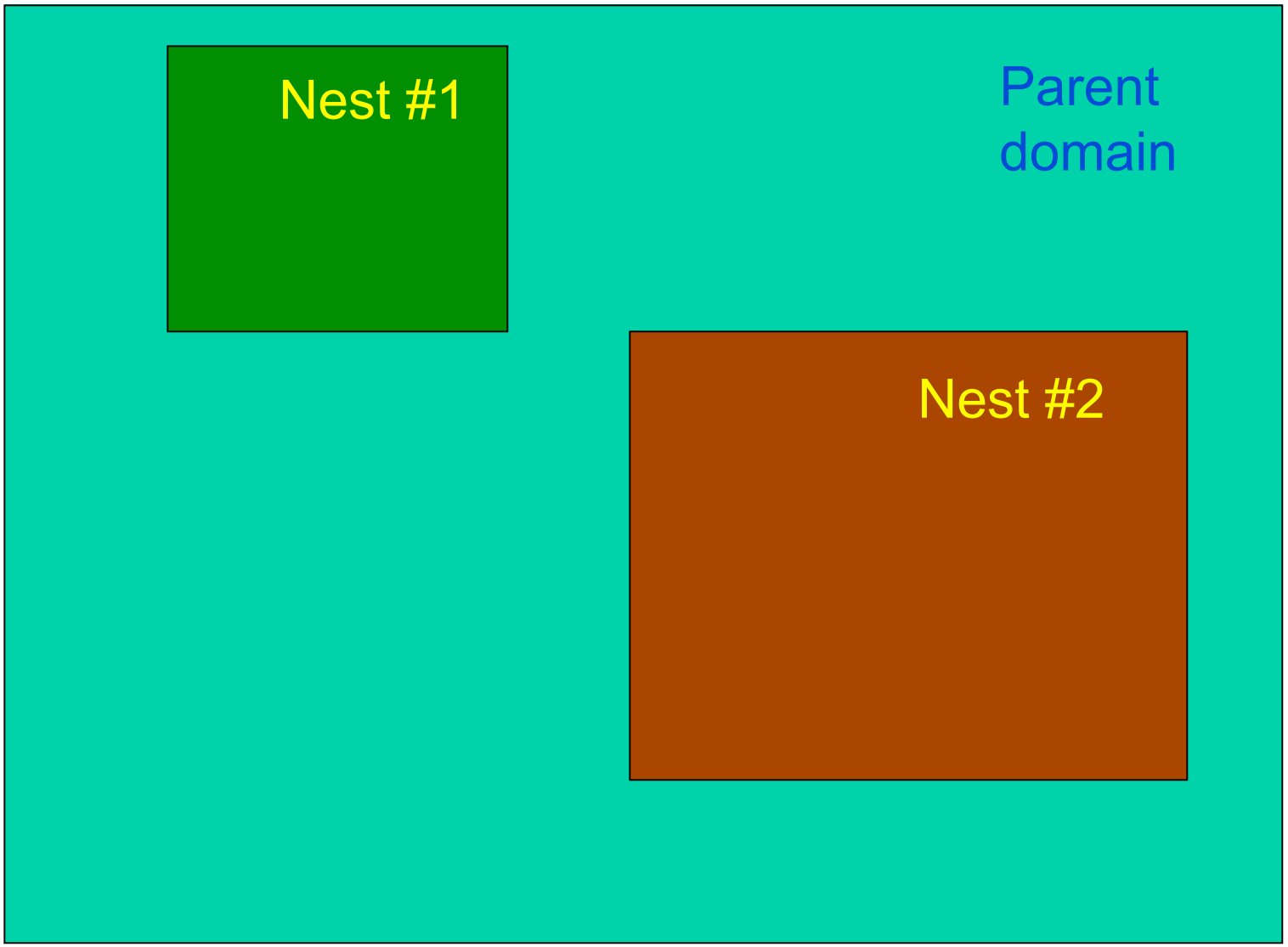
- One-way/two-way nesting with a single input file, all fields interpolated from the coarse grid
- One-way/two-way nesting with multiple input files, each domain with a full input data file
- One-way/two-way nesting with the coarse grid data including all meteorological fields, and the fine-grid domains including only the static files

- One-way/two-way nesting with a specified move for each nest
- One-way/two-way nesting with an automatic move on the nest determined through 500 mb low tracking

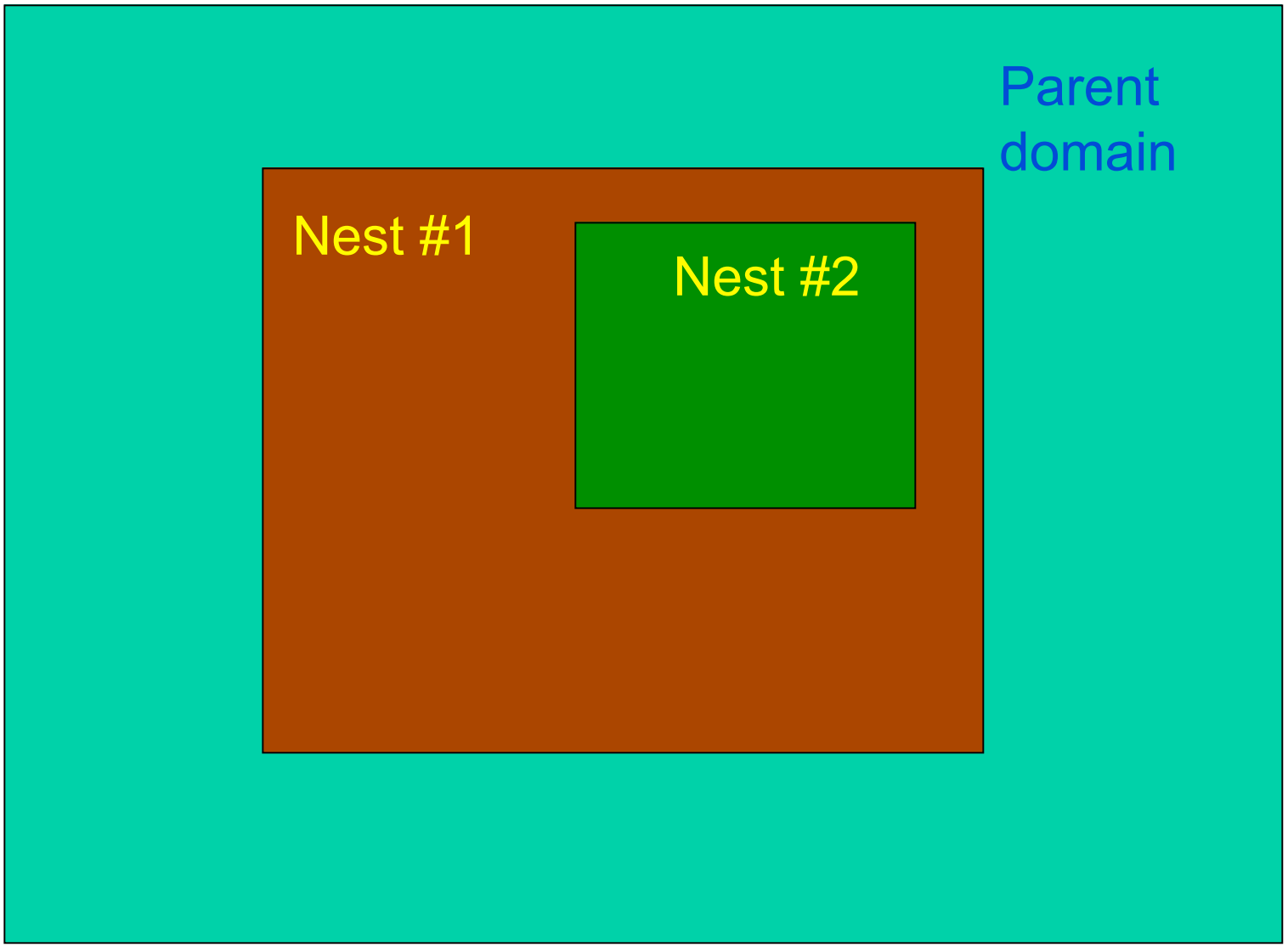
Some Nesting Hints

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

Two nests on the same “level”, with a common parent domain

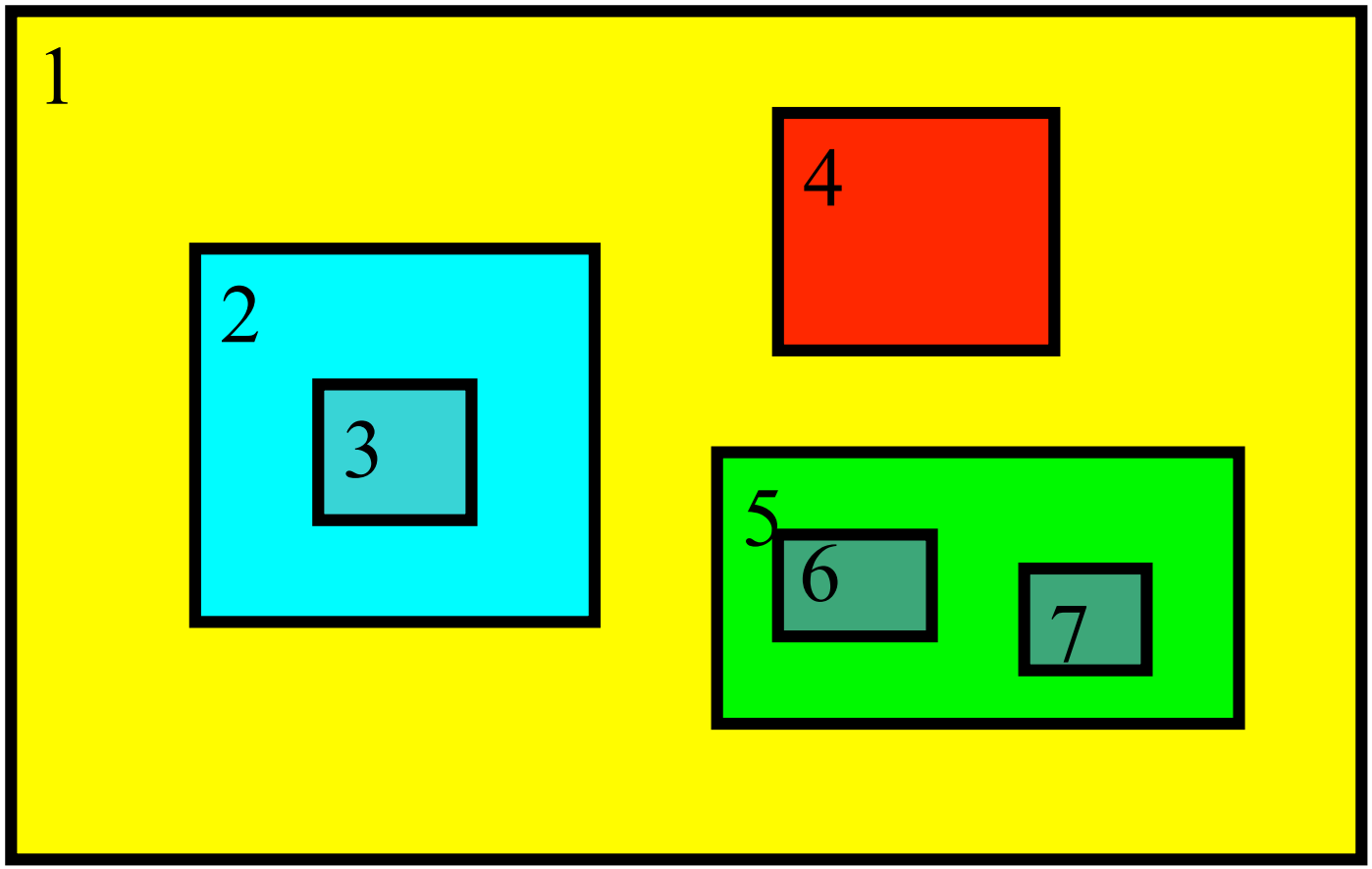


Two levels of nests, with nest #1 acting as the parent for nest #2



These are all OK

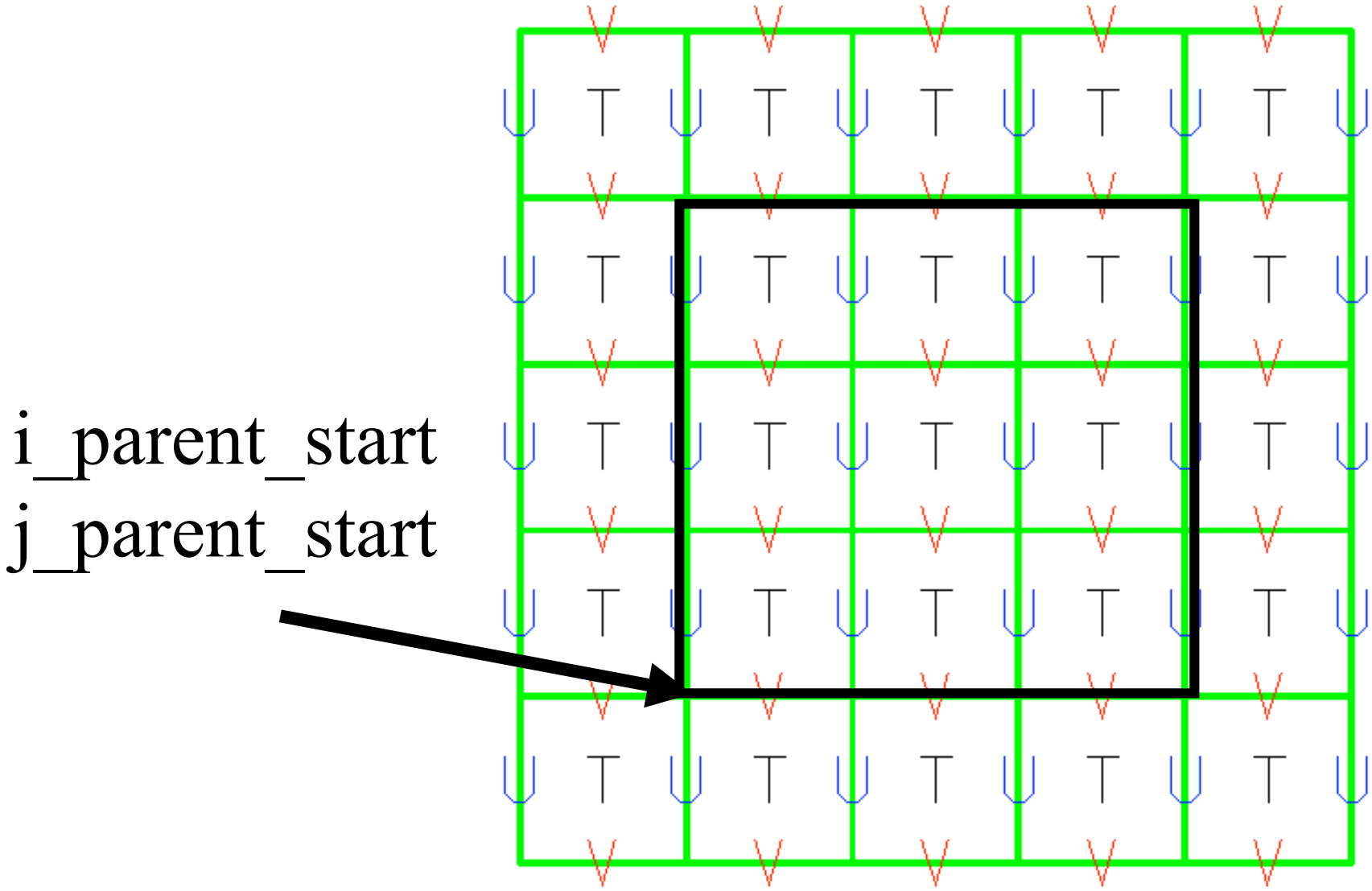
Telescoped to any depth
Any number of siblings



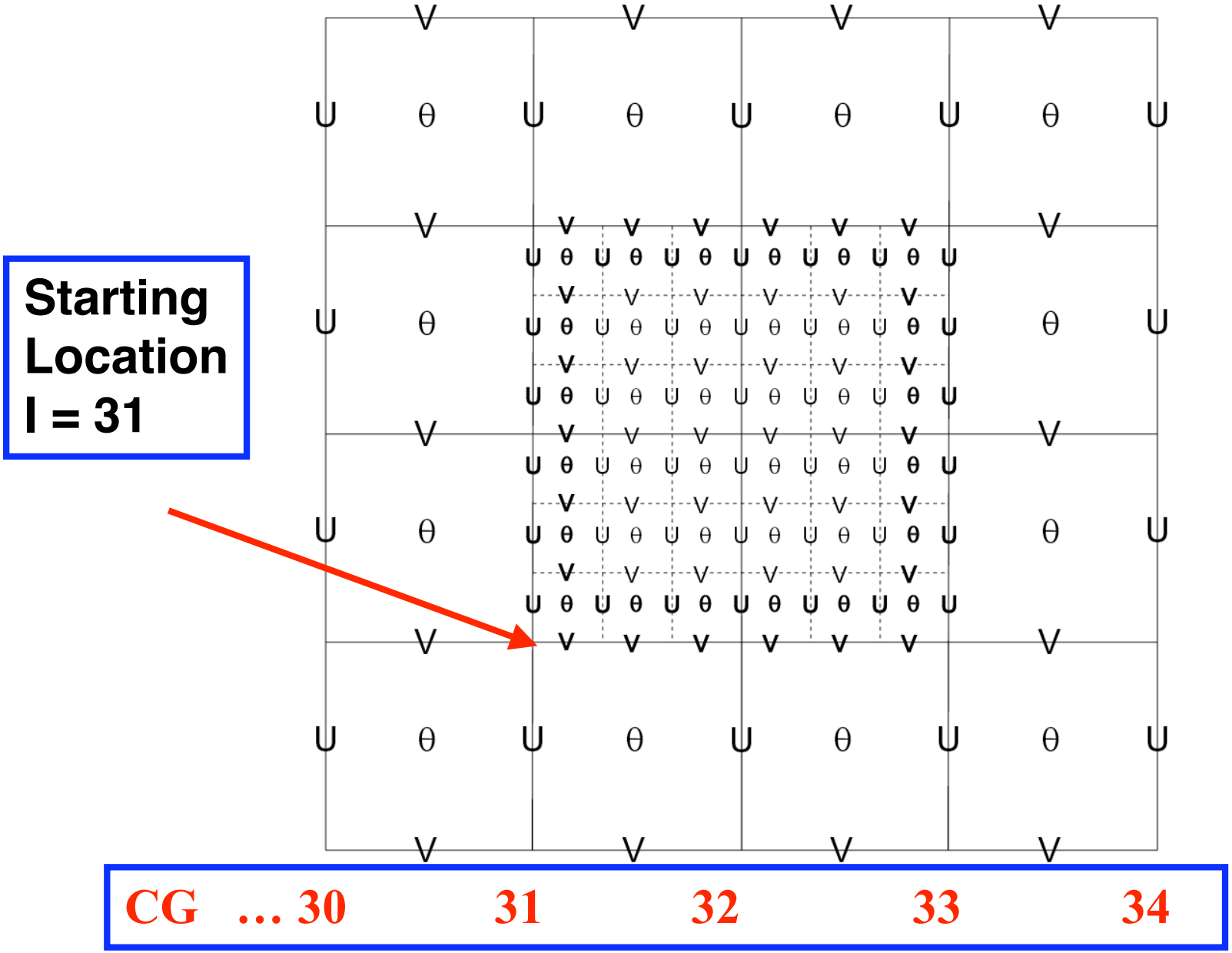
Some Nesting Hints

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

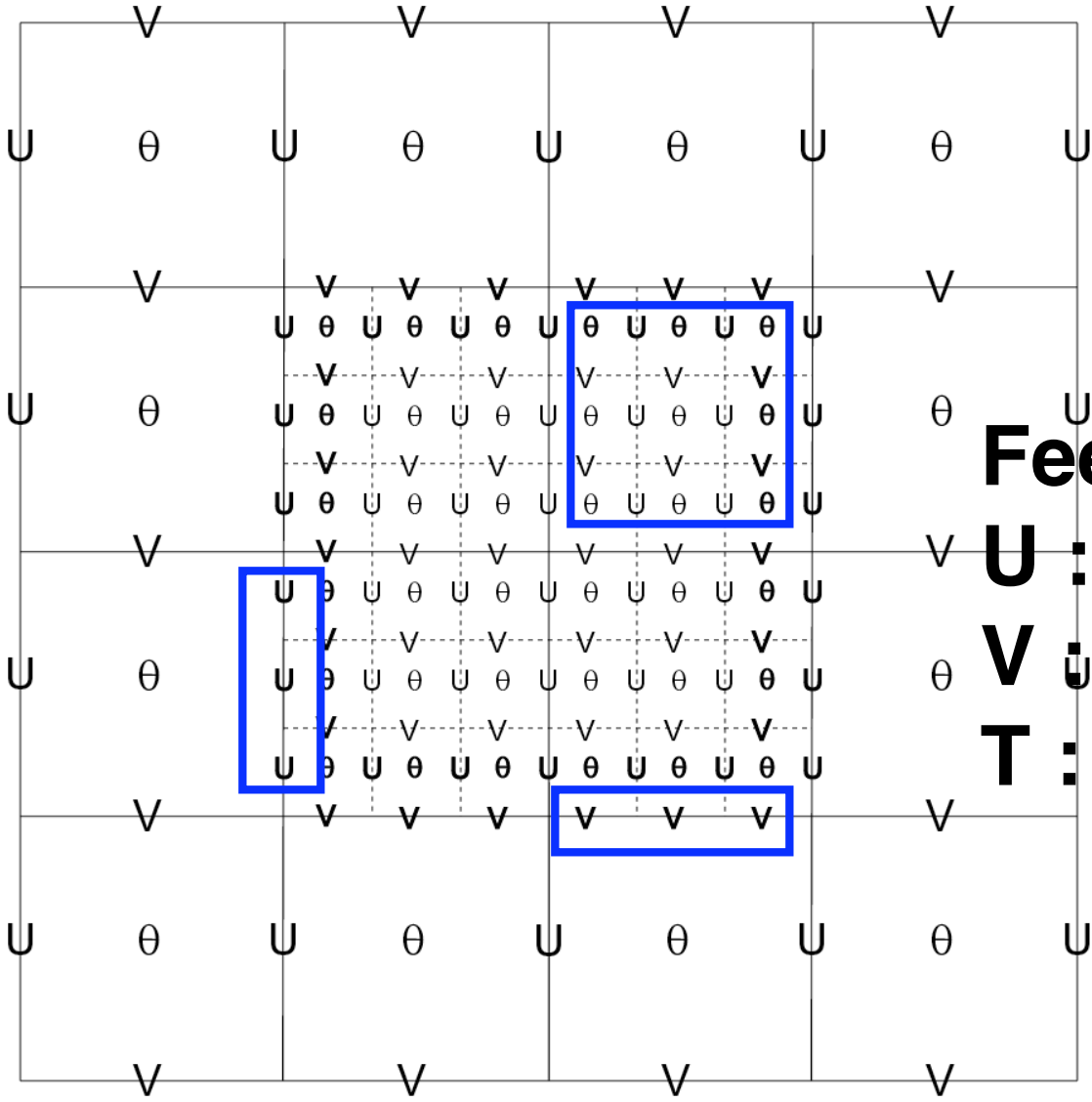
ARW Coarse Grid Staggering



ARW Coarse Grid Staggering 3:1 Ratio

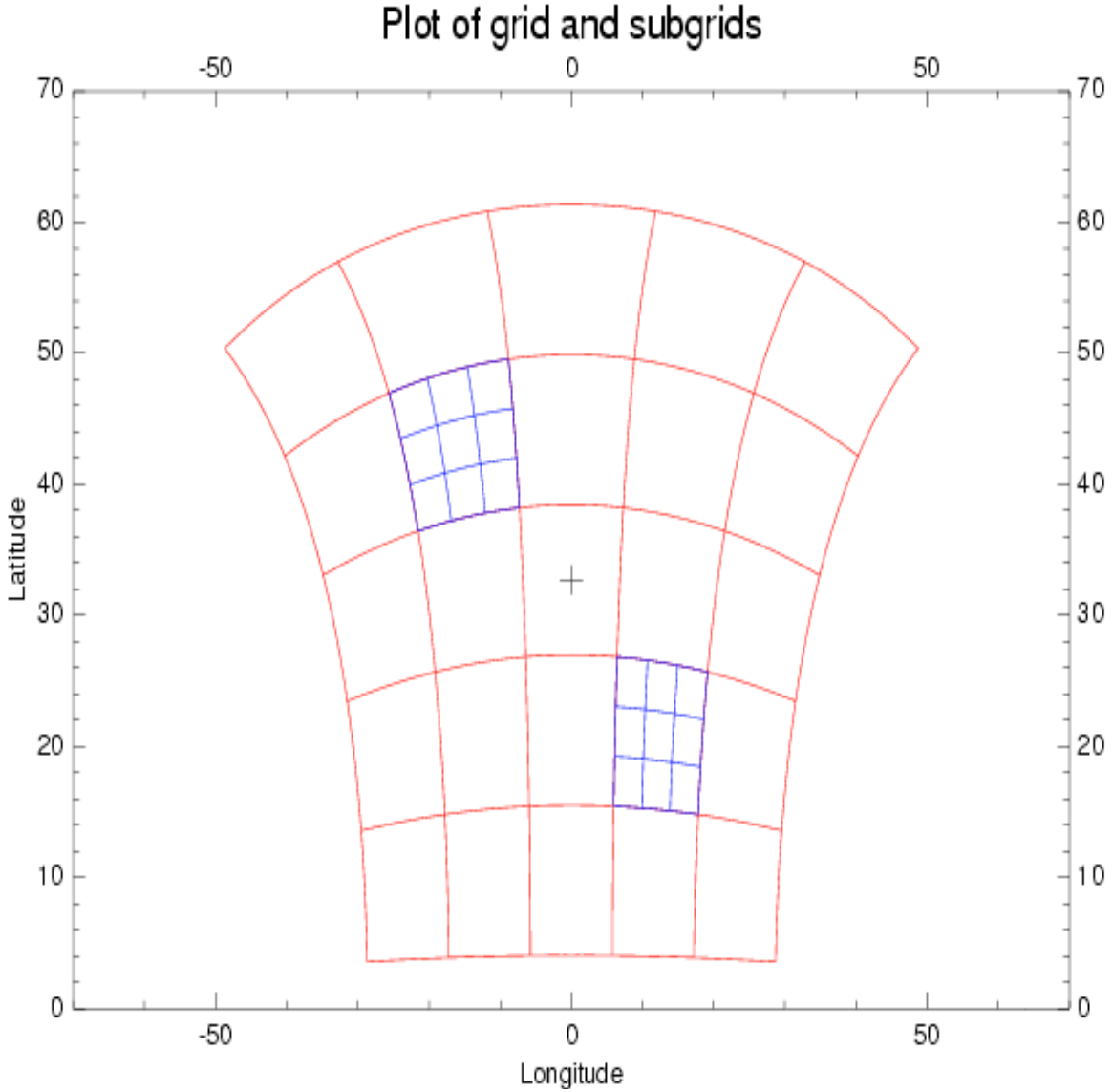


ARW Coarse Grid Staggering 3:1 Ratio



Feedback:
U : column
V : row
T : cell

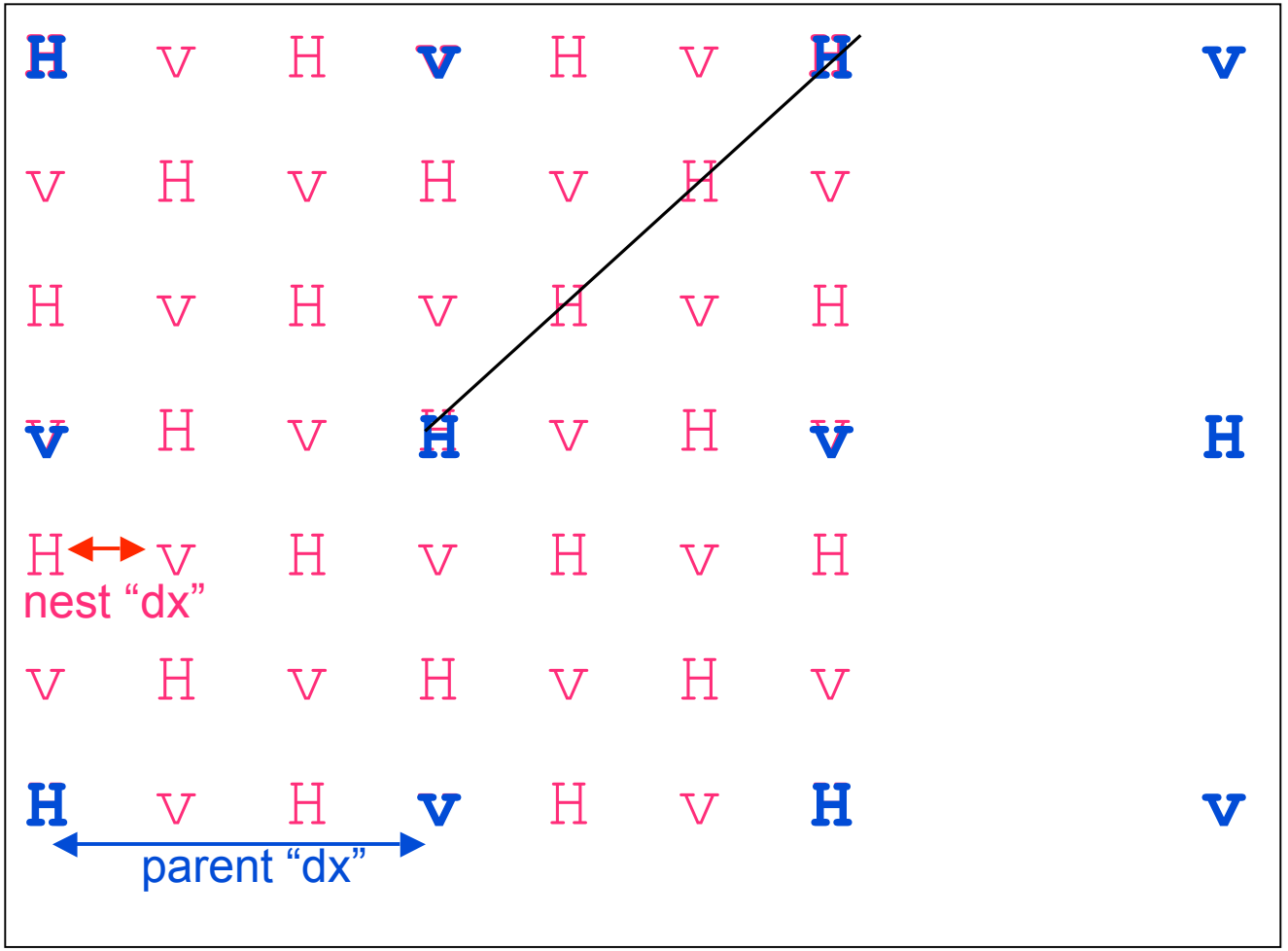
NMM Coarse/Fine Overlay



NMM Telescopic E-Grid

- Interpolations are done on the rotated latitude/longitude projection. The fine grid is coincident with a portion of the high-resolution grid that covers the entire coarse grid.
- The nested domain can be placed anywhere within the parent domain and the nested grid cells will exactly overlap the parent cells at the coincident cell boundaries.
- Coincident parent/nest grid points eliminate the need for complex, generalized remapping calculations, and enhances model performance and portability.
- The grid design was created with moving nests in mind.

An odd grid ratio introduces parent/nest points being coincident, and a 3:1 ratio is preferred as it has been extensively tested.



Some Nesting Hints

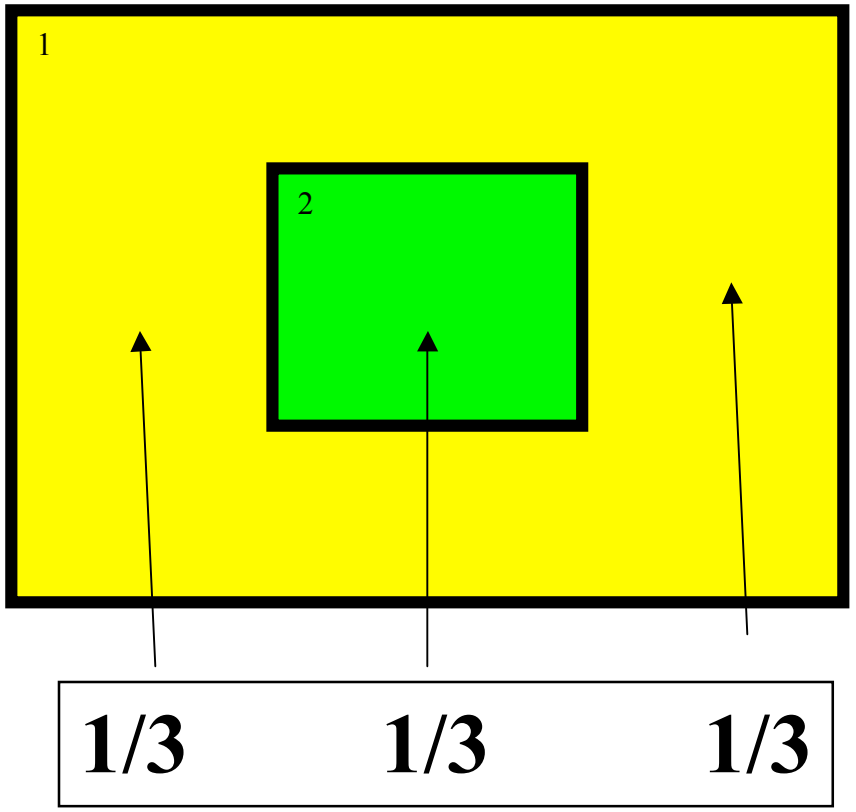
- Allowable domain specifications
- Defining a starting point
- Illegal domain specifications
- 1-way *vs* 2-way nesting

Nesting Performance

- The size of the nested domain may need to be chosen with computing performance in mind.
- Assuming a 3:1 ratio and the same number of grid cells in the parent and nest domains, the fine grid will require 3x as many time steps to keep pace with the coarse domain.
- A simple nested domain forecast is approximately 4x the cost of just the coarse domain.
- Don't be *cheap* on the coarse grid, doubling the CG points results in only a 25% nested forecast time increase.

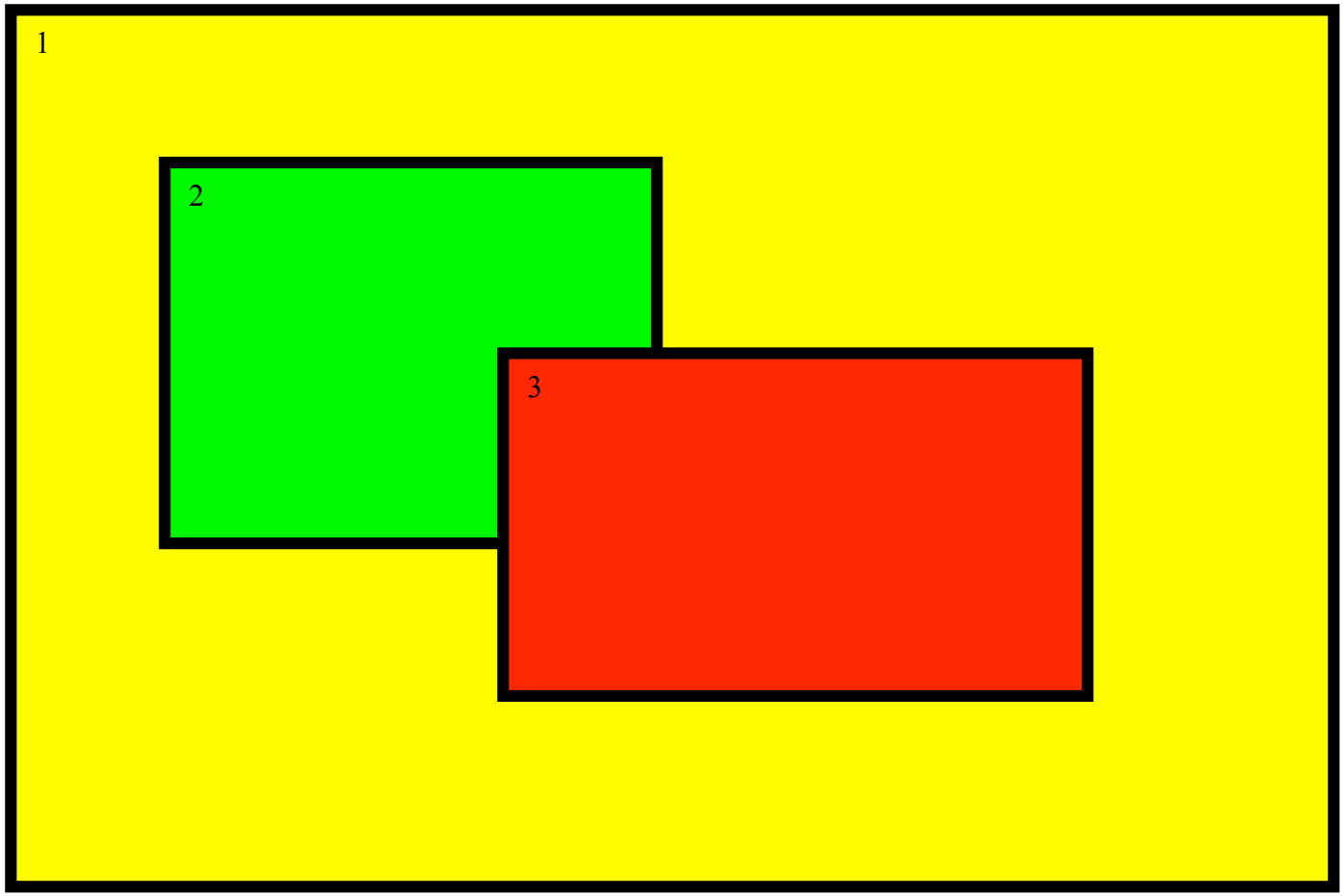
Nesting Performance

- The minimum distance between the nest boundary and the parent boundary is FOUR grid cells
- You should have a much larger buffer zone
- It is not unreasonable to have approximately 1/3 of your coarse-grid domain surrounding each side of your nest domain



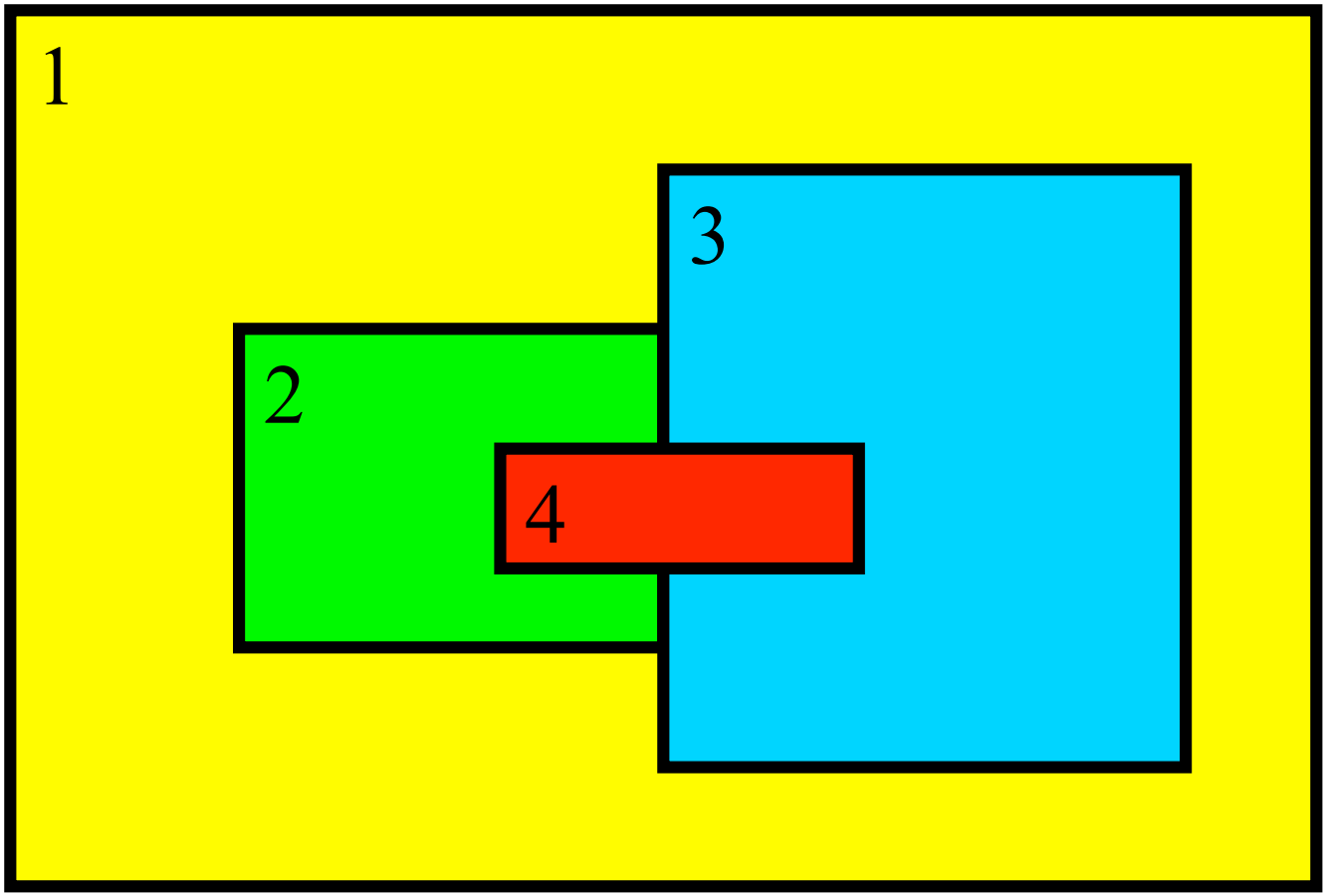
Not OK for 2-way

Child domains *may not* have overlapping points in the parent domain (1-way nesting excluded).



Not OK either

Domains have one, and only one, parent -
(domain 4 is NOT acceptable even with 1-way nesting)



Some Nesting Hints

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

Nesting Performance

- The size of the nested domain may need to be chosen with computing performance in mind.
- Assuming a 3:1 ratio and the same number of grid cells in the parent and nest domains, the fine grid will require 3x as many time steps to keep pace with the coarse domain.
- A simple nested domain forecast is approximately 4x the cost of just the coarse domain.
- Don't be *cheap* on the coarse grid, doubling the CG points results in only a 25% nested forecast time increase.

NMM: Initial Conditions

- Simple horizontal bilinear interpolation of the parent initial conditions is used to initialize all meteorological fields on the nest.
- A nearest-neighbor approach is adopted for prescribing most of the land-state variables.
- Topography and land-sea mask are redefined over the nested domain using the appropriate "nest level" of WPS info from geogrid.
- Quasi-hydrostatic mass balancing is carried out after introducing the high-resolution topography.

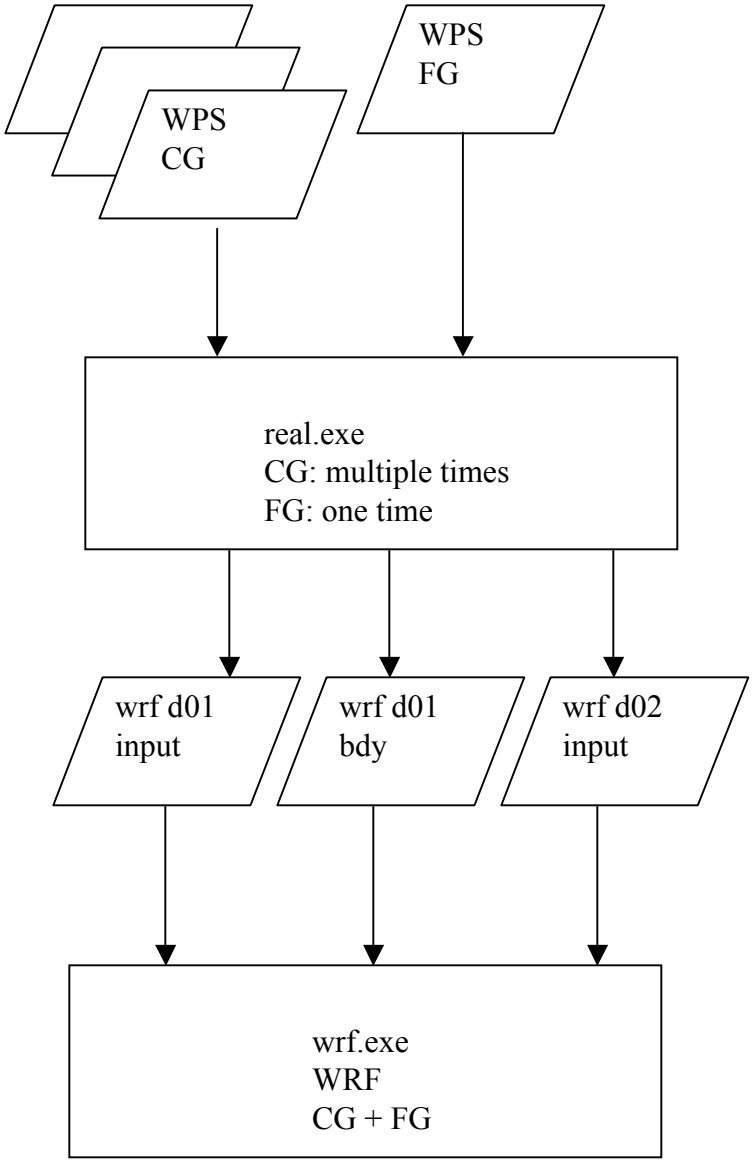
ARW: 2-Way Nest with 2 Inputs

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



ARW: 2-Way Nest with 2 Inputs

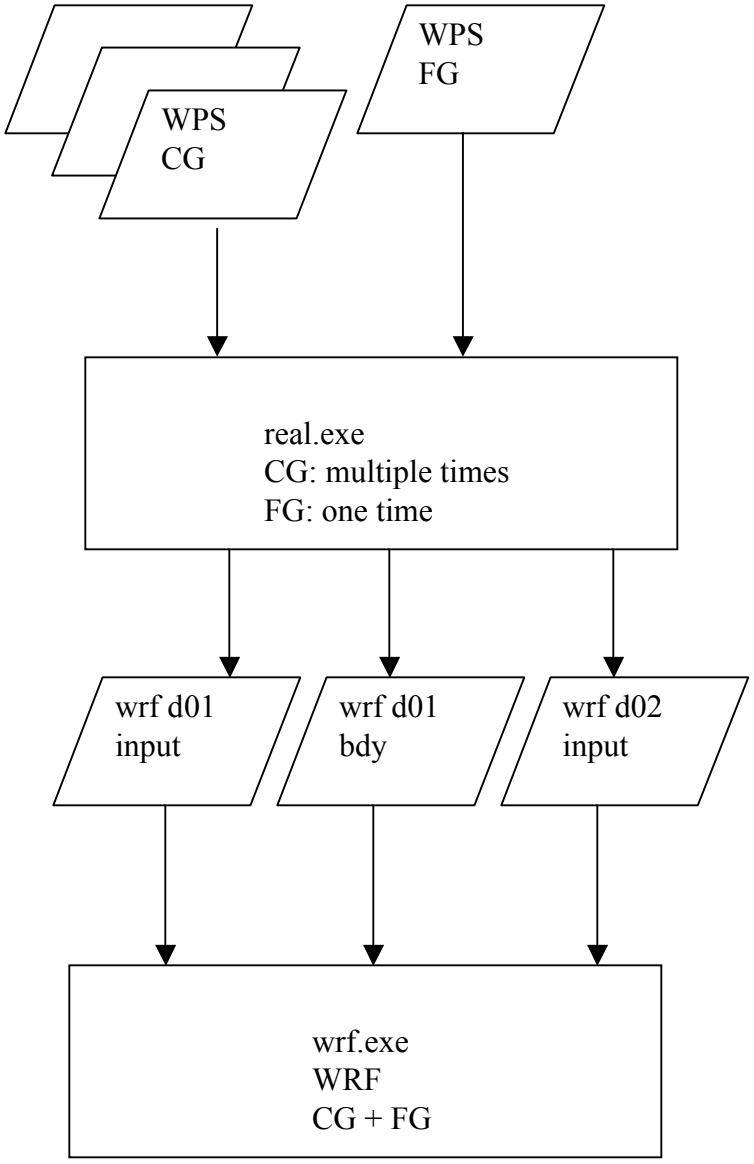
No vertical nesting

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

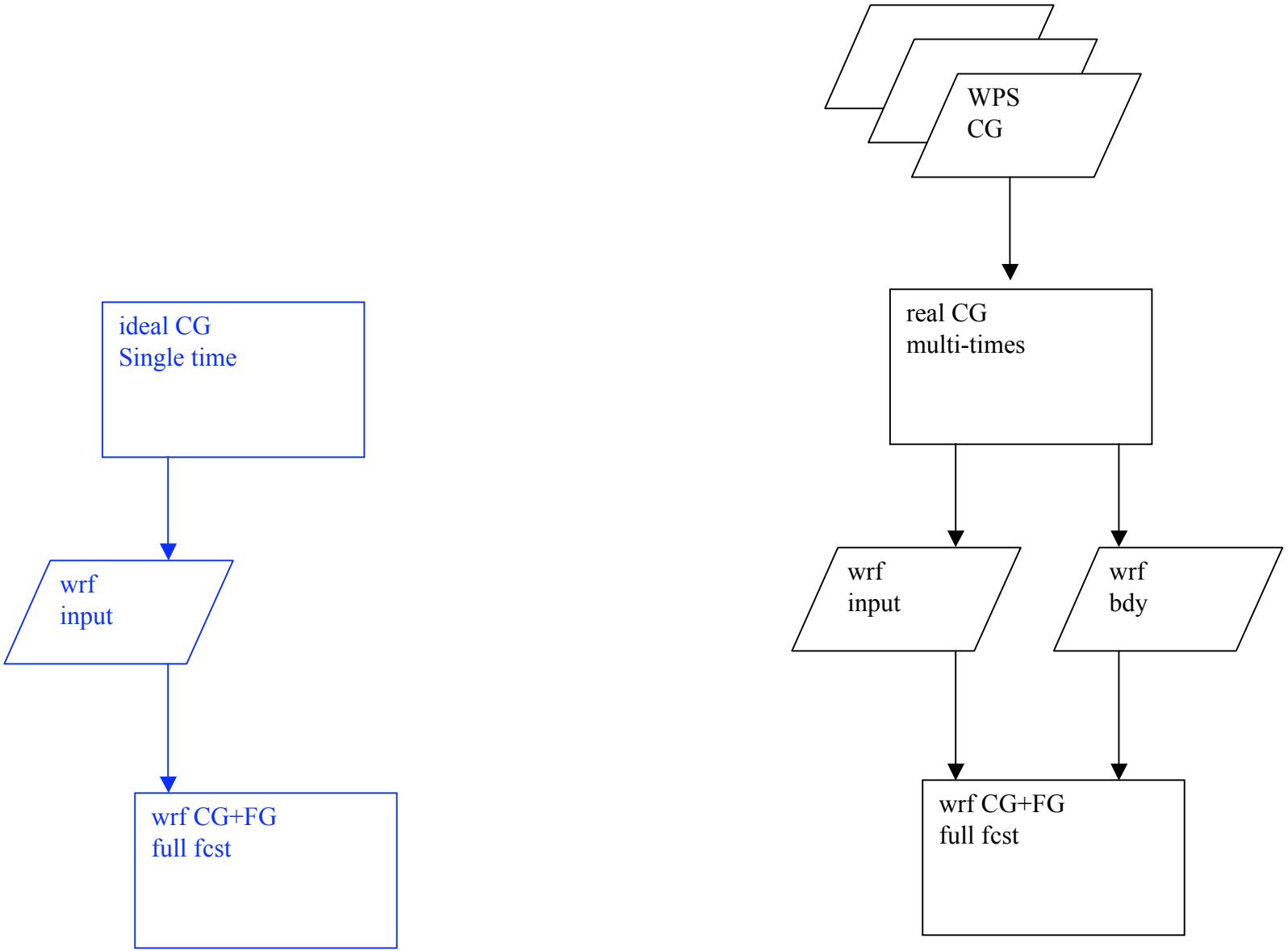
The grid distance ratio is not strictly tied to the time step ratio

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



ARW: 2-Way Nest with 1 Input



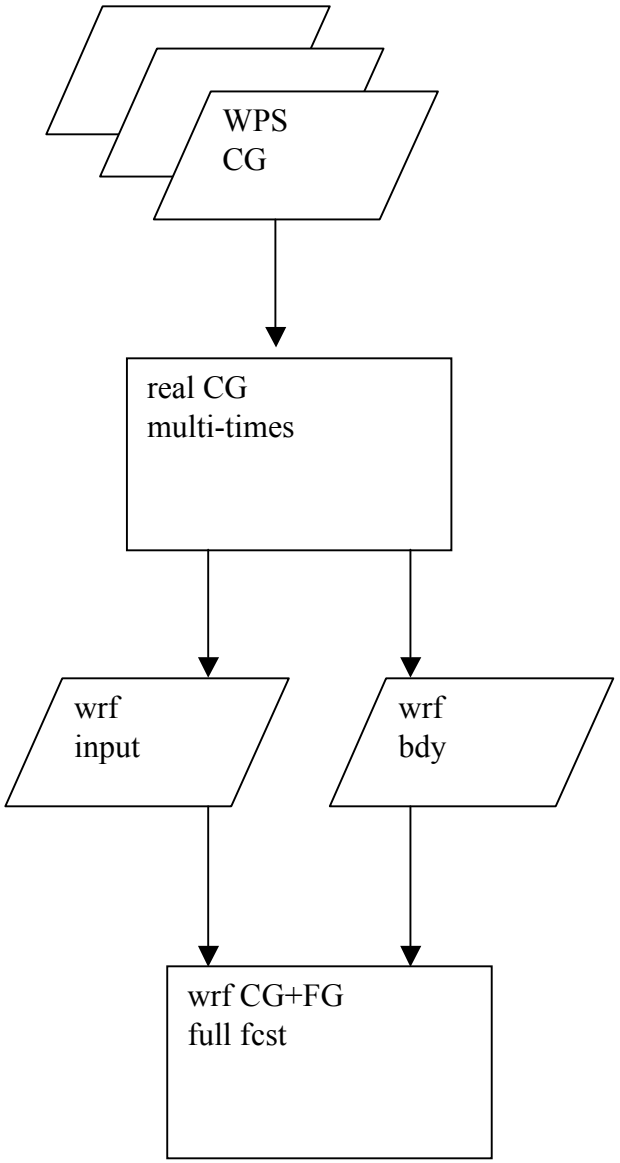
ARW: 2-Way Nest with 1 Input

A single namelist column entry is tied to each domain

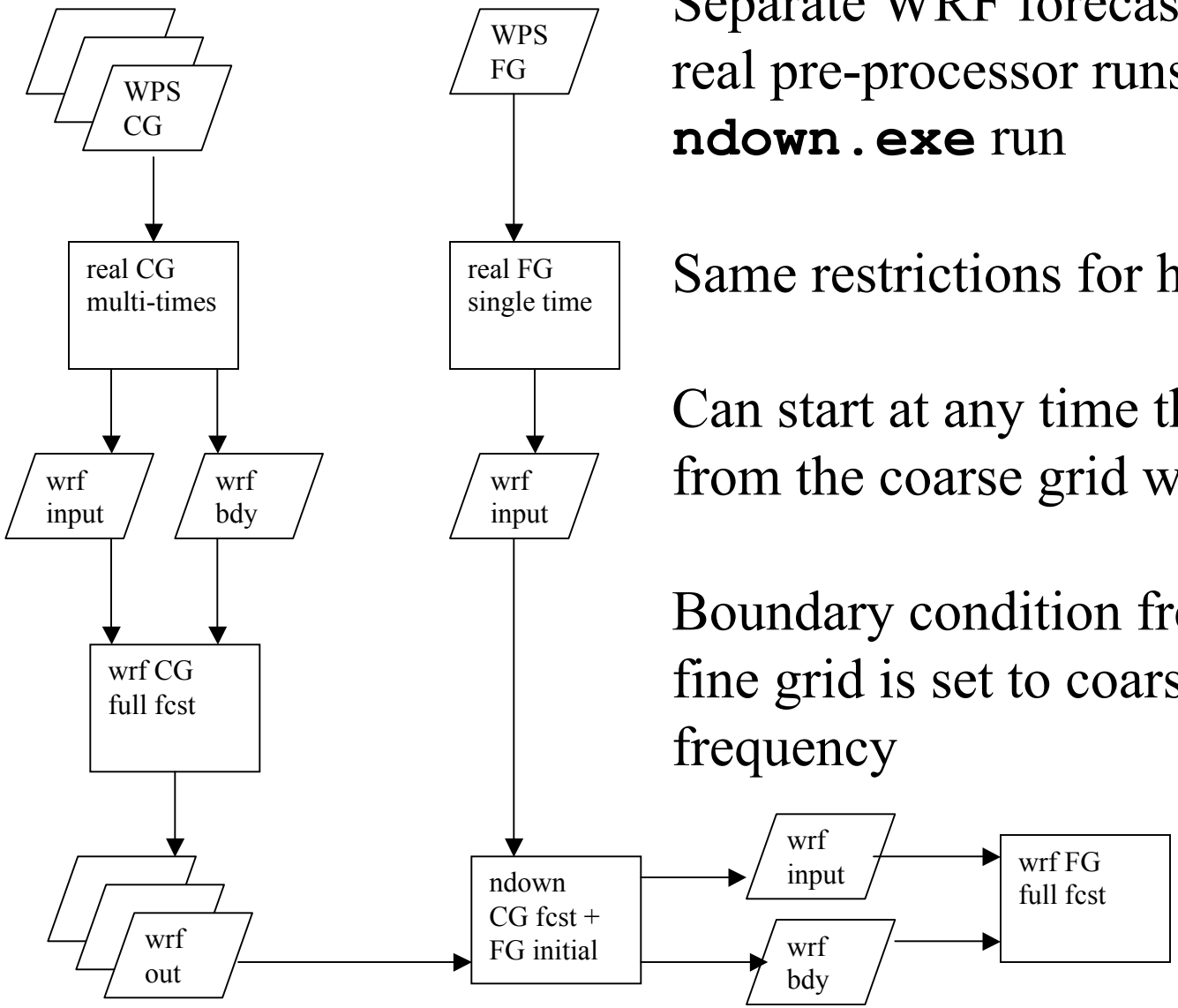
The horizontal interpolation method, feedback, and smoothing are largely controlled through the Registry file

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 **`./WRFV3/test/em_real`** directory



ndown: 1-Way Nest with 2 Inputs



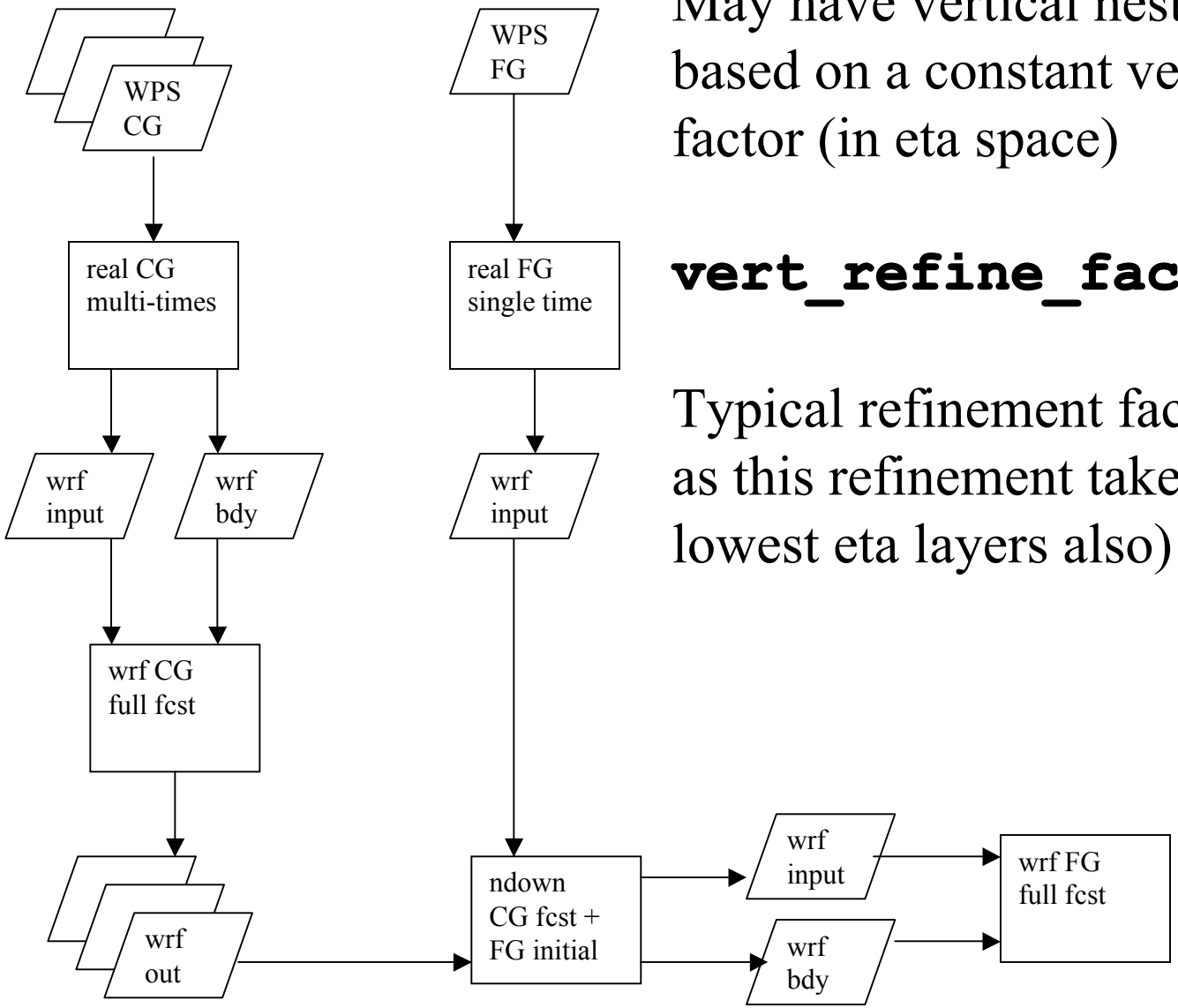
Separate WRF forecast runs, separate real pre-processor runs, intervening **`ndown.exe`** run

Same restrictions for horizontal 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

ndown: 1-Way Nest with 2 Inputs



May have vertical nesting on the fine grid based on a constant vertical refinement factor (in eta space)

`vert_refine_fact`

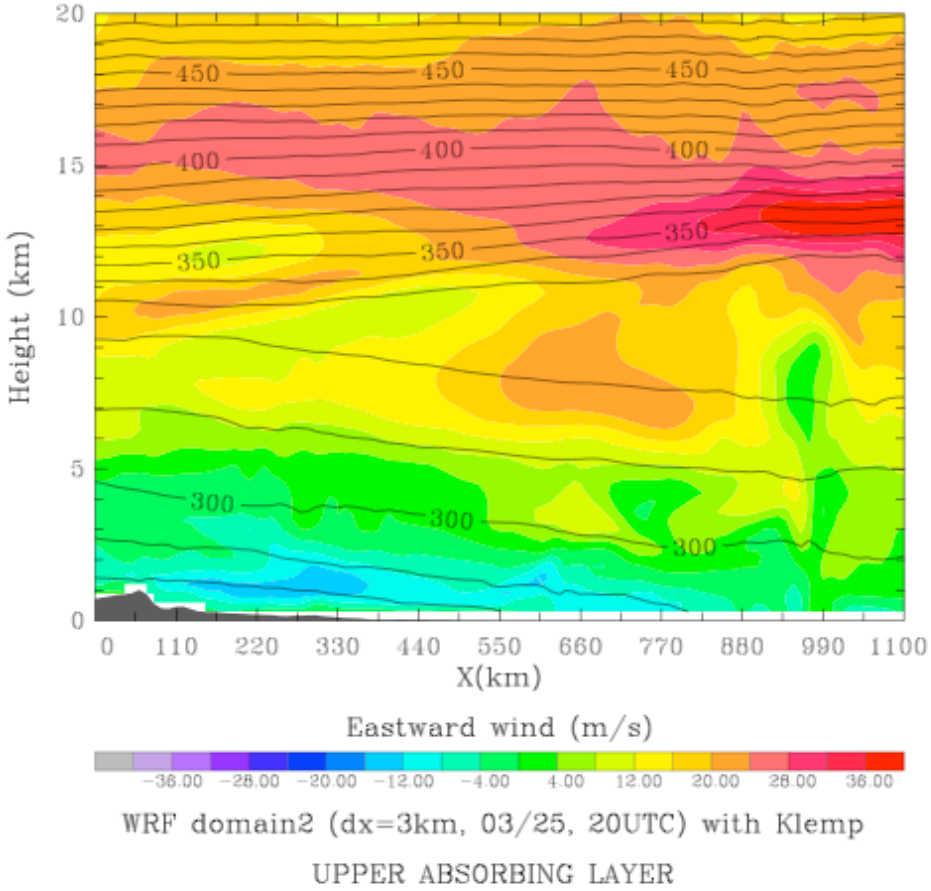
Typical refinement factors 2-5 (be careful, as this refinement takes place in the lowest eta layers also)

West East Cross section

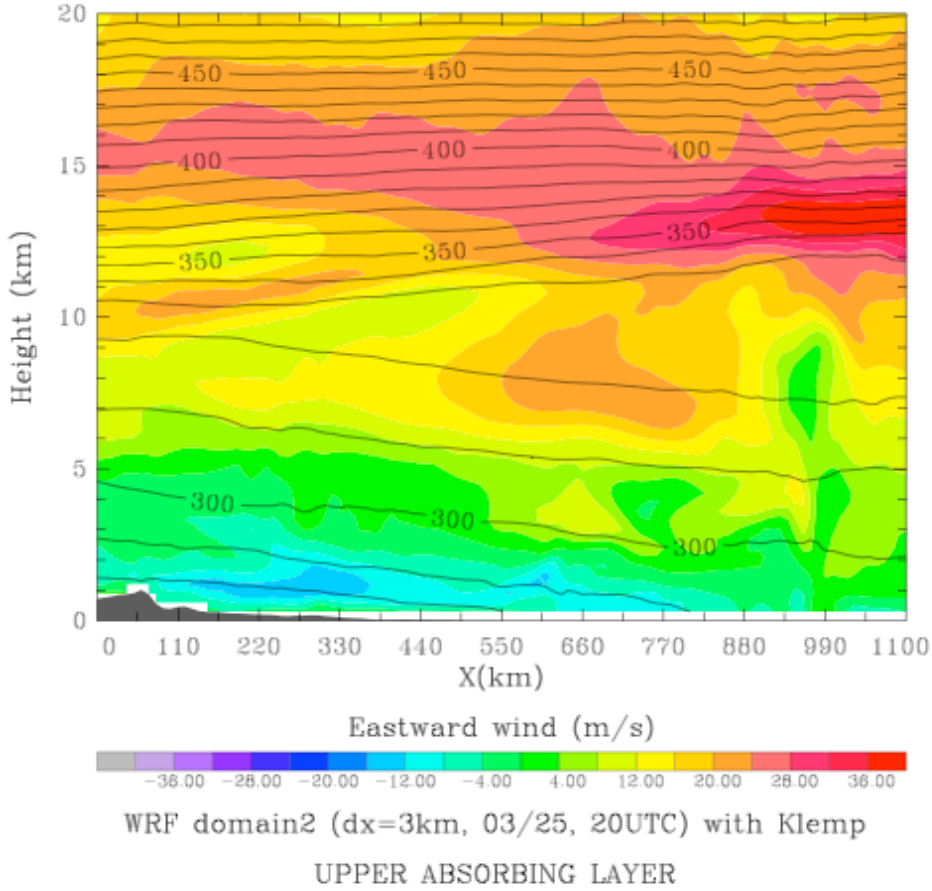
Shaded: v; Contour: theta

6-h Forecast, from Mohamed Moustaoui

Standard Levels



3x Refinement



Some Nesting Hints

- Allowable domain specifications
- Defining a starting point
- Illegal domain specifications
- 1-way *vs* 2-way nesting
- Nest logic in WRF source code
- Nest information in the Registry

Allocate and Initialize a Nest

```
DO WHILE ( nests_to_open( grid , nestid , kid ) )
  a_nest_was_opened = .true.
  CALL med_pre_nest_initial ( grid , nestid , &
         config_flags )
  CALL alloc_and_configure_domain ( &
         domain_id  = nestid ,   &
         grid       = new_nest , &
         parent     = grid ,     &
         kid        = kid        )
   CALL Setup_Timekeeping (new_nest)
   CALL med_nest_initial ( grid , new_nest, &
         config_flags )
END DO
```

All Siblings get Processed

```
DO WHILE ( ASSOCIATED( grid_ptr ) )
  CALL set_current_grid_ptr( grid_ptr )
  CALL solve_interface ( grid_ptr )
  CALL domain_clockadvance ( grid_ptr )
  CALL domain_time_test( grid_ptr, &
  'domain_clockadvance' )
  grid_ptr => grid_ptr%sibling
END DO
```

Recursive Nest Depth

```
DO kid = 1, max_nests
  IF ( ASSOCIATED( grid_ptr%nests(kid)%ptr ) ) THEN
    CALL set_current_grid_ptr( grid_ptr%nests(kid)%ptr )
    CALL med_nest_force ( grid_ptr , &
         grid_ptr%nests(kid)%ptr , config_flags )
    grid_ptr%nests(kid)%ptr%start_subtime = &
         domain_get_current_time(grid) - &
         domain_get_time_step(grid)
    grid_ptr%nests(kid)%ptr%stop_subtime = &
         domain_get_current_time(grid)
    CALL integrate ( grid_ptr%nests(kid)%ptr )
    CALL med_nest_feedback ( grid_ptr , &
         grid_ptr%nests(kid)%ptr , config_flags )
  END IF
END DO
```

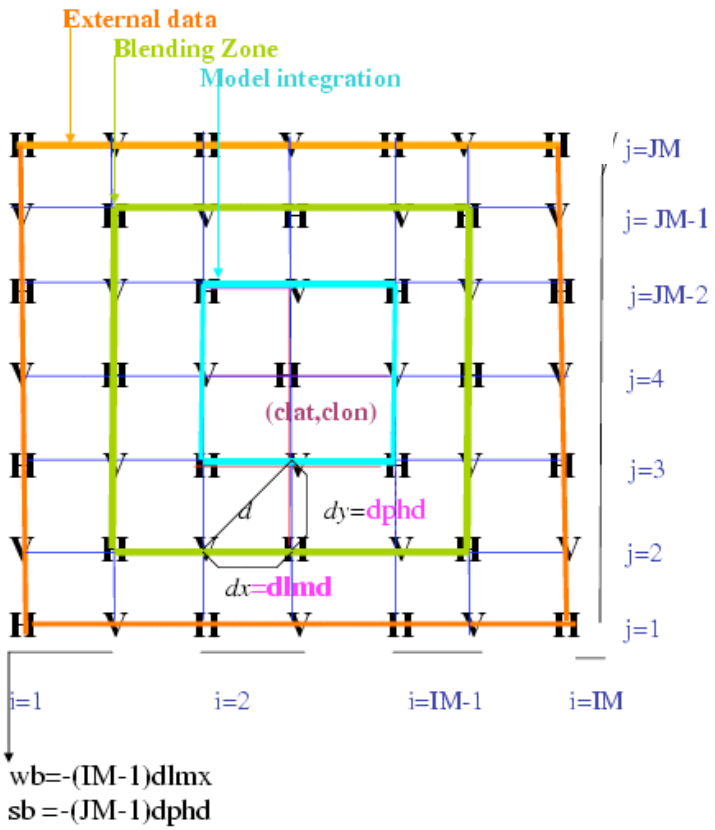
Input vs Interpolating

```
CALL med_interp_domain( parent, nest )

CALL init_domain_constants ( parent, nest )

IF ( nest_config_flags%input_from_file ) THEN

   IF ( nest_config_flags%input_from_file ) THEN
      CALL med_initialdata_input_ptr( nest , &
         nest_config_flags )
   ENDIF
```

Feedback and Domain Sync-ing

```
CALL med_nest_feedback ( parent , nest , &
   config_flags )

CALL start_domain ( nest , .TRUE. )

CALL start_domain ( parent , .TRUE. )
```

NMM Nested LBCs



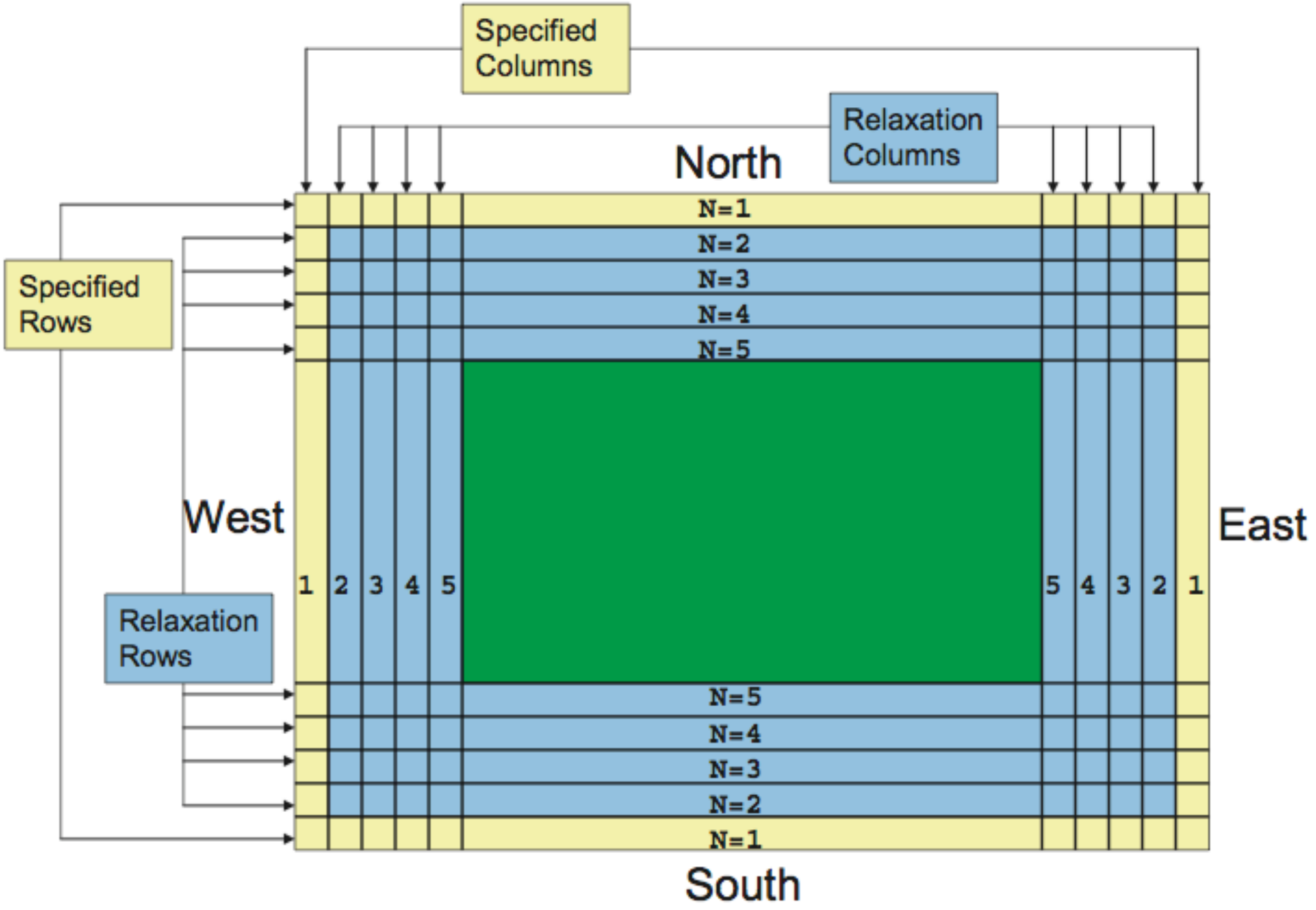
* Given wb,sb, clat and clon, the above rotated lat-lon grid system can be transformed to a lat-lon grid system.

- Nest boundaries generally are treated in the same way as the standard parent domain boundaries:
  - outermost row is prescribed
  - two rows in from boundary is freely integrating
  - in between is a blending zone (average of outermost and freely integrating points)
- The one key difference is frequency of boundary updates: *nested boundaries are updated at every time step of the parent domain*.

NMM Mass Balancing for LBCs

- The parent domain geopotential height, temperature, and moisture are all vertically interpolated (cubic splines) from the hybrid surfaces onto standard isobaric levels.
- Using horizontally interpolated information of the height field from the parent domain, and high-resolution topography from the nest level, mass is adjusted and revised hybrid surfaces are constructed.
- T and q: 1) horizontally interpolated to the nest domain on standard pressure levels, 2) vertically interpolated onto the new hybrid surfaces
- Approach produces an effective way of updating the nest interface without much distortion or noise

Real-Data Lateral Boundary Condition: Location of Specified and Relaxation Zones



ARW Nest Lateral Boundaries

```
fcx(loop) = 1 / (10*dt) * (spec_zone + relax_zone - loop) / (relax_zone - 1)
gcx(loop) = 1 / (50*dt) * (spec_zone + relax_zone - loop) / (relax_zone - 1)
```

Linear weighting outside of outer-most row and column

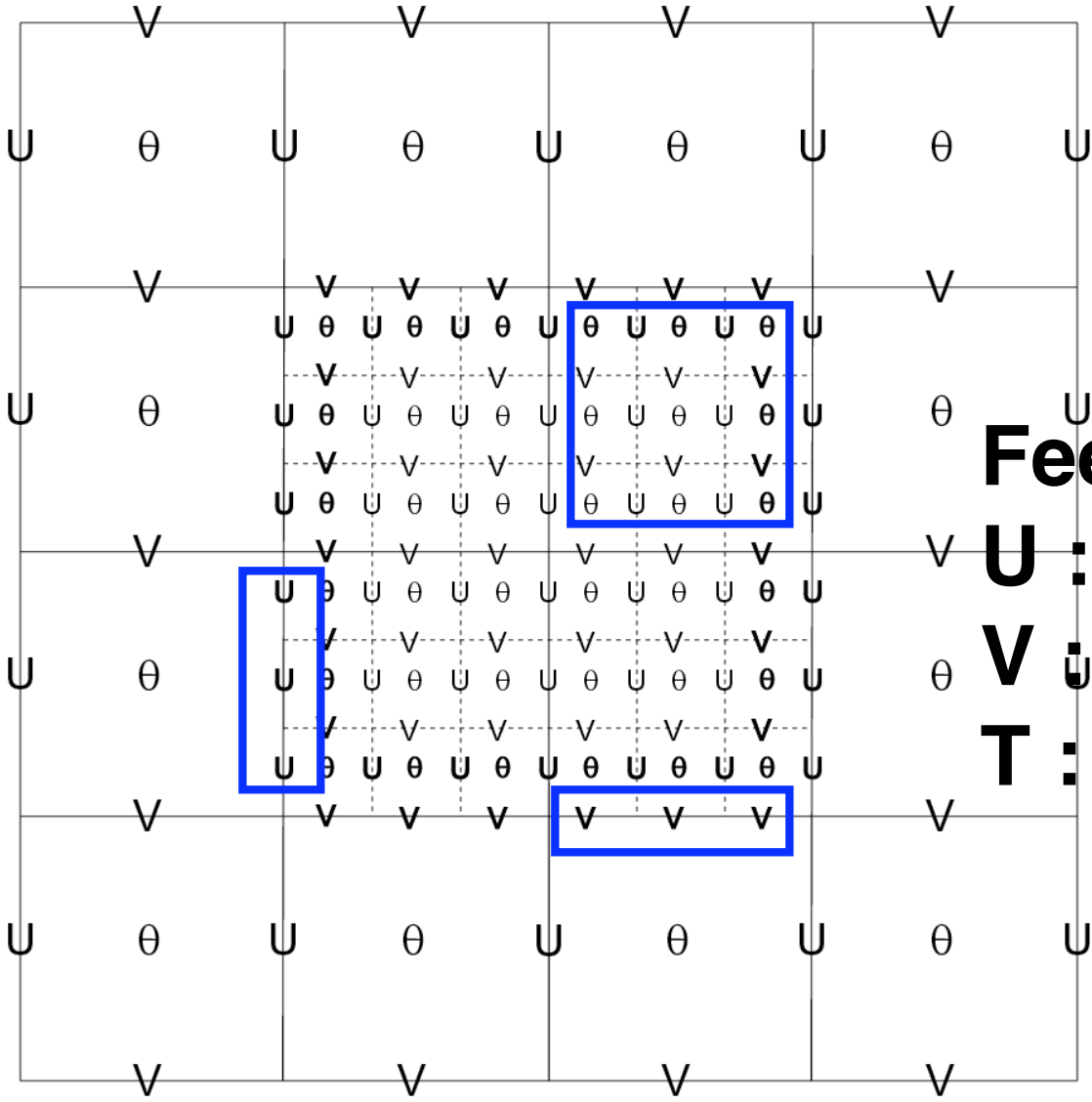
ARW Nest Lateral Boundaries

```
fls0 = field_bdy_ys(i, k, b_dist+1) &
     + dtbc * field_bdy_tend_ys(i, k, b_dist+1) &
     - field(i,k,j)
fls1 = field_bdy_ys(im1, k, b_dist+1) &
     + dtbc * field_bdy_tend_ys(im1, k, b_dist+1) &
     - field(im1,k,j)
fls2 = field_bdy_ys(ip1, k, b_dist+1) &
     + dtbc * field_bdy_tend_ys(ip1, k, b_dist+1) &
     - field(ip1,k,j)
fls3 = field_bdy_ys(i, k, b_dist) &
     + dtbc * field_bdy_tend_ys(i, k, b_dist) &
     - field(i,k,j-1)
fls4 = field_bdy_ys(i, k, b_dist+2) &
     + dtbc * field_bdy_tend_ys(i, k, b_dist+2) &
     - field(i,k,j+1)
```

ARW Nest Lateral Boundaries

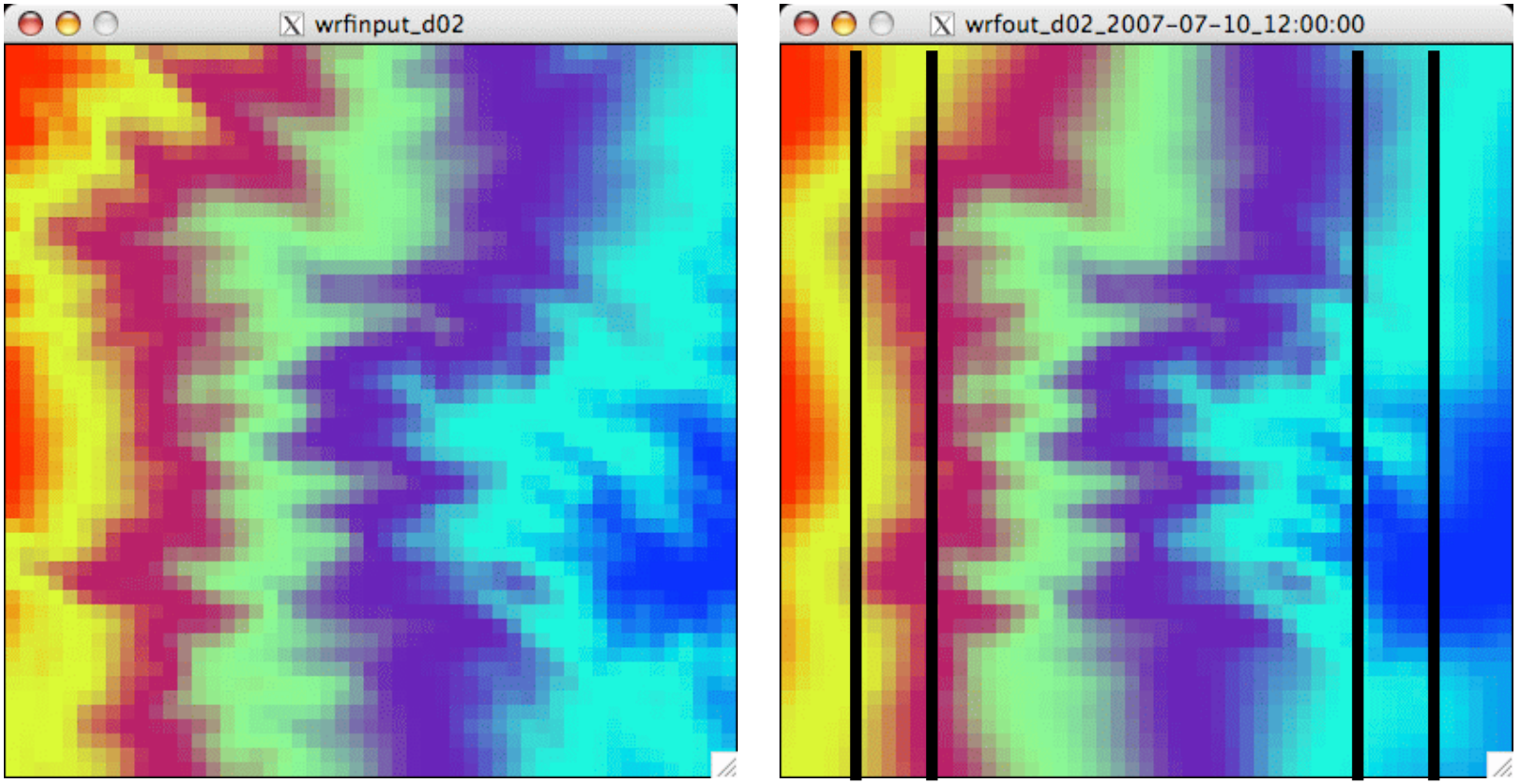
```
field_tend(i,k,j) = field_tend(i,k,j) &
     + fcx(b_dist+1)*fls0 &
     - gcx(b_dist+1)*(fls1+fls2+fls3+fls4-4.*fls0)
```

ARW Coarse Grid Staggering 3:1 Ratio

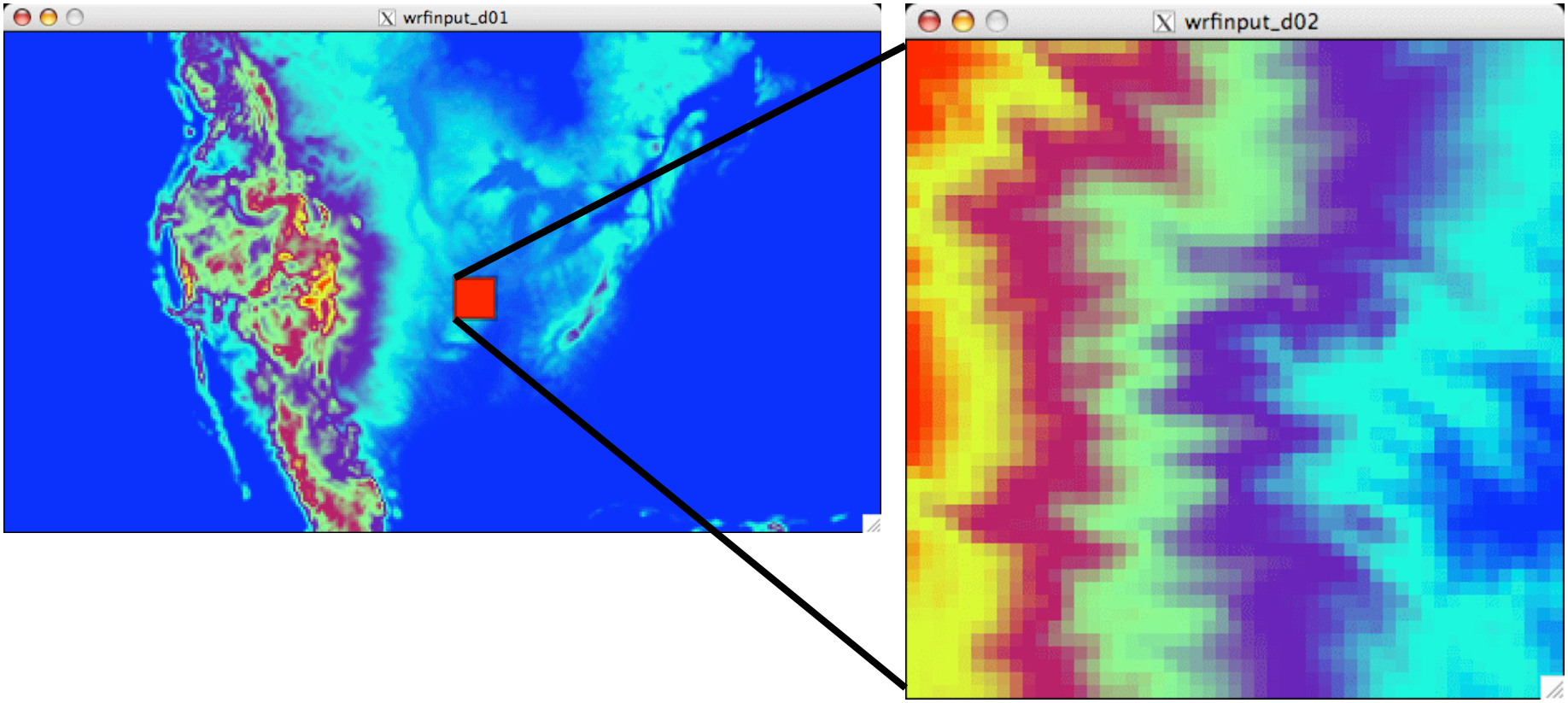


Feedback:
U : column
V : row
T : cell

ARW Lateral Smoothing



Intermediate Domains



The intermediate domain between a parent and a child is the resolution of the coarse grid over the size of the fine grid. It allows the model to re-decompose the domain among all of the processors.

Intermediate Domains - Part 1

```
grid => nested_grid%intermediate_grid
CALL alloc_space_field ( grid, grid%id , 1 , 2 , .TRUE. ,

grid => parent_grid
CALL model_to_grid_config_rec ( grid%id , &
   model_config_rec , config_flags )
CALL couple_or_uncouple_em ( grid , config_flags ,.true. &
#          include "em_actual_new_args.inc" )

grid => nested_grid
CALL model_to_grid_config_rec ( grid%id , &
   model_config_rec , config_flags )
CALL couple_or_uncouple_em ( grid , config_flags ,.true. &
#          include "em_actual_new_args.inc" )
```

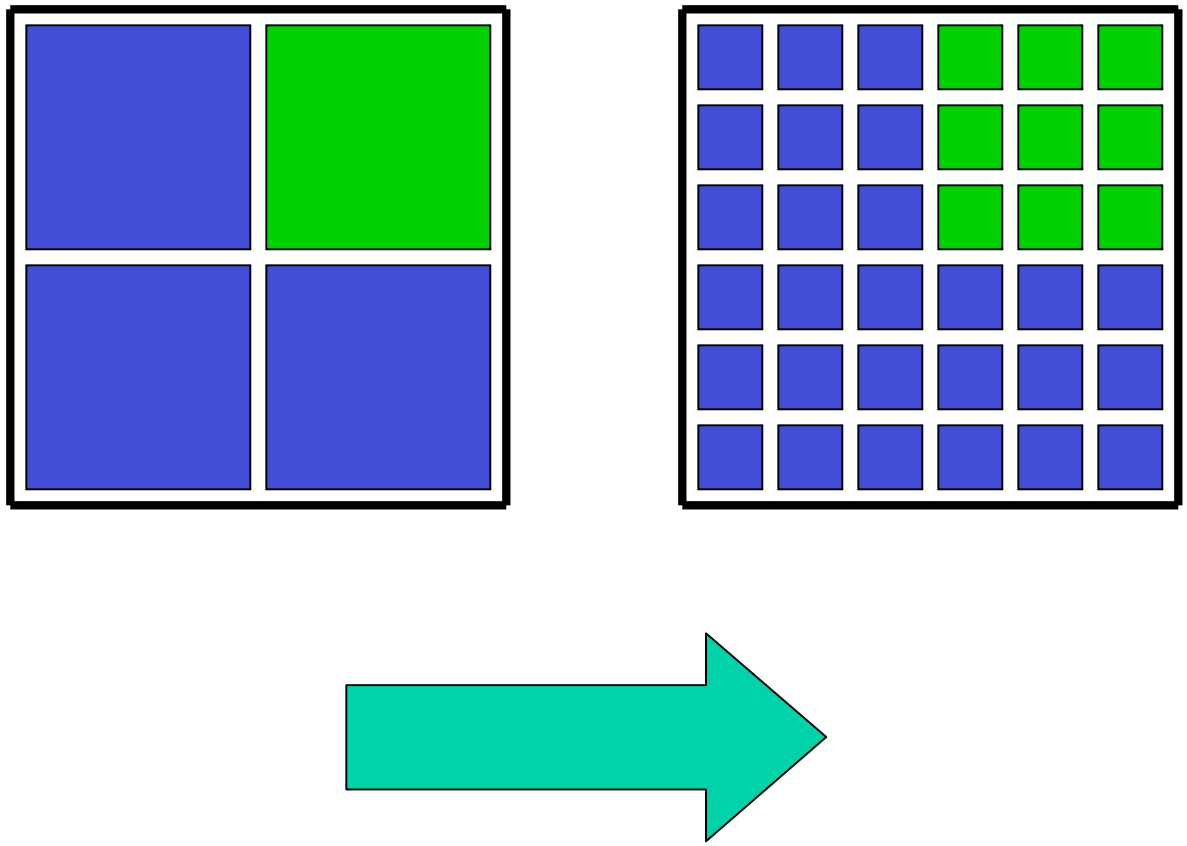
Intermediate Domains - Part 1

```
grid => parent_grid

CALL model_to_grid_config_rec ( grid%id ,
   model_config_rec , config_flags )

CALL interp_domain_em_part1 ( grid , &
   nested_grid%intermediate_grid, nested_grid, &
   config_flags   &
#         include "em_actual_new_args.inc" )
```

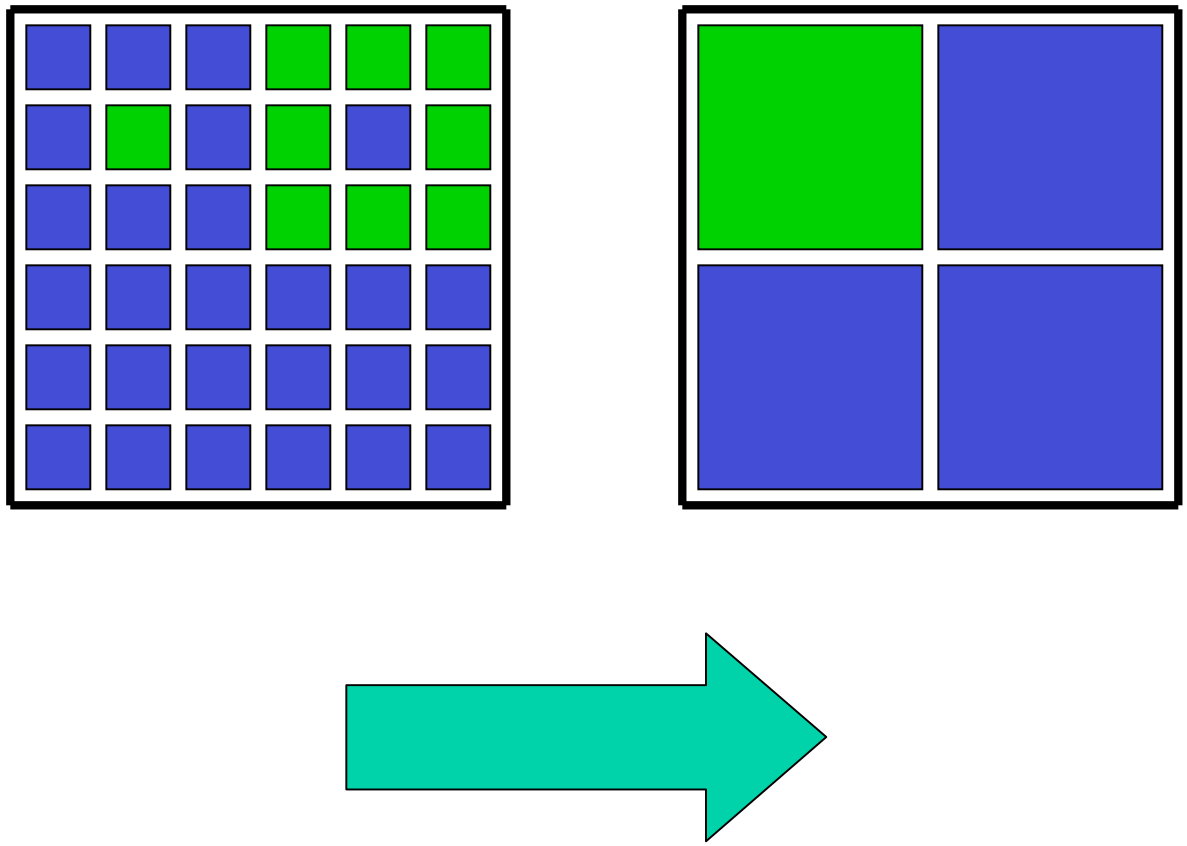
Intermediate Domains - Part 2

```
grid => nested_grid%intermediate_grid
CALL model_to_grid_config_rec ( nested_grid%id , &
   model_config_rec , config_flags )
CALL force_domain_em_part2 ( grid, nested_grid, &
   config_flags   &
#          include "em_actual_new_args.inc")

grid => nested_grid
CALL model_to_grid_config_rec ( grid%id , &
   model_config_rec , config_flags )
CALL couple_or_uncouple_em ( grid , config_flags ,.false. &
#          include "em_actual_new_args.inc" )

grid => parent_grid
CALL model_to_grid_config_rec ( grid%id , &
   model_config_rec , config_flags )
CALL couple_or_uncouple_em ( grid , config_flags ,.false. &
#          include "em_actual_new_args.inc" )
```

ARW Masked Interpolation



ARW Masked Feedback



Some Nesting Hints

- Allowable domain specifications
- Defining a starting point
- Illegal domain specifications
- 1-way *vs* 2-way nesting
- Nest logic in WRF source code
- Nest information in the Registry

What are those “usdf” Options

```
state real u ikjb dyn_em 2 X \
  i01rhusdf=(bdy_interp:dt) \
  "U" "x-wind component" "m s-1”
```

“f” defines what lateral boundary forcing routine (found in **share/interp_fcn.F**) is utilized, colon separates the additional fields that are required (fields must be previously defined in the Registry)

What are those "usdf" Options

```
state real landmask ij misc 1 -  \
  i012rhd=(interp_fcnm)u=(copy_fcnm)\
  "LANDMASK" "LAND MASK (1=LAND, 0=WATER)"
```

"u" and "d" define which feedback (up-scale) and horizontal interpolation (down-scale) routines (found in **share/interp_fcn.F**) are utilized

Default values (i.e. not a subroutine name listed in the parentheses) assume non-masked fields

At compile-time, users select options

What are those “usdf” Options

```
state real ht ij  misc  1  - i012rhdus  "HGT" \
  "Terrain Height"   "m"
```

“s” if the run-time option for smoothing is activated, this field is to be smoothed - only used for the parent of a nest domain, smoothing is in the area of the nest, excluding the outer row and column of the nest coverage

Whether or not smoothing is enabled is a run-time option from the namelist

Special IO Stream #2 Fields

```
state real msft ij  misc  1 -  \
  i012rhdu=(copy_fcnm)  "MAPFAC_M"   \
  "Map scale factor on mass grid" ""

state real msfu ij  misc  1 X  \
  i012rhdu=(copy_fcnm)  "MAPFAC_U"   \
  "Map scale factor on u-grid" ""

state real msfv ij  misc  1 Y  \
  i012rhdu=(copy_fcnm)  "MAPFAC_V"   \
  "Map scale factor on v-grid" ""
```